Reinforcement Learning in a Multi-agent Framework for Pedestrian Simulation

VNIVERSITAT DE VALÈNCIA

Memoria para optar al grado de Doctor presentada por:

Francisco A. Martínez Gil

Programa de doctorado de Informática y Matemática Computacional. Department d'Informàtica. ETSE

Universitat de València

Dirigida por:

Dr. D. Miguel Lozano Ibáñez, Dr. D. Fernando Fernández Rebollo

A mis padres, Paco y Emilia.

A Carmina, Clara y Elena.

Acknowledgements

First, I would like to acknowledge my supervisors Fernando Fernández and Miguel Lozano for the confidence that they have placed in me to develop this work. They have provided me with guidance and inspiration without which this thesis could have finished in the drawer of 'outlandish ideas'.

Secondly, I wish to thank the people from the Departament d'Informàtica for their contributions and the ideas that they have provided me with throughout these years. I would specially like to thank professors Wladimiro Díaz, Fernando Barber, Elena Díaz, Jesús Albert, Francesc Ferri, Salvador Moreno, Juan Domingo, Xaro Benavent, Vicente Cavero, Carlos Pérez and Francisco Grimaldo who helped me with the multitude of every day problems that a thesis creates. Along with professor Illés Farkas who kindly provided me with an effective version of the Helbing and himself's model for the crossing scenario.

I whish to thank senior researcher Juan Manuel Orduña for giving me the confidence to include me as a member of the GREV research group. Without his support this work would not have been presented in several international forums. Researchers Ignacio García and Ignacio Panach also gave me their confidence including me in different founded research projects. An important part of this work have been founded by spanish MEC under grant TIN-2009-14475-C04-04, European Commision FEDER funds under grant Consolider-Ingenio CSD2006-00046, TRA2009-0080, and University of Valencia under grants UV-INV-AE11-42609-20110575 and UV-INV-PRE-COMP13-115032.

Last, but not least, to my family, since this thesis would have not been possible without their patience and continuous encouragement and support. Carmina is (and probably will be) the person who has read this work the most. Her unconditional help is inside each and every page of it.

Francisco Martínez Gil

Universitat de València

July 2014

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Just another framework for pedestrian simulation with complex foundations, cumbersome adjustment and limited results? Yes,... It would be audacious to answer in other terms. Pedestrians, like fish, herds or swarms are interactive groupings of living organisms, and, therefore, naturally complex. Simulating complex systems where its components, in our case the individuals, locally interact is a challenging task by itself. First, the number of local interactions grows exponentially with the number of individuals, which makes a centralized control of limited usefulness. There are two approaches to this problem: i) to give each individual autonomy to manage the local interactions by him/herself, and ii) to forget the local interactions, abstracting the particles into a bigger structure and manage the group from a macroscopic point of view, similar to the way fluids are modeled. Second, there is a subtle and exciting problem. Local interactions in complex systems tend to generate structures. In real pedestrian groups, the interactions create structures in the behavior and then collective behaviors emerge. This phenomena of pedestrian self-organization is common in real life. For instance when lines of pedestrians are created to advance in a crowded shopping street, or when the individuals in a bottleneck divide spontaneously into two groups at the left and the right-hand sides of a door alternating their access (this is known as the zipper effect). This problem is reproduced manually by simulations based on the macroscopic approach. For simulations based on autonomous agents, this problem is challenging and its reproducibility indicates that the interaction model is correct or, at least, valid for that domain.

Facing this state of the art, some questions arose for people involved in this work: could we leave the burden of the model design to the responsibility of the individual actors? That is, what would happen if a group of embodied agents, with physical calibrated-like-pedestrians interactions, *learned* how to behave in a group to successfully navigate inside an environment with restrictions? Would they learn pedestrian-like behaviors? Would collective behaviors emerge? This work will attempt to answer these and other questions but with a modest goal in mind: the learned behaviors need to be realistic and plausible, not strictly real pedestrian behaviors.

1.1 Contextualization of the work

Pedestrian simulation has engaged the attention of researchers over the past few decades. Different technical areas, such as architecture, civil engineering, and game development can benefit from the simulation of pedestrians groups. For instance, in order to check the capacities of the facilities in a building, to prevent accidents and/or disasters, or to give realism to simulated urban scenarios. With the advent of computer graphics in the 1980's, the possibility of representing virtual pedestrians and groups with different purposes arose. One of the first attempts at simulating collective navigation is found in Reynolds' *Boids* (Reynolds, 1987). A Boid is a navigational entity that uses simple rules and the perception of its local dynamic environment to generate natural aggregate motion like flock of birds, herd of land animals or school of fish. Nowadays, there are several pedestrian simulators used in the architecture and urban planning fields to design spaces and facilities according to specific levels of service. Another field recently interested in pedestrian simulation is the "serious games" field. A serious game is a computer graphics game used to train the player in a skill (like managing a fire brigade). These systems need to recreate with increasing fidelity the real environments in which the trainee will develop the skill being learned. Secondary characters (such as pedestrians in a street) that convincingly simulate their roles without interfering with the main characters of the animation (Dignum, 2012) are then necessary.

There are two main approaches for the simulation of pedestrians according

to the level of abstraction: the macroscopic approach and the microscopic approach (Johansson & Kretz, 2012). In the first, the pedestrians are considered to be like particles and the model tries to reproduce macroscopic parameters such as flow, averaged speed or the main direction of the group. In the microscopic approach, the pedestrians are considered individually focusing on the local interactions of the pedestrians with their immediate environment. This approach is the most active nowadays in the simulation field because it seems to allow higher-level decision-making without major modifications of the basic behavioral model (Rindsfüser & Klügl, 2007). The decision-making mechanism is the center of interest of the present thesis and thus, the microscopic approach will be selected for the development of the general pedestrian simulation framework described later in this work.

Several microscopic pedestrian models have been developed for simulation. They can be classified according to how they model the individual behaviors. A first classification would include: the cellular automata models (Gipps & Marsjo, 1985), force-based models (Helbing & Molnár, 1995), rule-based models (Reynolds, 1987), and models based on psychological (Pelechano *et al.*, 2007; Sakuma *et al.*, 2005) and cognitive factors (Shao & Terzopoulos, 2005). Other models have been designed and calibrated using empirical data collected from video sequences or from experiments with real pedestrians (Daamen & Hoogendoorn, 2003; Robin *et al.*, 2009; Schadschneider & Syfried, 2011; Teknomo, 2002). Of especial interest in this work is the Agent-based model (ABM) (Musse *et al.*, 1998; O'Sullivan & Haklay, 2000). In the ABM's context, an agent is a computer system that is situated in some environment, and that is capable of performing *autonomous* decision-making in order to meet its design objectives (Wooldridge, 2013). An autonomous agent can take decisions, it is aware of the local environment and has a motivation in terms of a goal. Artificial intelligence (AI) techniques can be used to build a decision making module giving the agent flexible autonomous actions which imply reactivity (the agent can respond to changes), pro-activeness (the agent exhibits goal-directed behavior) and social activity (the agent interacts with other agents). In this work I will adopt this model by implementing a framework that will use embodied autonomous agents that learn to behave inside a multi-agent environment to achieve a navigational goal. An

embodied agent, also known as a situated agent, can sense physical interactions and constraints inside the simulated world. The embodied agents have additional intrinsic properties derived from their physical representation. They are capable of sensing their environment and they can perform actions in order to modify the environment. These actions generate new sensory stimulation, which, in turn affects future actions. In the words of Josh C. Bongard in his article (Bongard, 2013)

> In non-embodied AI, intelligence is something that arises out of introspection, while in robotics, the belief is that intelligence will arise of ever more complex interactions between the machine and its environment. This idea that intelligence is not just something contained within the brain of the animal or policy control of a robot, but rather is something that emerges from the interaction between brain, body and environment, is known as embodied cognition.

The use of AI techniques in the field of the pedestrian simulation or pedestrian modeling is relatively recent. The use of utility functions based on heuristics or in mathematical frameworks like the random utility theory (McFadden, 1981) is widely extended in this field to simulate specific behaviors such as pedestrian shopping. The problem of learning inside a Multi-agent system has been studied in the survey of Sandholm (2007). In the specific problem of decision-making applied to pedestrian modeling and simulation fields, works are scarce. The work of Zhu & Timmermans (2007) proposes the use of Genetic Algorithms (GA) to implement pedestrian shopping decision-making. In their paper, Kitazawa & Batty (2004) uses GA to emulate retail movements of shoppers in a large shopping center. Reinforcement Learning (RL) in navigation of autonomous agents has been considered mainly in robot domains over collaborative tasks (Fredslund & Matarić, 2002; Stone *et al.*, 2005). It is used relatively little in animation and simulation. Although a more detailed discussion can be found in Section 3.6, I mention here the work of Blumberg *et al.* (2002) who created an autonomous animated dog that was trained using RL to react to acoustic patterns, or the works of McCann & Pollard (2007); Treuille *et al.* (2007) focused on the use of RL for the selection of sequences in animations. At the time of writing, only

the study by Torrey (2010) concerning the use of RL to simulate crowds has similar insights. In this work I will use well-consolidated RL algorithms to get independent multi-level decision making modules to guide the embodied agents. Several important problems arise in common with the robotic world (autonomy of each learning agent, generalization and adequate representation of the state space, and efficiency of the learning algorithms among others) which will be considered in the forthcoming chapters.

Another issue addressed in this work is related to the multi-level behavior of the agents. Daamen observed that individuals make decisions following a hierarchical scheme: strategical, tactical and operational (Daamen, 2004). The destinations are chosen at the strategical level, the route choice is performed at the tactical level and the instantaneous decisions to modify the kinematic state are taken at the operational level. Several microscopic simulators that focuses on the reproduction of the local interactions function at the operational level (Robin *et al.*, 2009). However, due to the complexity of multi-agent collision avoidance, it is difficult to generate lifelike group motion following only local rules (Patil *et al.*, 2011). Most agent models separate the local interactions from the necessary global path planning. To do this, there are two main approaches. One is to pre-compute or user-edit a path-planning map that is represented as a guidance field (Patil *et al.*, 2011) or as a potential and velocity field (Treuille *et al.*, 2006). The other consists of separating the local and global navigation problems in a layered model (Sung *et al.*, 2004). If this separation takes place inside the agent model, it has the advantage that intelligent or psychological properties can be introduced to the agent behavior (Pelechano *et al.*, 2007). RL also permits the abstraction of tasks in a layered learning model (Stone, 1998). As a counterpart, this separation of tasks into different levels means that the emergence of collective behaviors is difficult to achieve. Therefore, a trade-off between control and realism must be set. The issue of multi-level behaviors in our framework will be studied in two experiments. Through these experiments, the presence of tactical and planning capabilities in the agents' learned behaviors will be detected indicating that the learning algorithms solve the navigational problems operating at different levels intrinsically, without specific software architecture.

To conclude, let me introduce the following thoughts included in the conclu-

sions of Renault *et al.* (1990) that link with the aim of this dissertation:

> As mentioned by Weizenbaum (1976), a real understanding may be only obtained by experimenting with the world and developing an internal database representing these experiments. A child builds his knowledge of the world by experimenting and learning. Weizenbaum states that it would be necessary for a robot to build its knowledge database of the world by exploring its environment, because the introduction by hand of appropriate data concerning the world is impossible. A robot walking, like a human being, in the everyday environment, does not exists and will probably never exists. But a society of synthetic actors living in a synthetic world with their specific behavior is for tomorrow or after-tomorrow. This is certainly a new interesting vivarium for artificial intelligence

1.2 Dissertation outline

This dissertation is divided into the following parts:

- Chapter 2 introduces the state of the art in pedestrian modeling and simulation. First, different models of pedestrians are introduced indicating their macroscopic or microscopic nature. Then, the main approaches in pedestrian simulation are described and related with the model or models in which they are inspired. The chapter concludes with a review of research and commercial simulation systems.

- In Chapter 3 the theoretical foundations of Reinforcement Learning are reviewed. The presentation is restricted to the areas related directly with the present work. In this chapter, the algorithms and techniques used in this dissertation are discussed. Not only learning algorithms are presented but also techniques for state space generalization and transfer knowledge are introduced. The chapter ends with a view of the application of reinforcement learning to the simulation field.

- Chapter 4 outlines the motivation and the objectives of this thesis.

- Chapter 5 describes the Multi-agent architecture of the framework. It begins with a functional description of the different modules that compose the framework. Then, the calibration of the physics module is addressed. Next to calibration, two experiments are carried out to validate the system.

- Chapter 6 describes two iterative algorithmic schemas based on Q-learning and Vector Quantization as the generalization method of the state space. These schemas allow the incorporation of techniques of knowledge transfer and the study of their impact inside the learning process. They are tested in two different scenarios and also compared with Helbing's social forces model. In this chapter, a study of the dynamics learned is performed using tools to analyze them at the micro and macro levels. Moreover, the question of the emergence of collective behaviors is considered.

- Chapter 7 presents new experiments using another learning configuration: Sarsa(λ) with tilecoding as the generalization method of the state space. This configuration also uses knowledge transfer techniques. The experiments are carried out with the aim of testing that the framework can operate at higher level (specifically route choice and path finding). Again, comparisons with Helbing's model are carried out. The problem of emergence of collective behaviors is also addressed with this learning configuration. In addition, a performance comparison is made between both learning configurations (Q-learning and Sarsa(λ)) in a specific scenario focused on the influence of knowledge transfer methods in the learning performance.

- Chapter 8 sets out the conclusions and indicates future work.

Chapter 2

State of the art in pedestrian modeling and simulation

Despite studies about pedestrian movement being more recent than other classic problems in urban planning and transportation, such as car and public transport, interest in them has increased dramatically in recent decades. Modeling and simulation of the dynamic and the behavior of pedestrians are interrelated, but their interests are different. While pedestrian modeling concentrates on the adjustement of the model to the data collected from studies with real pedestrians, the pedestrian simulation field is focused primarily on the appearance of the natural behavior of pedestrians. Despite these different interests, both activities come together, especially since the boom in computer graphics in the 1980s. Today, modelers have the opportunity to check their models in virtual environments with physical laws and specific requirements. Simulations in these realistic environments can be useful to check whether the model is able to generate the specific characteristics of real pedestrian movement. Otherwise, urban environment simulations need models to simulate pedestrians inside them in order to, at least, give realism to the scenario. In recent times, the requirements have increased with the new generation of video games and the emergence of a new industry in the computer graphics field dedicated to serious games. The simulated pedestrians have to be proactive (that is, they must have some kind of autonomy) and there is a need to be aware of the scene that is being rendered. Simulation is now im-

posing on models not only new capabilities of reproducing pedestrian movements but also psychological abilities such as decision-making or/and awareness. In this chapter, I will describe briefly the main approaches that have been developed in the two branches (modeling and simulation) of this field of study. The work presented in this thesis is related to the simulation of pedestrians because, as was stated in the introduction and it will be indicated in the objectives, the aim is to generate plausible pedestrian dynamics

2.1 Pedestrian modeling and simulation

Empirical studies of pedestrians groups and crowds began with the works of the psychologist Gustave Le Bon in the XIX century. Le Bon studied human crowds and multitudes from the point of view of psychology, specifically in subordination relationships, stating in his work *La Psychologie des Foules* (1896) that the individual personality in a crowd is submerged and then, the collective crowd mind dominates. This collective mind is characterized by being unanimous, emotional and intellectually weak. In the XX century the first studies began in the fifties with the work of Hankin and Wright in 1958 concerning passenger flow in subways and Older in 1968 studying the movement of pedestrians on foot paths in shopping streets. The evaluation methods initially applied were based on direct observation, photographs and time-lapse films. From the seventies, pedestrian studies acquired great importance. In that decade, the first important works supported by data appeared. Especially, Fruin's analysis of the level-of-service concept (Fruin, 1971a), the work of Jake Pauls in evacuation from buildings (Pauls, 1977), the studies of Templer concerning the movement of pedestrians on stairs (Templer, 1974) and the analytical formulas for crowds extracted from manually collected data of soldiers movements (Predtechenskii & Milinskii, 1978). In the eighties, the studies of pedestrians took two different directions: first the studies focused on the analysis and modeling of the movement of individual pedestrians, groups and crowds aided by the use of new technologies (mainly image analysis using videos and CCTV footage); a second direction, as a consequence of the exponential growth of computer graphics techniques from the 80s till now, was the simulation of pedestrians necessary to generate computer graphic simulations of

3D environments populated by collections of animated virtual humans (Pelechano *et al.*, 2008). The first branch had a direct application in engineering tasks for designing pedestrian facilities, calculating capacities, assistance in the egress design of airports and rail stations and making planning guidelines for emergencies and evacuations. The second one was important to allow the creation of virtual autonomous agents that offer realistic scenes in virtual worlds, video games, training systems and educational systems. While the first looks for the correlation of the measures provided by the model with the data of real pedestrians, the second pursues the resemblance of individual agents' behaviors with those of real pedestrians.

2.2 Pedestrian modeling

Pedestrian dynamics is difficult to characterize because it has many influences from various sources. Walking, contrary to other displacement models, is not associated with a vehicle, and the underlying infrastructure is highly heterogeneous (sidewalks, stairs, elevators, crossings, shopping malls, etc.). Besides, environmental factors influence walking (traffic lights, trees and public furniture, advertisements, shopwindows, etc.) as well as the atmospheric conditions (wind, rain, etc.). Demographic factors (percentage of elderly population and child population in the group), and sociological factors (handicapped persons) are also important. On the other hand, walking alone is different to walking in a group. The spatial presence of others affects the walking speed, and this relationship is not monotonic, in fact, low speeds are associated with very low and very high pedestrian volume in specific urban environments such as shopping streets. In addition, psychological facts and cultural conventions influence the collective movements of pedestrians. For instance, the space granted by a pedestrian in a group depends on the cultural and social characteristics of the interacting pedestrians (Sobel & Lillith, 1975). For these reasons, it is not possible to unify pedestrian dynamics under a single model. In this section, I review the main approaches, which are not necessarily computational models. Following the classic characterization of pedestrian dynamics, and also other kinds of vehicular traffic, the presentation distinguishes between microscopic and macroscopic levels (May, 1990).

2.2.1 Macroscopic characteristics of pedestrian dynamics

From the macroscopic perspective, pedestrian movements are described using magnitudes like flow, average speed and area module. This type of analysis derived from vehicular traffic studies, and does not consider direct interactions between pedestrians.

2.2.1.1 Fruin's levels of service

Fruin studied macroscopic pedestrian characteristics (Fruin, 1971a,b). The most important contribution of these studies was the concept of level of service (LOS) that was initially defined as a criteria for safety in places of public meeting. Fruin defined the different comfort levels for pedestrian movements based on these macroscopic magnitudes. Each level of service represents a range of operating conditions where level A represents the best operating conditions and level F the worst. The criteria to determine the LOS for pedestrian are based on objective parameters (such as the speed and the average space available) and subjective parameters (such as the pedestrian's ability to cross a pedestrian stream). Table 2.1 describes the LOS for pedestrians using macroscopic magnitudes.

Level of service	Space (m^2/ped)	Average speed (m/s)	Flow rate (ped/min/m)
A = Free Flowing	≥ 12.077	≥ 1.321	≤ 6.562
B = Minor Conflicts	≥ 3.716	≥ 1.270	≤ 22.966
C = Some Restrictions to Speed	≥ 2.230	≥ 1.219	≤ 32.808
D = Restricted Movement for Most	≥ 1.394	≥ 1.143	≤ 49.213
E = Restricted Movement for all	≥ 0.557	≥ 0.762	≤ 82.021
F = Shuffling Movements for all	≥ 0.557	≥ 0.762	variable

Table 2.1: Fruin's Levels of Service for pedestrians

Fruin applied his calculations to urban environments such as city streets under normal conditions. Thus, in other environments, Fruin's data do not adequately describe reality. For example, in crowded environments such as observations taken at the exits of Wembley Stadium, higher densities than the Fruin's data were observed in which the pedestrians moved without restrictions Still (2000).

2.2.1.2 The fundamental diagram of pedestrian dynamics

The study of the macroscopic parameters continued with the work of Predtechenskii & Milinskii (1978). It shows that the averaged speed of the flow of pedestrians is not only a function of the density but it is also of the type of path and the conditions in which the movement takes place. In that work, the descriptive capacity of the fundamental diagram of pedestrian movement is given an important role. The fundamental diagram for different situations or paths (horizontal paths, stairs (ascent), stairs (descent) and openings) under different circumstances: emergency, normal and comfortable conditions was designed.

In the following years, the researchers focused on a deeper study of the fundamental diagram. The most comprehensive survey concerning this subject is the work about free walking (where the pedestrians walk in a space without restrictions) by Weidmann (1993) who used 25 different studies of pedestrians under normal conditions to compose his general fundamental diagram. This is a reference in planning studies for estimating capacities of facilities.

The fundamental diagram of pedestrians in planar facilities has the following characteristics:

- The velocity decreases with growing density, although the relationship shows a non-trivial form (Schadschneider *et al.*, 2008).
- There are some important points that characterize the dynamics described in the diagram. The capacity [1] of a facility is directly defined by the maximum of the flow/density curve. The free speed corresponds to the mean maximum velocity. The critical density corresponds to the lower bound for unconstrained free walking. The jam density corresponds to the point of null speed and flow (Daamen, 2004).
- It can be described using an empirical analytic expression known as the Kladek formula (Lämmel *et al.*, 2009):

$$v_d(D) = v_f(1 - e^{-\gamma(\frac{1}{D} - \frac{1}{D_{max}})}) \tag{2.1}$$

[1] The capacity of a facility is defined as the maximum sustainable flow rate at which persons reasonably can be expected to traverse a point or uniform segment of a lane during a specified time period; usually expressed as persons per hour.

with γ $(1/m^2)$ a free parameter, v_f (m/s) the speed at free flow, D $(1/m^2)$ the actual density and D_{max} $(1/m^2)$ the density at which no flow occurs. Empirical studies showed good results with $\gamma = 1.913\ m^{-2}$, $v_f = 1.34\ m/s$ and $D_{max} = 5.4\ m^{-2}$, although it depends on the specific experimental conditions.

- The net-time headway is defined as

$$\hat{T} = \hat{D}/v = (\frac{1}{\sqrt{D}} - \frac{1}{\sqrt{D_{max}}})/v \tag{2.2}$$

  and settles at a constant value around 0.5 seconds (Johansson, 2009).

- The fundamental diagram can vary significantly in densities $< 0.2\ m^{-2}$ and $\geq 4\ m^{-2}$. In low densities, the pedestrians are free to choose their own speed of movement. With high densities, jams and crowds appear and the flow can be turbulent and chaotic and it depends on individual circumstances (den Berg & Bouvy, 1994).

Beyond these common properties, empirical studies with real pedestrians, performed under different conditions, reveal different shapes of the fundamental diagram. Figure 2.1 shows empirical fundamental diagrams that correspond with different studies in planar facilities used as references in planning guidelines. Although all the curves describe the dynamics of real pedestrians walking on a plane surface, they are different in shape (note for instance the differences in the ranges of density ρ for the different curves). Several explanations have been suggested including differences between uni and multi-directional flow (Navin & Wheeler, 1969), cultural and population differences (Johansson *et al.*, 2007) or psychological factors (Predtechenskii & Milinskii, 1978).

The fundamental diagram is a basic tool for engineering methods in the analysis of the real pedestrian flows, the design of pedestrian facilities and the study of infrastructures such as arenas or stadiums (Nelson & Mowrer, 2002; Schadschneider & Seyfried, 2009). Furthermore, it is used for the evaluation of pedestrian models (Helbing & Molnár, 1995) and is a primary test of whether the model is suitable for describing pedestrian streams (Hoogendoorn *et al.*, 2001; Schadschneider & Seyfried, 2009b; Steiner *et al.*, 2007). In the context of this work the

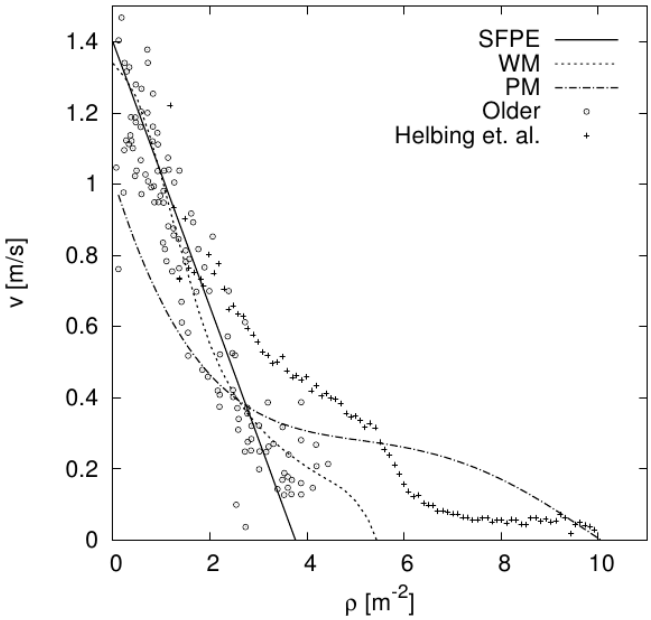


Figure 2.1: Empirical fundamental diagrams for pedestrians in planar facilities. The curves are extracted from planning guidelines (SFPE Handbook (Nelson and Mowrer 2002), PM (Predtechenskii and Milinskii 1978), WM (Weidmann 1993), Older (1968), Helbing, Johansson and Al-Abideen (2007))

fundamental diagram will be considered as a main tool to compare and analyze the simulated behavior obtained by the RL agents. A methodological description of the fundamental diagram is introduced in Section 5.4.2.

2.2.2 Microscopic characteristics of pedestrian dynamics

The microscopic level involves individual units with local characteristics such as speed, position and interactions. Unlike macroscopic characteristics, microscopic pedestrian characteristics are not well defined. One of the first ideas to characterize the microscopic dynamics of pedestrians was flow performance. It was used by the TRANSYT software to determine the performance of a traffic network Vincent *et al.* (1980). Helbing *et al.* (1997) proposed a flow performance based on two concepts: efficiency measure and uncomfortableness measure. Both measures are used as evaluation parameters to optimize pedestrian facilities and

they describe the interaction among pedestrians and between pedestrians and facilities. Teknomo (2002). describe them as follows: i) The efficiency measure $\tilde{E}$ calculates the mean value of the velocity component into the desired direction of motion in relation to the desired walking speed:

$$\tilde{E} = \frac{1}{N} \sum_i \frac{\bar{x}_i}{v_i^o} \tag{2.3}$$

where v_i^o is the intended velocity of pedestrian i, N the number of pedestrians and $\bar{x}_i = \frac{\sum_{t_1}^{t_2} \vec{v_i}(t)\vec{e_i}(t)}{t_2 - t_1}$ is the component of the velocity in the desired direction of pedestrian i. ii) The uncomfortableness measure $\tilde{U}$, reflects the frequency and degree of sudden velocity changes due to crashes or avoidance maneuvers

$$\tilde{U} = \frac{1}{N} \sum_i \frac{\bar{y}_i}{\bar{h}_i} \tag{2.4}$$

where $\bar{y}_i = \frac{\sum_{t_1}^{t_2} (\vec{v_i}(t) - \vec{g_i}(t))^2}{t_2 - t_1}$ and $\vec{g_i}(t) = \frac{\sum_{t_1}^{t_2} \vec{v_i}(t)}{t_2 - t_1}$.

Pedestrian flow performance can be measured through distances, and angles of moving direction. It can be valued over time as a speed (linear or angular), as an acceleration or as a rate of the acceleration (jerk). Other parameters have been proposed as candidates to measure flow performance such as pace index or variation of the walking displacement but most of them have no significant impact on the pedestrian flow (Teknomo *et al.*, 2003).

One of the most important problem for microscopic studies is data collection. Automatic systems have been designed to collect data from video files and from surveillance cameras as reported in the works (Hoogerndoorn *et al.*, 2003; Teknomo *et al.*, 2003) but their use is restricted to the research scope. Besides, parameter extraction is difficult in crowded places or in panic and emergency scenarios.

There is no hermetic separation between the macroscopic and microscopic models. Seyfried *et al.* (2005) experimentally analyzed the microscopic causes of the velocity decrease in the presence of medium or high densities, such as the frequency of passing maneuvers and internal crowd frictions. On the other hand, Kessel *et al.* (2002) proposed a microscopic model based on the fundamental

relationship between walking speed and crowd density.

2.2.3 Categorization of the existing pedestrian models

In pedestrian modeling, there are several categories: space representation, population representation, population behavior representation, purpose or availability (Kretz, 2007). Among them, population representation that divides the models between macroscopic and microscopic models has important implications for software design. The difference between these kind of models is the way to control the dynamic parameters by the component individuals.

The macroscopic models focus on the problem of space allocation for individuals. The individuals have no autonomy either to change their dynamic parameters (velocity, direction) or to control their interactions. A typical problem to solve in these macroscopic models is: given a number of pedestrians and a level of service, provide the space allocation (i.e. width of the facilities) and the flow and average speed in each facility. Of interest is the reproduction of the observed macroscopic data (mean speed, flow, density) and, therefore, groups of pedestrians, where these magnitudes are meaningful, are considered. The implementation of macroscopic models in computer devices has clear advantages with respect to microscopic ones: low computation time and reduced calibration and validation effort.

In microscopic models, each individual can control different parameters related with his/her own dynamics. The most representative is velocity. This control can be total (in case of an autonomous agent) or partial (defined as a constraint by means a desired velocity). The possibility of controlling their own velocity leads to the ability to control the interactions among individuals. Situations such as over takings, collisions or congestions can be managed. These models may be more suitable in cases where the geometry of the facility is unusual (and therefore its capacity is unknown) or when changes in pedestrian behavior affect the pedestrian flow (Hoogerndoorn *et al.*, 2003). On the contrary, many microscopic models do not take into account higher levels of behavior such as route choice and their applicability in simulations has to face computational efficiency problems derived from the individual control of the simulated pedestrians.

In the following subsections I present different microscopic and macroscopic models that constitute trends in the pedestrian modeling area. In this taxonomy, the classes are not hermetic and specific examples in one model can be included in another. For example the agent-based model is a generic model where different techniques borrowed from other models can be used, analogously Cellular Automata is also a generic model where other approaches can be implemented inside like floor fields (Nishinari *et al.*, 2004).

2.2.4 Discrete choice models

Discrete choice models are a family of macroscopic models that have been applied in the context of travel decisions (Ben-Akiva & Lerman, 1985; McFadden, 1981; Train, 2003). These models are based on random utility theory. Following the work of Bierlaire & Robin (2009), consider a decision-maker n who is performing a choice among a set $C_n \in J_n$ of alternatives. The decision-maker n associates an utility U_{in} with each alternative C_i and selects the alternative corresponding to the highest utility. The utility is modeled as a random variable to account for uncertainty due to various issues such as unobserved variables and measurement errors. The utility is discomposed in a deterministic term and a probabilistic error term, ε, so that

$$U_{in} = V_{in} + \varepsilon_{in} \tag{2.5}$$

and the probability that individual n is selecting the alternative i is

$$P_n(i|C_n) = Pr(U_{in} \geq U_{jn} \forall j \in C_n) \tag{2.6}$$

The specifications of V_{in} includes the selection of the attributes of i relevant to n, as well as the socioeconomic characteristics of n. The utility has a functional form. The complexity of the model comes from the distributional assumptions about the random variable ε_{in}. The most widely used model is the logic model, which assumes that the ε_{in} are independent across both i and n, and identically distributed. These assumptions lead to a simple and tractable formulation.

The set of choices that a decision-maker has to consider covers many dimensions of pedestrian behavior. Given an individual at a point of time, the main

choices to take into account are:

1. Activity choice. This choice focuses on what to do next. In the case of pedestrians, Hoogendoorn & Bovy (2004) distinguishes between the choice of an activity pattern performed at the strategic level of decision, from a scheduling activity performed at the tactical level. However, Borgers & Timmermans (1986) considers that the choice activity is not planned but triggered by stimuli in the pedestrian's environment.

2. Mode choice. This is the most traditional discrete choice model type. Two types are considered in the literature on pedestrian travelling. First, the usual transportation mode, where walking is one of the alternatives. For instance, Ewing *et al.* (2007) analyzes travel decision of students going to school. The second type focuses on the choice among stairways, escalators, or elevators while walking. This type of choice is typically small (less than a handful of alternatives). It is of increasing interest for health applications in general and overweight and obesity issues in particular (Eves *et al.*, 2006).

3. Route choice. The choice of itinerary is a critical dimension of pedestrian behavior. Route choice models are traditionally based on a network structure. Okada & Asami (2007), incorporated utility at nodes in a pedestrian flow model, and derived route choice probability using an aggregate logic model. Seneviratne & Morrall (1985) evaluated the factors affecting the choice of route and emphasize the importance of distance, while the level of service, safety or visual attractions appear to be secondary.

4. Choices of speed and next step. The choice of speed depends on the environment in which walking takes place. This type of choice can be integrated into the next step choice or can be taken independently. Many variables may explain the speed behavior and can be included in the model specification. Among the macroscopic type, flow and density are considered (Lam & Cheung, 2000) and also the type of environment such as crosswalks (Knoblauch *et al.*, 2007) and airport terminal corridors (Young, 2007). Among the microscopic type, overtakings, internal frictions and crashes, age and trip purpose are also relevant.

Discrete choice models of pedestrians have been successfully used in pedestrian shopping scenarios to explain behavioral reasons such as why the pedestrian chose one store rather than another (Zhu & Timmermans, 2007). An open source freeware named BIOGEME (Bierlaire, 2003) is available for the estimation of discrete choice models. It allows the estimation of the parameters of several models including nonlinear utility functions.

2.2.5 Cellular Agent models

Cellular Agent models (CA models) are a main class of computational microscopic models for pedestrians dynamics. One of the main characteristic of these models is the explicit representation of the environment as a lattice of cells whose state includes information about the presence and direction of individuals, and about environmental obstacles and relevant objects. Their dynamics are rule-based and usually stochastic. The transition probabilities p_{ij} are defined to one of the neighboring cells (i, j), where usually either Moore or von Neumann neighborhoods are used. The transition probabilities for a specific particle are determined by the position of other particles in its vicinity and they define the model. An example of the model specification can be seen in Figure 2.2.

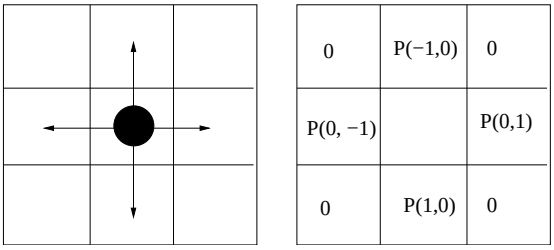


Figure 2.2: The possible directions of a cellular agent and the corresponding transition probabilities in a von Neumann neighborhood.

The following properties define CA models (Kretz, 2007): i) they are discrete dynamical systems ii) their update rules are local, that is, a cell's next state only depends on the neighbor cells and, iii) all the cells are updated synchronously and the update rules are identical for all cells.

CAs are widely used in traffic simulation. The simulation of a lane of vehicular traffic was proposed in the seminal work by Nagel *et al.* (1996b). Their

rule system was extended to a multi-lane model and has been applied to simulate traffic networks in the TRANSIMS project (Nagel *et al.*, 1996a). CA are basically one-dimensional. A new rule set was proposed in the work of Blue & Adler (2001) that adapted the model to pedestrian traffic and extended it to bi-directional pedestrian flows. In the work of Meyer-König *et al.* (2001) an example of adaptation of CA models for pedestrian flow is explained. The basic rules are:

1. The floor is divided in quadratic cells of length 0.4 m.
2. Each cell is empty or occupied by one person.
3. Individuals have their own characteristics reflected by a set of parameters.
4. The motion is described by their direction and walking speed and obeys universal laws.
5. Walking speed and direction might be altered non-deterministically with certain probabilities. This accounts for psychological and social factors not directly represented in the model.
6. The walking speed is at most 5 cells/second where every person has an upper limit v^i_{max}.
7. The positions are updated sequentially, where the current person is selected at random.

CA models have been used in different pedestrian navigational problems. The work by Burstedde *et al.* (2001) defines a CA capable of reproducing several collective effects of pedestrians such as lane formation, or the evacuation of a large room. The problem of pedestrians evacuations has been extensively studied with this model, and also mixed with other models, in different scenarios: buildings (Yang *et al.*, 2005), with obstacles (Varas *et al.*, 2007), considering forces (Wei-Guo *et al.*, 2006), fluid dynamics (Gipps & Marsjo, 1985) and using a bionics-inspired CA model (Kirchner & Schadscheneider, 2002).

2.2.6 Queuing models

Queuing theory is a branch of stochastic processes inside the operation research field. It is based on the concept that a queue or waiting line is formed when pedestrians need more service on arrival at service node than they are provided with (Rahman *et al.*, 2013). The theory tries to set up a model for the dynamics of the queues that in our context represent a pedestrian flow or a traffic flow in a lane. Using queuing theory, pedestrian macroscopic models have been developed to study pedestrian traffic flows and, in addition, the design of the physical systems accommodating these flows. One pioneering work was the model proposed by Lovas (1994). In this model, different pedestrian facilities were modeled as a network of walkway sections. Pedestrian flow in this network was modeled as a queuing network process, where each pedestrian is treated as a separate flow object, interacting with the other objects. The network models the environment, where the nodes can represent doors, rooms, intersections and the links can be corridors or other facilities. Because this model is concerned with flow control, it can be considered within the macroscopic type.

The basic entities which characterize a queuing model are: i) the arrival date, ii) the service mechanism iii) the queue strategy (e.g. first come first served) and iv) the number of service nodes. These properties are often referred to using Kendal notation that consists of several symbols (e.g. M/G/1). The first symbol is shorthand for the distribution of the inter-arrival times (e.g. Gaussian, Poisson); the second for the distribution of service times and the third indicates the number of servers. In this example M/G/1 represents a Poisson distribution for arrival times, a General distribution for the service and 1 server in the system.

Queuing theory has been used in the past mainly to describe traffic behavior at signalled and unsignalled intersections (Heidemann, 1997; Vandaele *et al.*, 2000). However in recent times, several queuing models have been proposed for different pedestrian scenarios. For instance, Li & Han (2011) proposed a grid-based model of a queue simulation system considering human physiology and psychology. It was capable of reproducing the traffic shock wave phenomenon effectively. This phenomenon consist of transition zones between two traffic states (congestion and free movement) that propagate through a traffic environment as

a wave. Shock waves are a fundamental property of road traffic congestion and can be seen by the cascading of brake lights upstream along a highway. The work by Kim *et al.* (2013a) models a cinema ticketing booth system. They use a micro-simulation software called Visim based on Helbing's social forces model to simulate pedestrian movements. Their queuing model controls the movements to reproduce a single queue with a multiple servers system of a cinema ticketing office. The results conform with the real analyzed data.

Queuing theory is able to calculate and predict the number of people waiting and the waiting time in the queue spaces. However, it cannot deal with heavy congestions and complicated movements of pedestrians. In large urban spaces, lots of kinds of pedestrian movements are merging and intersecting. This converts the queue spaces into circulation spaces that influence the dynamics of the queue formation. Queue theory focuses on the queue dynamics and does not consider external influences. More complex queue models have been developed. That of Okazaki & Matsushita (1993) takes into account other pedestrian behaviors outside the queue such as approaching queues and getting out of them.

2.2.7 Navigation fields based models

In Navigation-field based macroscopic models, the space is divided into cells where relevant navigational information is stored. One class of Navigation fields are the floor field models. The simplest approach consists of using static floor fields (Kretz, 2009) that are determined in the initialization phase of the simulation. The type of tessellation selected to discretize the floor is important. In Leng *et al.* (2014) a hexagonal tessellation with weights to compensate the non isotropy of the hexagon in the orthogonal directions is proposed. This compensation is important for modelling specific scenarios such as crossroads. The static navigation behavior often leads to unrealistic simulation results (i.e. pedestrians head very closely towards the congestions until they perceive the congestion).

Another approach takes inspiration from the motion of ants which is based on processes of chemotaxis, a chemical form of communication. Ants deposit so-called pheromones to mark their paths. A similar mechanism is used in the floor field model to take into account the interactions of pedestrians and those

with the infrastructure (Chowdhury *et al.*, 2005).

In Schadschneider & Seyfried (2009a) a dynamic floor field approach based on a CA model is presented. In this model, the probabilities of movement are encoded in the so-called matrix of preference. These probabilities are modified by two discrete floor field: a dynamic floor field D and a static floor field S. The first one represents a virtual trace left by moving pedestrians which has its own dynamics (in terms of diffusion and decay) and leads to broadening and dilution with time. The static floor S does not change in time and reflects the infrastructure. In the case of an evacuation, S describes the shortest path to an exit door. Seitz & Köster (2012) introduces the Optimal Steps Model in which pedestrians navigate along a floor field constructed by superposing escalar fields. Three scalar functions express the orientation towards a target, the need to avoid getting too close to neighbors and the need to skirt obstacles.

Another approach is presented by Hartmann (2010) that uses navigation fields to indicate the shortest distances to the pedestrian's target with respect to arbitrary metrics (e.g. metrics depending on the local terrain). The author proposes that if the metric correlates inversely with the expected speed, these distances could be interpreted as expected travel times. Based on this idea, the author presents a simulation of the shortest path *vs.* quickest path dilemma that will also be considered in my work in the experimental part. Using a distance metric weighted relative to the local pedestrian density, the agents tactically avoid congestions using the farthest door from the target.

When the information inside the navigation field represents directions that satisfy certain constraints, they are denominated vector fields. Gilman *et al.* (2005) presents the Dynamic Navigation Field (DNF) as a family of case-based reasoning algorithms for wayfinding. DNF produces vector fields according to the spatial situation of the particles. It gives important directional information to the particle inside the corresponding cell and only relevant information is calculated in the required spatial situations. This information can directly be the recommended direction or it can be used to calculate the desired velocity of the particle or the desired force to apply to it. However, dynamic fields require continuous updating which is computationally expensive, particularly for large domains. A general approach to vector fields and their uses in planning can be

found in the book by (LaValle, 2006).

2.2.8 Fluid-dynamic based models

Under this name, different macroscopic models exist. What they have in common is the use of mechanics-derived equations of fluids (Navier-Stokes equations of fluid-dynamics) to represent traffic flow. Therefore, it is assumed that a similarity between fluid dynamics and different types of traffic dynamics exists. Some of the most popular are gas-kinetic equations. They were first used for the description of traffic flow (Prigogine & Herman, 1971) and, some time after, were adapted to describe pedestrian flows. However, the models that use these equations imply momentum conservation which is not the case in pedestrian collisions. The gas-kinetic formulation of pedestrian behavior with Boltzmann-like equations has some analogies with the description of ordinary gases, but it takes into account the effect of pedestrian intentions and interactions.

In order to understand the type of similarities found between gases and pedestrians, let's take an example from Helbing (1992a) that describes a dance floor scenario. On a dance floor like that of a discotheque, two types of motion can be found: one type represents individuals who want to dance, the second type represents individuals who look at the dancers and do not want to move, although they remain on the dance floor. The first type intend to move with high velocity variance θ_h and they can be assimilated, by analogy, with particles with high temperature. The second type intend to have low velocity variance θ_l, that is, they have low temperature. Dancers and spectators are in equilibrium only if the mutually exerted pressure ($P = \rho\theta$) of both groups agrees ($P_h = P_l$). As a consequence, the dancers are expected to demonstrate a lower density (ρ) than the spectators ($\rho_l < \rho_h$). This phenomenon can actually be observed in real scenarios (see Figure 2.3).

Henderson was the first to apply gas-kinetic and fluid-dynamic models to empirical data of pedestrian crowds (Henderson, 1974). Observing the movements of students on a campus as well as children in a playground, he realized that their motion fits the Maxwell-Boltzmann distribution that describes particle speeds in idealized gases. In order to apply this theory to pedestrians, he had to make

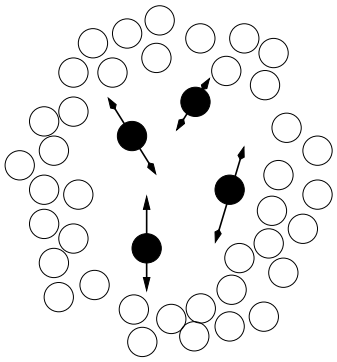


Figure 2.3: Distribution of dancers and spectators on a dance floor, as explained in the work of Helbing (1992a)

some assumptions and restrictions e.g. the crowd fluid had to be homogeneous, meaning that each particle (pedestrian) must have the same mass and probability density function for velocity (Sahaleh *et al.*, 2012). However this model assumes the conservation of energy and momentum which are not true for interactions between pedestrians. Helbing (1992a) proposes a better fluid-dynamical based description of pedestrian movement where anisotropies of pedestrian interactions and the preferred direction of motion are included. In this work, differential equations for the spatial density, mean velocity and velocity variance of motion types are proposed. These equations resemble those of ordinary fluids. The Hughes model of flow continuum for crowds (Hughes, 2003) is based on well-defined observations (hypotheses): i) the speed of the pedestrians is determined solely by the density of surrounding pedestrians ii) pedestrians have a common sense of the task that they face to reach their common destination and iii) pedestrians seek to minimize their estimated travel time. This model has been applied especially to provide assistance in problems characterized by high density crowds such as the annual Muslim Hajj, or to understand the behavior of the English and French infantry at the battle of Agincourt (1415).

The gas-kinetic and fluid-dynamic models can include other forces to represent pedestrian's intentions. These forces are in charge of changing the pedestrian's type of motion. They are guided by stochastic laws which are parametrized to represent a demand for commodities, location of stores or city center entry points (Helbing, 1992a). A hybrid model of this kind is presented in the

work Helbing (1992b).

2.2.9 Social force model

The "social force model" of Helbing & Molnár (1995) is a microscopic model where each individual moves as a result of a instantaneous local force which is the result of several forces (external and internal) that actuate in the pedestrian. The social force is not exerted by the environment on a pedestrian's body. Rather, it is a quantity that describes the specific motivation to act. The resulting force is a reaction to the perceived information that the pedestrian obtains about the environment. Following the explanation described in Helbing & Johansson (2009), the model assumes that each individual α is trying to move in a desired direction $\vec{e}^0_\alpha$ with a desired speed v^0_α, and that he/she adapts their actual velocity $\vec{v}_\alpha$ to the desired one $\vec{v}^0_\alpha = v^0_\alpha \vec{e}^0_\alpha$, within a certain relaxation time τ_α. The dynamics of an individual α is described by the equation

$$\vec{f}_\alpha = \frac{1}{\tau_\alpha}(v^0_\alpha \vec{e}^0_\alpha) + \sum_{\beta(\neq\alpha)} \vec{f}_{\alpha\beta}(t) + \sum_i \vec{f}_{\alpha i}(t) \tag{2.7}$$

where the terms $\vec{f}_{\alpha\beta}(t)$ and $\vec{f}_{\alpha i}(t)$ denote the repulsive forces describing attempts to keep a certain safety distance from other pedestrians β and obstacles i respectively. The first term of the sum represents a force that corrects the deviation from the desired velocity (due to an avoidance process or a necessary deceleration process) within a certain relaxation time. The repulsive forces to avoid contact or collisions with other pedestrians or objects are implemented as gradients of a potential field. The potential field for the repulsive force between pedestrians α and β, $V_{\alpha,\beta}[b]$ is a monotonic decreasing function of $b = b(\vec{r}_{\alpha,\beta})$ with potential lines having the form of an ellipse that is directed into the direction of motion. Thus, the equation 2.8

$$\vec{f}_{\alpha,\beta}(\vec{r}_{\alpha,\beta}) = -\nabla V_{\alpha\beta}[b(\vec{r}_{\alpha,\beta}]_{\vec{r}_{\alpha,\beta}} \tag{2.8}$$

is the repulsive force between two pedestrians which is typically exponential. A similar formulation is developed for the repulsive force between a pedestrian

and an object.

The social force model has been the basis for the development of other approaches. In the work by Pelechano & Badler (2006), it is used as a low-level local motion controller that is a part of a more complex model of a crowd were individuals take on different roles, such as trained personnel, leaders and followers. This model is used to study a scenario of a building evacuation. In the study by Zainuddin & Shuaib (2010), the model is modified to incorporate decision making capabilities to study unidirectional flows of pedestrians. The work by Mehran *et al.* (2009) proposes the use of this model to detect abnormal behaviors in crowd videos. For this purpose, a grid of particles is placed over the image, and it is advected using the space-time average of the optical flow. The interaction forces of the particles are estimated using the social force model which create a force pattern that can be analyzed in terms of normality. Another proposed model is the centrifugal force model (Yu *et al.*, 2005). It consists of a new expression for the repulsive force which differs from the social force model and its variants. It considers both the headway $\vec{R_{ij}}$ and the relative velocity v_{ij} among pedestrians as part of the definition of the repulsive force

$$\vec{F_{ij}^{rep}} \propto f(v_{ij}, \|\vec{R_{ij}}\|^{-1}) \frac{\vec{R_{ij}}}{\|\vec{R_{ij}}\|} \tag{2.9}$$

This repulsive force reflects new interesting ideas (Chraibi *et al.*, 2011): i) the force is anisotropic since its range of influence is reduced to the range of vision of pedestrians (180°) and, ii) it takes into account the influence of the relative velocity, i.e. faster pedestrians in front of slower pedestrians do not affect their movement.

The social force model has been considered to be one of the more successful microscopic models mainly due to its versatility. It supports the addition of different forces that can represent different pedestrian's social or psychological motivations. For example, the tendency to keep away from dangerous places is reflected by repulsive forces, while the effect of a stage, or of a window display can be described by attractive forces. The same applies to the tendency of group or family members to stay together (Helbing *et al.*, 2005). It has also been adapted to study specific problems such as escape panic and evacuations (Helbing *et al.*,

2000). Another interesting effect that has been successfully modeled inside the evacuation context is "freezing by heating" (Helbing & Johansson, 2009) where blocking situations appear inside a corridor when groups of pedestrians walking in opposite directions experience stress and the individuals begin to fluctuate in their behavior (the individual is getting nervous). When this situation gets worse, the individuals block each other and the created congestion becomes a frozen situation where no individual is capable of going ahead.

Despite their popularity, the social force model and in general force-based models have problems derived from their Newtonian formulation. Chraibi *et al.* (2011), who presents an extensive enumeration of different force-based models, indicates two main problems. The first one is derived from Newton's third law which states that two particles interact by forces of equal magnitudes and in opposite directions. For pedestrians this is unrealistic since normally the collision forces between pedestrians are not conservative (e.g. a pedestrian normally does not react to another pedestrian who pushes into a queue). The second problem emerges from the assumption that forces acting on a pedestrian are additive according to the superposition principle of forces. This can lead to undesired effects when modeling pedestrian dynamics, especially in dense situations (in form of high velocities). Further problems are related when describing particles with inertia using the Newtonian model, leading to overlapping and oscillations of the modeled pedestrians.

2.2.10 Agent-based models

The Agent-based model(ABM) (Musse *et al.*, 1998; O'Sullivan & Haklay, 2000) is a general-purpose microscopic model which uses agents that interact within an environment. In the ABM, the agents are in charge of the simulation of the behaviors of the studied phenomena; it is, therefore, a microscopic approach. The basic characteristics of this model have been summarized by Bonabeau (2007): i) agents can exhibit complex behavior patterns and provide valuable information about the dynamics of the real-world systems that they emulate; ii) agents provide a natural way to describe complex systems of individual entities with autonomous behaviors (like vehicle traffic, crowds) and, iii) the natural way of implementing

the ABM in pedestrian simulation is through a Multi-agent System, where each virtual agent has encapsulated different behavioral levels (reactive, pro-active) and exploits them in an autonomous way. ABMs can be considered models of complex systems. The ABM approach considers that simple and complex phenomena can be the result of interactions between autonomous and independet entitites. Thus, agents and ABMs should not only be considered as a technology, but also as a modeling approach that can be exploited to represent some system properties that are not simply described as the sum of their members' properties or functionalities (Bandini *et al.*, 2009).

Of particular interest for this work are the intelligent agents. They have an autonomous decision making module which creates the particular behavior of the agent. Wooldridge (2013) distinguishes among four types of architectures for intelligent agents: i) *logic based agents*–in which decision making is realized through logical deduction, ii) *reactive agents*–in which decision making is implemented as a direct mapping from state to action, iii) *belief-desire-intention agents*–in which decision making depends upon the manipulation of data structures representing the beliefs, desires and intentions of the agent and, iv) *layered architectures*–in which decision making is realized via various software layers that represent different levels of abstraction. These types of intelligent agents are also named *rational agents*. A rational agent is able to perform independent and autonomous actions in order to meet its design objectives, making good decisions about what to do (Wooldridge, 2000). The concept "making good decisions" implies the maximization of a utility function. Finding the optimal solution of a function or the optimal sequence of actions is often a problem with intractable computational complexity, therefore, modeling behavior based on full rationality may be impractical (Rosenfeld & Kraus, 2009). In recent years, the use of agents with bounded rationality has been extended in Computer Science. Bounded rational agents do not find the optimal behavior, but rather the non-optimal behaviors that can fulfill their goals. The agent ceases to be an optimizer and becomes a satisfier who seeks good-enough solutions instead of optimal ones. This thesis is related with the concept of bounded rationality because reinforcement learning algorithms find optimal solutions only after infinite experiences in the environment.

ABM has become very popular over the last decade. One of the areas in which it has been extensively used is in social sciences, where it is used to model population behaviors (Gilbert, 2007). It has been applied to the study of racial segregation in American cities (Schelling, 1971), to the understanding of the development of political opinions (Deffuant *et al.*, 2000) or to examining the interactions among different factors (friends, advertising, fashion, etc.) that influence customers to buy a product (Izquierdo, 2007). In the case of pedestrian simulation, *embodied agents* (autonomous virtual agents capable of physical interactions) represent pedestrians bringing together different levels of operation (reasoning, reactive and pro-active behavior) and different levels of interaction (coordination, negotiation). The ABM approach is an abstract approach that has, many times, included other models creating hybrid models. Examples of this mixture can be found in the works by Batty (2003, 2005), which uses a CA model to define fine scale movement at the pedestrian level in terms of the relationship with neighbor pedestrians, these movements take into account factors such as visibility, collision avoidance or attributes of other pedestrians. In the work by Ward (2007), a model of Covent Garden Market in central London is presented. This ABM uses also a simple social-force model in which agents have certain tasks to perform such as shopping and entertainment.

Inside the ABMs, an important approach is the Belief-Desire-Intention agent architecture (BDI agent). The philosophical fundation of BDI has its roots in practical reasoning. Practical reasoning is defined as reasoning towards actions as opposed to theoretical reasoning, which is reasoning about beliefs. The key concepts in BDI architecture are: i) beliefs: what I know about the world; ii) desires: what I want to do; iii) intentions: how I plan to fulfill the desires. A nice feature of this architecture is the ability to act in both a reactive and proactive manner. A BDI agent is a motivated agent that has an internal representation of the situation and is capable of dynamically changing his/her interests (re-planning) depending on environmental circumstances or interactions with other agents. Several works have applied BDI agents to the field of pedestrian simulation. The work by Ronald & Sterling (2005) ilustrates how BDI architecture adapts naturally to the representation of a pedestrian, where the beliefs can represent useful information for route planning (e.g the congested main street at lunch

time), the desires can represent places to go and the planning is developed by the intentional module. However the authors recognize that it is difficult for this model to handle continuous events such as stepping. In the paper by Shendarkar *et al.* (2008), a methodology involving Virtual Reality with BDI architecture to construct a crowd simulation is presented. The realistic attributes that govern the BDI characteristics of the agents are reverse-engineered by conducting human-in-the-loop experiments inside a virtual environment. The work by Okaya & Takahashi (2011) applies a BDI model in which human relationships affect at the stages of the sense-reason-act cycle of agents, and adopts a social forces model to build the agent intentions. This model is used to generate emergent behaviors in a crowd evacuation as a result of interactions in the crowd.

2.2.11 Optimization-based models

Under this name, several approaches are gathered. They have the common property of representing the goal or behavior to achieve as a function, commonly named utility function, that has to be optimized. The principle of least effort from Zipf (1949) (applied to the pedestrian context) proposes the idea that real pedestrians' trajectories are created by an optimization process of energy. A physical consequence of this principle is that pedestrians minimize metabolic energy when walking at roughly 1.33 m/s, as has been verified in observational studies (Henderson, 1971). Although other models intrinsically use a notion of optimization[1], I will focus in this subsection on the approaches that explicitly uses a function as a model of the behavior that has to be optimized. The scope of the model (macroscopic or microscopic) depends on the application of the utility function. If the utility function is applied to all the pedestrians, based on location criteria, the model will be macroscopic. On the contrary, if the utility function is defined for the individuals, as an individual optimization problem, the model will be microscopic.

In the work by Hoogendoorn & Bovy (2004), the velocities of the pedestrians are calculated optimizing an utility function $W_j(t, \vec{x})$ that reflects the minimal

[1]For instance, the models that use potential fields or vectorial fields, use implicitly a notion of optimization when they calculate the gradient of the field in a point of the space.

expected route cost of walking from the current location $\vec{x}$ at instant t to the destination area. Some constraints are imposed to model real circumstances such as incurring a penalty (ϕ_0) when not arriving at the destination before the terminal time t_f or when arriving too early (given by a function of the final arriving time $\phi(T, \vec{x})$). The NOMAD commercial pedestrian simulator has been built with Hoogendoorn's model. In the context of route choice, Ramming (2002) proposes a utility function to generate a route choice set in combination with simulation. Simulation is used to calculate route utilities according to the defined utility function. This includes perception parameters and weights representing preferences, where the route with the highest utility is added to the choice set.

The work that implement Zipf's principle of least effort to model pedestrian dynamics is that reported by Guy *et al.* (2012). The authors propose a model based on the optimization of the caloric energy of pedestians to develop a global navigation system that avoids collisions. For a given trajectory Π, the energy used by a person is dependent on the squared velocity and can be modeled as

$$E(\Pi) = m \int_{\Pi} (e_w |\vec{v}|^2 + e_s)\, dt \tag{2.10}$$

where e_w captures how efficiently calories are used and e_s is a person's rate of energy consumption when standing still. The minimization of Equation 2.10 under certain velocity restrictions yields a least-effort trajectory that avoid collisions among pedestrians. Moreover, the authors demonstrate that the model is capable of generating emergent phenomena, specifically lane formation or the arching congestion around a door or passage as the individuals try to come as close as possible to the exit in order to minimize the time spent in a congestion.

2.2.12 Models for Crowds

A crowd is a group of individuals who temporarily share the same place and focus. A common observation on human crowd behavior is that an individual may behave quite differently in a crowd when compared to acting individually (Zhou *et al.*, 2010). Therefore, modeling crowds is a specific problem that has its own characteristics. Contrary to macroscopic models, a crowd can be heterogeneous, that is, it can consists of people with differing goals and behaviors. This means

that the individuals within these crowds must navigate to their goals despite potentially congested environments and the conflicting paths of others (Guy *et al.*, 2012). In addition, the dynamic of a crowd depends on the dynamics and the local organization of the individuals, which implies that crowds present specific emergent patterns in determined scenarios. Despite these considerations, the explicit representation of many crowd models in the literature are actually adaptations of the previous revised models of this section (especially microscopic models). Many works revised in the previous subsections that consider groups of pedestrians (especially those related with the social force model and the agent-based model) are considered in this context as crowd models. In fact, they often appear in the introductions of papers related to crowds. I will not review those cases that have been mentioned previously.

A revision of crowd models is made by Zhou *et al.* (2010). It proposes a categorization of crowd models based on the size and the timescale of the crowd phenomena of interest. Attending the first criteria, crowds may consist of thousands (huge-sized), hundreds (medium-sized) or tens (small-sized) of individuals. The crowd size determines the types of approaches used to model a crowd. Huge-size crowds treats the crowd as a whole and focus on the global trend of the crowd using macroscopic approaches. With respect to medium-sized and small-sized crowds, they are modeled as complex systems whose dynamics results from local behaviors of individuals and their interactions with the neighborhood. The dynamics of a crowd is characterized by three factor categories:

1. Physical factors which refer to external characteristics of an individual such as position or velocity. A crowd model that considers only physical factors aims to investigate how the movement of an individual affects the group.

2. Social factors like culture, norms or leadership. Social studies are the areas interested in these kinds of factors. An example is the work by Pan *et al.* (2005) that used a multiagent-based framework to demonstrate some emergent human social behaviors (competing, queuing and herding).

3. Psychological factors such as emotion, happiness or sadness that play an important role in human decision making. The appropiate framework for these model is the agent-based model, which considers that individuals have

their own autonomous decision making module, and the layered models which incorporate an emotional or psychological layer (Gratch *et al.*, 2009; Pelechano *et al.*, 2007).

I will follow this characterization of the crowd dynamics to group different studies concerning crowd modeling.

In the group of crowd models that focus on physical factors, previously revised models of other approaches that center on crowd problems can be found. A non-exhaustive list is: (Burstedde *et al.* (2001); Yang *et al.* (2005) in CA models, Hartmann (2010) in navigation-field models, Henderson (1974) in fluid-dynamics based models, and Helbing & Johansson (2009) in the social-force model. Also, the studies concerning evacuations or panic can be included in this group because they are focused on group patterns and collective behaviors directly derived from individual behaviors. Examples of these works are Helbing *et al.* (2005); Waldau (2002).

In the group of crowd models that include social factors, the work by Manenti & Manzoni (2011) models groups of pedestrians using fundamental elements derived from sociology. Focusing on the primary structure of local human groups, they apply methods of network analysis to identify relevant structures in the crowd. This approach is based on the concept of the crystal group which states that a crowd is made up of small and rigid groups of individuals, strictly delimited and of great constancy. The groups do not disappear inside the crowd because a static relationship exists among its components: family, friendship, working and so on. The model uses anthropological and sociological elements to study the dynamics of these crystal groups inside the crowd. Another work that considers social group structures inside the crowd is that of Qiu *et al.* (2010) which presents a MAS-based model in which these social group structures exploit inter and intra relationships.

In reference to the crowd models that consider psychological factors, Fridman & Kaminka (2010) proposes a cognitive model for crowd behavior based on Festinger's Social Comparison Theory (SCT). This is a social psychology theory from the 1950's which proposes that humans, lacking objective means to evaluate their state, compare themselves to others who are similar. The authors implement an algorithmic framework for SCT and demonstrate, empirically, that it

generates behaviors more in-tune with human crowd behaviors than the existing non-cognitive models. Moussaïd *et al.* (2011), proposes a cognitive science approach to the problem. Pedestrians are modeled using behavioral heuristics using two simple cognitive procedures (based on visual information) to adapt their walking speeds and directions. This model predicts the emergence of self-organization phenomena such as lane formation and stop-and-go waves.

Other scientific fields like ethology can provide ideas concerning this issue. Collective patterns of crowds are not restricted to humans and can be observed in other biological systems. In the work by Shiwakoti *et al.* (2011), a mathematical model for crowd evacuation under panic conditions derived from collective animal dynamics is proposed. Specifically, the model is developed and validated by data from experiments with panicking Argentine ants (*L. humile*). These ants are reported to display a natural evacuation process because in their natural habitat (Paraná river banks in South America), flooding regularly forces colonies to evacuate their nests and seek refuge in trees. The experimental investigation of human crowd stampedes and evacuations is a problem as most experiments are too dangerous to perform. The authors justify the use of animal models because data from animals (for example insects) is easy to get and, being living creatures, they are more life-like than equations.

2.3 Pedestrian Simulation

The microscopic simulation of pedestrians took off in the mid 1980s mainly due to the increment in computing power. The pioneer works on virtual humans made a computational approach to this problem possible. Among those early works, Korein & Badler (1982) focused on goal-directed movements of articulated structures while Zeltzer (1991) defined frameworks for integrating the different representation levels (physical, behavioral, social) for virtual humans. In this line, the concept of *behavioral animation* that will be assumed and/or developed in many microscopic models presented in the following subsections is important. This concept assumes a virtual human, or individual, immersed in a virtual environment where he/she is able to perceive and interact (Monzani *et al.*, 2004). In the behavioral animation, each individual has his/her own personality and internal

motivations for acting. One of the best known application that uses this kind of animation is the computer game 'The Sims', which focuses on the simulation of virtual humans in everyday life. Many revised approaches in this section use some kind of behavioral animation to give autonomy to the simulated individuals.

Compared with vehicle simulation, which is one-dimensional, pedestrian simulation is bi-dimensional. This increase in dimensionality results in more degrees of freedom which requires more computational power. It has three implications in pedestrian modeling, following the arguments of Johansson & Kretz (2012): i) pedestrian modelers have to balance their creativity in using mathematical tools with respect to computational costs; ii) the refinement of the model in terms of temporal and spatial resolution can be tuned depending on the computational power available; and, iii) there is a trade-off among computational time, model complexity and scale, and accuracy of results.

In this section I will revisit models mentioned in the previous section but considering them from a computational perspective. As there is not a clear separation among models, many works should be included in different classes, especially the agent-based model which could include other approaches such as layered models. I include crowd simulation as a separate approach due to its importance inside the general problem of pedestrian simulation. However, many works described in other subsections refer to (or are applied to) human crowd simulation.

2.3.1 Rule-based simulations

In rule-based simulations, the agents are directed by decisions triggered by rules. In different works, the rules control different behavioral levels, from physical maneuvers to social or cognitive behaviors. If the rules are centralized in a decision module common to all the individuals, the system follows a macroscopic approach. Such kinds of rule-based approaches are useful in pedestrian flow studies for transport facilities. For instance, Seer *et al.* (2010) defines a macroscopic rule-based system to control the flow of simulated pedestrians in critical areas such as subway stations platforms close to the main soccer stadium in Vienna.

The seminal example of local rule-based microscopic individual simulation is Reynold's Boids (Reynolds, 1987). His system is based on a particle model, where

each individual (boid) has its own orientation and follows specific rules. The rules describe how a boid maneuvers based on the positions and velocities of its neighbors. The basic rules are separation, alignment and cohesion. With these simple rules, Reynolds simulated several groups of animals (flocks, herds) with a realistic appearance in terms of group navigation. This kind of behavioral animation is almost impossible if a traditional approach is used. In Reynolds' approach, each member of the group decides its own trajectory without animator intervention. The same author applied a similar rule-based approach to the specific problem of pedestrians (Reynolds, 1999).

The rules do not always control the basic movements of the pedestrian. Musse & Thalmann (1997) defined a rule-based model that allows virtual humans inside a crowd to switch groups based on sociological factors. Each group inside the crowd has its general behavior (which is determined by social rules such as polarization, domination or adding), but individual behaviors are created by a random process using a probability distribution characteristic of the group. This means that there is a trend shared by all the individuals of the same group although each member of the group has his/her own personal behavior.

Other examples include the rule system inside the decision making module of an agent, which also has other cognitive capabilities. For example, Shao & Terzopoulos (2005) used a rule-based system for the reactive part of their autonomous pedestrians while a planning algorithm is used to find the shortest path in the navigational level.

2.3.2 Vision-based approaches

J.J. Gibson is the grandfather of active vision. He stressed in pre-computational terms the importance of modeling the active observer situated in the dynamic environment (Gibson, 1979). The concept of active vision was developed by Aloimonos *et al.* (1988), and includes the set of problems in vision under the assumption that the observer is active (an observer is called active when engaged in some kind of activity whose purpose is to control himself or something). One of the first computer applications that used active vision inside software-built artificial animals to give them the ability to move, perceive and understand their virtual

world was Terzopoulos' artificial fishes (Terzopoulos & Rabie, 1997). His fishes are autonomous virtual robots with active perception systems that autonomously control their eyes and muscle-actuated body.

The problem of animating a synthetic pedestrian may be divided into two parts (Renault *et al.*, 1990): i) provide the actor knowledge of his/her environment, and ii) make react him/her within this environment. The first problem consists of creating an information flow from the environment to the actor. The approaches that use some kind of visual flow (real or synthetic) to provide information to the actor about the environment are vision-based approaches. In Renault *et al.*'s work, the authors propose synthetic actors with synthetic vision capabilities. Synthetic vision has advantages over real image understanding, essentially skipping the problems of pattern recognition and distance detection. Synthetic vision-based pedestrians are able to avoid obstacles inside a narrow corridor indicating that this capability is effective in providing a stable solution to the 'awareness of the environment' problem. This proposal is also used in the work of Noser *et al.* (1995) that extends the idea combining synthetic vision with a dynamic octree that serves as a 3D visual memory and allows an actor to memorize his/her environment.

In the paper by Kuffner (1999), the author combines a low-level path planner, a path following controller and 3D visual perception rendering hardware to guide the virtual humans inside the 3D virtual environment. The graphics rendering hardware is used to simulate the visual perception of a character so that it reacts in response to obstacles in its visual field. This synthetic vision system provides a feedback loop to the overall navigation internal model that is updated as new sensory information arrives, and a new navigation plan is computed if necessary. Peters & O' Sullivan (2003) has developed a system for the automatic generation of bottom-up visual attention behaviors in virtual humans. Bottom-up visual attention refers to the way in which the environment solicits one's attention without regard to task-level goals. Their system endows virtual agents with the ability to pay attention to their surroundings. The approach of Ondrej *et al.* (2010) explores a vision-based model for collision avoidance between walkers that fits the requirements of interactive crowd simulation. The virtual pedestrians detect future collisions as well as their dangerousness from visual stimuli. The

authors show empirically that their visual system reinforces the emergence of self-organized patterns of walkers. In addition, the visual system improves the overall efficiency of the walkers' traffic and allows avoiding improbable locking situations.

Inside the area of commercial pedestrian simulators, Massive software agents (http://www.massivesoftware.com) are provided with synthetic vision, but this is not used to directly control the movement of the agent.

2.3.3 Social force based simulations

The model of social forces has been simulated on computers for a large number of interacting pedestrians confronted with different situations. Despite the fact that the proposed model is simple, it is capable of describing observed phenomena (e.g. simulation of escape panic or emergent collective behaviors) in a realistic way.

One simplified approach to this model uses forces or potential fields to direct the navigation of the particles. For instance, Heïgeas *et al.* (2003) used a particle system to simulate a crowd that congregates in an ancient greek Agora. The interactions among individuals are modeled as physical forces, specifically mass-damp-spring force systems to create repulsing forces between walkers and between a walker and an object. The authors observe the formation of very basic patterns like jamming and flowing. Wagoum *et al.* (2010) studied several types of force-based models from the perspective of computational efficiency to simulate crowds for an evacuation assistant. Specifically, they studied Message Passing Interface (MPI) and the Open Multi-Processing application programming interfaces. In the work by Ali & Shah (2008) three different potential fields to track the movement of the particles inside a crowd are proposed: i) a static floor field that specifies the areas of the scene which are attractive in nature (e.g. an exit), ii) a dynamic floor field that defines the immediate behavior of the crowd in the vicinity of an individual and, iii) the boundary floor field that defines the influences exhibited by the barriers in the scene (e.g. walls, no-go areas). The combination of all the three fields allows individual targets in high density crowds to be tracked.

Although Helbing carried out simulations with social forces models (Helbing

et al., 1997), they were directed at the visualization of the output of his model rather than the true pedestrian simulation. Pelechano & Badler (2006) uses a low level local motion system based on social forces to animate crowds in building evacuation experiments. The work by Gayle *et al.* (2009) presents a new approach to compute collision-free paths for multiple robots (or virtual agents) subject to local coordination constraints (specifically the initial and final configurations and possibly some additional coordination constraints). To solve this problem, their approach generalizes the social potential field method to be applicable to both convex and non-convex polyhedra. Social potential fields are integrated into a physics-based motion planning framework which uses constrained dynamics to solve the motion planning problem.

Pedestrian simulators like STEPS (Yan, 2010) or VISSIM (Almeida *et al.*, 2013) are based on the social forces model.

2.3.4 Psychological force based simulations

A set of different approaches inspired by Helbing's social forces model have been developed following the original idea: to use forces to represent impulses that are out of the physical level. The work by Braun *et al.* (2003) extended the social forces model to deal with different individualities for agent and group behaviors.

In their article, Pelechano *et al.* (2007) present a microscopic crowd simulation system named HiDAC, where social forces are combined with psychological and geometrical rules in a layered behavioral architecture of the agent. The system is capable of emergent behaviors (agent line formation and pushing behavior). Pedestrian behavior is sensitive to the current situation, the personalities of the individuals and the perceived social density. An extension of HiDAC named OCEAN (Pelechano *et al.*, 2008) provides each agent with a personality model in order to examine how the collective behavior of the crowd is affected. Each personality is attached to an automation of the low level process, therefore, different personalities have different effects in the crowd.

2.3.5 Agent-based simulations

Agent-based simulation has become increasingly popular as a modeling approach in the social sciences because it enables one to build models where individual entities and their interactions are directly represented. It allows modelers to represent multiple scales of analysis, the emergence of structures and various kinds of adaptation and learning (Gilbert, 2007). They are flexible in several aspects such as adding more individuals to the systems, adjusting the complexity level of individual behaviors or defining different degrees of aggregation. These characteristics make agent-based systems appropiate for pedestrian and crowd simulation (Johansson & Kretz, 2012). In the animation and virtual environments contexts, ABM for pedestrian simulation is an active research field which considers simulations that can vary from small groups to crowds.

One type of agent-based simulation is that of using real data to calibrate and specify the movements of the agents. A work by Pettre's group calibrates a pedestrian following model to be used in queue simulation from real data (Lemercier *et al.*, 2012). Assuming a numerical model (the Aw-Rascle traffic model) they use real data to calibrate the parameters of the model to represent pedestrians in a queue. The model fits the real data and the fundamental diagram of the problem. In the paper by Paris *et al.* (2007), a reactive method for solving interactions between pedestrians in a crowd is proposed. The approach is predictive. For each entity, at desired rates, they search for a solution-move satisfying the constraints and guaranteed to remain valid for the desired time window. The reachable space is represented as a cone in the (x, y, t) space (including the temporal dimension). The calibration of the system is carried out by analyzing video captures of real crowd motions. A commercial pedestrian simulator which also uses this real-data calibration-based approach is Legion (http://www.Legion.com). Legion is an agent-based pedestrian simulation software based on a large collection of real pedestrian measurements. Their algorithms have been calibrated and validated based on this real data, and they are based on the principle of least effort (Still, 2000). The idea is to optimize the path of each pedestrian over several restrictions: first, by the individuals' speed distribution and secondly, by the requirement of visiting certain places or sub regions in some order as part of the

route plan of an individual. The problem is to minimize a cost function based on the length, total time or total effort satisfying the constraints. The optimization process uses a type of simulated annealing over a set of allowed paths, randomly varying them and selecting the cheaper candidates in a iterative scheme.

Another kind of agent-based simulation is made up of geometrically-based local avoidance models. These models carefully check the absence of future collisions locally, and can be combined with other high-level approaches such as layered behavioral architectures to enable high-level goals in complex environments to be achieved. In the works by Olivier *et al.* (2012); Pettré *et al.* (2009), a metric named 'minimal predicted distance' to detect the critical distance in order to avoid collisions between walkers is defined. Pettré uses real pedestrian data to propose a model of collision avoidance with a few parameters that can be calibrated for different situations using real data. The model is able to correctly determine if, when and how motion is adapted to solve interactions. A different approach to the collision problem is described in Feurtey's thesis (Feurtey, 2000). This is based on a representation of trajectories in a three-dimensional space with time as the third dimension (x, y, t). The trajectories of the pedestrians are represented in this space, and each agent selects his future movements to avoid collisions, minimizing detouring and speed variation. The approach by Karamouzas *et al.* (2009) is based on the hypothesis that an individual adapts his/her route as early as possible, trying to minimize the amount of interactions with others and the energy required to solve these interactions. Each pedestrian computes if other pedestrians are on collision course within a certain anticipation time. After the calculation, each pedestrian makes an efficient move to avoid the collisions, anticipating future collision situations. The Gamma group from the University of North Carolina has also considered this problem from another perspective. They propose ClearPath, a multi-agent collision avoidance method based on convex optimization and use a discrete optimization method to efficiently compute the motion of each agent (Guy *et al.*, 2009). The resulting algorithm can be paralleled by exploiting data-parallelism and thread-level parallelism. As a result, ClearPath can handle from tens to hundreds of thousands of heterogeneous agents in milli-seconds.

Another large group of work on agent-based simulation is focused on crowd

simulation. This group will be revised in the forthcoming subsection dedicated to crowds.

2.3.6 Navigation Fields and other global macroscopic navigation approaches

The work by (Patil *et al.*, 2011) applies navigation fields to pedestrian simulation. They define a navigation field as a function $N : \mathbb{R}^2 \longrightarrow \mathbb{S}^1$, $\mathbb{S}^1 = [0, 2\pi]$ that exhibits the following properties: i) it must be free from local minima, except for the presence of sinks at specified goals (point of arriving), ii) agents should trace out paths of least effort (minimum cost) to their goals and, iii) these fields should be almost smooth and should be planned around static obstacles in the environment. In this work, they define a second layer, called a guidance field, in which the user can arbitrarily edit his/her preference about navigation. The guidance field is transformed into a navigation field that is guaranteed to be free of local minima and which is blended with the existing vector field to be transformed into a new one.

An alternative data structure to the navigation field is the navigation graph (Yersin *et al.*, 2005). It represents a set of navigable areas as the nodes of a graph. Navigation is possible only between connected areas. Velocity fields are computed based on the environment description given by the navigation graph. The work by Pettré *et al.* (2007) also uses navigation graphs for realtime navigation of a very large number of entities. In this case the graph is the main structure for a route planning system which uses it to answer navigation queries with a set of solution paths from which the planned navigation is executed in an efficient manner.

Continuum flow theory has been applied to crowd simulation mainly. For instance, Treuille *et al.* (2006) introduce the idea of combining two different fields to solve the problem of moving large crowds without explicit collision avoidance. One is a maximum speed field that determines the velocity of an individual at position $\vec{x}$. With this field, preferences for certain paths can be expressed. The other is a discomfort field so that, all things being equal, people would prefer to be at point $\vec{x}$ rather than $\vec{x'}$. This field can be changed dynamically to imple-

ment collision avoidance between people and other moving obstacles. Another approach to the continuum theory is 'flow tiles' (Chenney, 2004), based on the design of velocity fields defined in small, confined areas named tiles. Tile fields can be constructed to meet a wide variety of external and internal boundary conditions. Each flow tile contains a small field, and many tiles can be combined to produce large flows. The corners and edges of the tiles are constructed to ensure continuity across boundaries between tiles. The resulting tiling is divergence-free and hence suitable for representing a range of effects. This technique has not only been applied to pedestrian navigation but also to simulate river flow and swirling fog. In the work by Narain *et al.* (2009) a dual representation (discrete and continuous) for the crowd is proposed. In the continuous representation, the crowd is represented as a fluid described by a density and a flow velocity. To simulate the local interactions, this work defines the unilateral incompressibility constraint, a macroscopic property of the crowd to the local inter-agent collision avoidance. This property of the crowd produces a non negative pressure p, which prevents the flow from converging to a density higher that a value ρ_{max} such that the following constraint is satisfied at any point in the crowd $\rho < \rho_{max} \implies p = 0$.

Another group that has appeared recently presents data-driven methods for constructing group behavior models based on real crowd video footage. The approach presented by Lee *et al.* (2007) is a vision-based method of simulating a crowd of virtual humans. They record the motion of a human crowd from an aerial view using a camcorder, extract the two-dimensional moving trajectories of each individual and then teach an agent model from observed trajectories using a regression-based method. Another work that also uses computer vision techniques is that reported by Musse *et al.* (2007). In this work, the trajectories of the pedestrians are captured automatically from filmed video sequences and grouped into similar classes using an unsupervised clustering algorithm. An extrapolated velocity field is generated for each class. A simulator is used to animate virtual humans, aiming to reproduce the trajectories fed to the algorithm, avoiding collisions with other agents. One specific technique that uses real data is the data-driven animation models. This approach emerged from the context of motion capture techniques to animate virtual actors and avatars. The work by Courty & Corpetti (2007) proposed deriving a similar approach for crowd

motions, that is, *crowd motion capture*. In their framework, the motion of the crowds is represented as a time series of velocity fields estimated from a video of a real crowd. This time series is used as an input of a simple animation model that advect people along this time-varying flow.

2.3.7 Layered-behavior-based simulation

The decision-making process in microscopic simulators can follow a hierarchical scheme: strategical, tactical and operational (Daamen, 2004). The destinations and path planning are chosen at the strategical level, the route choice is performed at the tactical level and the instantaneous decisions to modify the kinematic state are taken at the operational level. Most agent-based models separate the local interactions from the necessary global path planning defining the local and global navigation problems in a layered model (Sung *et al.*, 2004). Making this split inside the agent model has the advantage that cognitive or behavioral capabilities can be introduced to the agents.

One of the first layered model for pedestrians is the work by Reynolds (1999). It presents autonomous agents to be used in games or animations guided by steering behaviors which are independent of the means of locomotion. The pedestrian global behavior is composed of three levels: locomotion, steering and strategic level. The steering level can be used to achieve higher level goals. In this level, the steering behaviors can be combined to create more complex behaviors, while the physical development of the pedestrian's movement is left to the locomotion level. In the work by Sakuma *et al.* (2005), a two-layer architecture for the pedestrian is presented. The low layer is a navigation system with a reactive behavior to implement collision avoidance. This navigation is controlled by environmental information through a vector field. The high level uses a psychological perception model and virtual memory to perceive the environment and implementing a sort of mental state of stress to control the collision avoidance system. The work by Shao & Terzopoulos (2005) on autonomous pedestrians integrates motor, perceptual, behavioral and cognitive components. Their human characters are defined as a hierarchical artificial life model, where different layers that progress in levels of abstraction provides different models of behavior, from reactive to cognitive lev-

els. This model was tested in a virtual 3D environment representing the old Penn Station at New York populated with over 600 autonomous pedestrians with five different types of pedestrians: commuters, tourists, performers, policemen and soldiers.

Another approach consists of the combination of different models to address different behavioral levels. The HiDAC architecture by Pelechano *et al.* (2008), introduced in Section 2.3.4, combines a social and physical forces model with psychological and geometrical rules to build the behaviors of individuals inside a crowd. This combination of a social-forces model with a rule-based model enables a variety of behaviors like queuing, pushing and falling. The behaviors are directly managed by the forces model, but HiDAC is also capable of simulating panic situations, or impatience, because the psychological rules can change the configuration of the agent's velocity (in the case of panic) or force the planning module to find alternative routes (in the case of an impatient behavior).

2.3.8 Crowds simulation

Simulating crowds is a challenging task. In order to execute the simulation efficiently and effectively, a crowd simulation system should meet certain requirements from the computational point of view. These requirements have been identified by Zhou *et al.* (2010) as:

1. Flexibility: This refers to the ability to adapt to different situations. A model is flexible if it is able to embrace changes, especially when these changes are unexpected. Many crowd models are specific for a situation.

2. Extensibility: A crowd model should accommodate new features (e.g. new behavioral factors in the decision making module) without much difficulty.

3. Execution Efficiency: This concerns the time needed to execute a crowd simulation for a given scenario. One especially challenging problem is real-time crowd simulation.

4. Scalability: This is the ability of increasing the size of the crowd without loosing significant performance.

Crowd simulation brings together works that focus on many different problems. Here, I will present a small selection of the works that have been carried out so far which I consider relevant because they propose different lines of research inside this domain.

One approach to the human crowd simulation problem is derived from the ethology field. The study of social animals (especially those easy to observe and control such as insects) suggests different strategies for collective displacements such as pheromone trails in the case of ants or leader following in the case of flocks. The paper by Banerjee *et al.* (2005) studies the behavior of ant colonies in order to derive strategies for crowd stampedes induced by panic. They simulate a scenario of connected cities in a war affected country. A spatial network based on frequently used paths is implemented. The individuals generate a pheromone network that is in fact a potential field that guides the movements of the crowd. The more transited a path, the more intense is the generated field. By analogy with ant pheromones, the dynamic field diffuses and evaporates.

Several works have focused on the realtime simulation problem. The work by Musse & Thalmann (2001) uses a hierarchy composed of virtual crowds, groups and individuals. They use different animating techniques depending on the group size, from scripted behaviors, through behavioral rules to external control. Van den Berg *et al.* (2008) proposes a formulation that uses a precomputed road map for wayfinding and combines it with a fast localized navigation for each agent. The agents sense the environment independently and compute a collision-free path. Their algorithm performance scales almost linearly with the number of agents, running at interactive rates on multi-core processors. The work by Pettré *et al.* (2006) uses a preprocessing technique giving rise to navigation graphs which are then used to navigation and simulation tasks. This graph supports path planning which distinguishes the navigable areas from impassable obstacles. The simulation model uses local instantaneous population densities to move fast groups of individuals. The system also uses realtime rendering techniques to be more efficient. Lister & Day (2012) address the problem of crowd simulation in realtime without proposing a specific technique but instead a bone-parallel, OpenCL-accelerated interpretation of the traditional character pipeline. The method does not rely on pre-processing; provides a fine grained control over

the animation of a crowd and crowd members and user-controlled avatars can be handled without distinction.

The scalability problem has different points of view. With respect to the classic scalability problem, that is, the ability to increment the number of individuals, Qiu & Hu (2013) proposes a crowd model using a spatial activity-based approach. Contrary to conventional agent-based models where pedestrians make movement decisions in a time-based manner, this model allows pedestrians to make decisions only when needed, that is, when there is a significant change in the pedestrians' spatial position. This can lead to more efficient simulations and better scalability. The problem with behavioral scalability, that is, the problem of adaptively increasing the complexity of the individual behaviors in a crowd, is tackled in the work by Sung *et al.* (2004). The authors propose a situation-based control structure for each agent. Basic agents have limited behaviors. As they enter new situations, additional, situation-specific behaviors are composed on the fly to enable agents to respond appropriately. The authors define several behavior functions that transform a behavior into a set of transition probabilities on the states. The probabilities of the different functions are calculated in each situation based on the available information, the history of the agent and his/her position. The work by Lozano *et al.* (2009) analyzes different computer architectures to support virtual environments and proposes a scalable computer architecture to support thousands of autonomous agents. This architecture consists of a cluster of computers with a hierarchical client-server software architecture that efficiently provides consistency.

Another approach to achieve low computational cost behaviors for scalability, as well as to create extensible systems in terms of the managed behaviors, is to include behavioral information in the environment. In the work by Gutierrez *et al.* (2005) each object in the virtual environment is not only a 3D shape but it is also a dynamic entity with multiple visual representations and functionalities. The popular computer game 'The Sims' uses a similar approach, because the objects have information about the correct human behavior that manipulates them. In the work by de Pavia *et al.* (2005), the virtual humans that populate a virtual urban environment are directed following a semantic included in the virtual space. Thus, children go to school and adults go to work at the usual time. The ontology

is included in a basic crowd simulator with collision avoidance. The result is an urban environment with realistic and coherent population behaviors.

Another problem is the creation of natural individual movements inside the crowd to make it more credible. The work by Karamouzas & Overmars (2012) centers the problem on the simulation of small groups of pedestrians. Inspired by the studies of Moussaïd *et al.* (2011) that states that a crowd is actually made up of small groups which have their own internal dynamics, Karamouzas defines groups of three or four pedestrians which follow three different formations (line-abreast, V-like and river-like). The members of the group have several restrictions for admissible desired velocities and collision avoidance. For each individual, a velocity is determined using the optimization of an utility function that depends on the deviation from the group's desired velocity, and the minimal predicted time to collision. Gu & Deng (2011) focuses on the diversification of styles of motion in the neighborhood of each pedestrian. Their approach dynamically controls agents' motion styles to increase a crowd's motion variety. The idea is to maximize local varieties of walking styles while maintaining a consistent global style. An off-line pre-processing algorithm extracts primitive motions from a motion capture system and stylizes them. The approach shows superior flexibility when compared to traditional random distribution of motion styles.

Two important books on crowd simulation are Thalmann & Musse (2007) and Pelechano *et al.* (2008).

2.4 Commercial and research-oriented pedestrian simulators

There are a great variety of commercial and research tools for pedestrian and crowd simulation. In this subsection I will enumerate a set of them that I consider to be more relevant, although the list is far from being complete.

1. Legion. This is a commercial piece of software by Crowd Dynamics which implements a multi-agent pedestrian model. It has been applied to civil planning problems such as evacuation scenarios to improve safety. This software uses algorithms calibrated based on a large amount of video footage

observations in 40 different scenarios. Initially it was developed by G.K. Still using algorithms based on the least-effort principle.

2. Massive. This was originally developed by Stephen Regelous for use in the Lord of Rings trilogy to reproduce the battle scenes with thousands of warriors. It is an agent-based system where each agent has an AI layer to drive the motion. This is the leading software for crowd related visual effects.

3. Ped-Sim. This is a microscopic pedestrian crowd simulation system based on the social forces model. The agents have two layers: the physical layer that takes care of the physical aspects such as the movement, and the mental layer which is in charge of the behavior simulation. The mental layer is configurable with different strategies. It can be used as a stand alone system which provides its own data (numeric and graphics) or it can be used embedded in another simulation system as a pedestrian dynamics engine.

4. PEDGo. This is simulation software developed by TraffGo HT of Germany as the result of a research project by Tim Meyer-Köning and Hubert Klüpfel. It is able to simulate 10000 people on a domestic computer. It uses a multi-agent model with vector fields in a discrete space to simulate the microscopic movements of the pedestrians. PedGo meets IMO MSC.1/Circ.1238 guidelines for evacuation analysis for new and existing passenger ships certified by the See-Berufsgenossenchaft Hamburg.

5. PEDROUTE. This is software developed byhe tHalcrow Group Ltd. It is a macroscopic simulation system which treats the problem using the continuum flow approach. A central planning unit provides the Origin-Destination flow matrix and the travel time functions based on the Bureau of Public Roads model.

6. SimWalk. This is software developed by Savannah Simulations of Switzerland. It is used for evacuation simulations of crowds. It is based on the social forces model. It can model relatively large crowds and is used by several engineering institutes.

7. STEPs. This is developed by Mott MacDonald. It is an agent-based tool for the simulation of pedestrian movement under both normal and emergency conditions. The software has been validated against NFPA (a standardization agency for rail systems). It is able to simulate tens of thousands of pedestrians on a powerful computer, giving details of velocities and movements of individuals.

8. NOMAD. This is a microscopic simulation system developed by Serge Hoogendoorn of the Delft University of Technology to assess pedestrian walking infrastructure. Its agents work at the tactical level and the operational level developing route choice capabilities and walking behavior. It is activity-based, implying that the actions of the pedestrians are determined by the different activities pedestrians have planned to perform.

9. Golaem Crowd. This is a tool that integrates into Autodesk Maya. This tool provides the user with a library of pre-computed pedestrian behaviors and a set of editors that allow you to edit behaviors to adapt them to specific requirements and combine this behaviors to create thousands of different characters.

10. MassMotion. This is a commercial pedestrian simulation and crowd analysis tool developed by Oasis Software for transport planning. It is an agent-based simulation system. The agents are able to recognize different situations in the environment such as congestion or panic and plan alternative routes. Each agent has his/her personal agenda and develops a schedule based activity.

11. RVO2 Library. This is an implementation of the optimal reciprocal collision avoidance (ORCA) formulation of the Gamma Group of the University of North Carolina. It is a realtime algorithm for interactive navigation and planning of large numbers of agents that supports the collision avoidance capability. It has been included in the video game "Warhammer 40,000: Space Marine".

A summary of the different tools is displayed in Table 2.2.

There are not many independent articles that consider the analysis of existing commercial tools for pedestrian simulation. The paper by Zhou *et al.* (2010) offers a taxonomy of pedestrian software and pedestrian models using two parameters: crowd size and time scale. A total of four categories are defined combining large and medium-small sizes with short term and long term. Although the size of the crowd is an obvious parameter, this is not the case with the time parameter. For the authors, a crowd simulation model of a long term phenomenon usually focuses on some intangible social and psychological characteristic rather than the physical characteristics of the crowd. On the other hand, simulation models of short-term phenomena usually describe a crowds' physical characteristics, especially positions and movement patterns. The authors evaluate and clasify several pedestrian models and software tools using these criteria. In the paper by Sarmady *et al.* (2007), the authors study the most suitable software package that meets their requirement for simulating the crowd at the Masjid Al-Haram mosque in Mecca. The authors categorized the tools into two groups, evacuation and normal situation oriented tools. They analyzed seven software packages (Simulex, PedGo, GridFlow, AERI, Legion, STEPs and Simwalk) using four criteria: capability of simulating large crowds, geometry design tools, model of behavior and existence of reporting and evaluation tools.

Name	**Author**	**Model**	**Use**	**Scale**	**Calib./Valid.**
Legion	Crowd Dynamics	Least Effort principle	Civil planning	Large	Video footage
Massive	Stephen Regelous	ABM with flow fields	Movies	Large	No
Ped-Sim	Christian Gloor	Social forces model with mental layer	API for third party applications	Large	No
PedGo	TraffGo	Vector fields in a discrete space	Evacuation simulations	Large	Meet IMO/MSC.1 guidelines
PedRoute	Niels Hoffmann	Continuum flow model	Passengers flow analysis in airports and train terminals	Large	?
SimWalk	Savannah Sim.	Social forces model	Public transport facilities analysis	Large	Real world data
STEPs	Mott MacDonald	ABM with CA principles	Pedestrian facilities design	Large	NFPA validation
NOMAD	Serge Hoogendoorn	Layered model	Identification of problems in infrastructures	Large	Real data of controlled experiments
Golaem Crowd	Autodesk Maya	Library of precomputed behaviors	Population generator in graphic systems	Large	No
MassMotion	Oasis Software	ABM with social forces model	Transport planning	Large	Contrasted with data of Fruin's works.
RVO2	Gamma Group	ORCA	API for third party applications	Medium	?

Table 2.2: Summary of the pedestrian simulation systems. In the column 'Scale' the classification of 'Large' means thousands of individuals. The classification of 'Medium' means hundreds of agents. The question marks mean that no information have been found.

Another type of problem considered around pedestrian software analysis is determining the type of simulation techniques used by proprietary software, where technical information is scarce. A clear understanding of how the model works is necessary to help the user establish appropriate input parameters and boundary conditions. The work by Rogsch & Klingsch (2012) proposes a set of test scenarios and assessment conditions that determine the type of model that a commercial tool uses. Through the use of these simple scenarios, the user of software for pedestrian simulation can determine certain important characteristics of the system such as: i) how acceleration, deceleration and overtaking are implemented, ii) the use (or not) of potential fields to direct the trajectories, iii) the correctness of the implemented boundary conditions especially near the exits inside closed rooms, iv) route choice capabilities, v) adequacy to the fundamental diagram of pedestrian dynamics and, vi) the order of updating the pedestrians positions in each time step (which is important for evacuation modeling).

2.5 Chapter highlights

1. Two basic approaches exist in pedestrian modeling and simulation. From the macroscopic perspective, pedestrian movements are described using magnitudes like flow, average speed and area module. It is interested in crowd movement and shape and it is difficult to model emergent behaviors. However, it can be computationally efficient and easily scalable for the number of individuals. The microscopic approach involves individual units with local characteristics such as speed, position and interactions. When interactions are considered individually, the emergence of collective behaviors is possible.

2. There are no clear limits among the different approaches that belong to a group (macroscopic or microscopic). For example Agent-based models can include individuals who generate their dynamics using the social force model. Analogously, the social force model can be a layer of more complex behavioral models that include psychological levels or planning.

3. The fundamental diagram displays at a point the relationship between the

average speed of pedestrians and the density. It is a basic tool to measure and compare these macroscopic key indicators of pedestrians streams quantitatively.

4. Many commercial pedestrian simulators exist. Most of them follow a microscopic model that permits the design of complex behaviors. The field of applicability of existing pedestrian simulators are mainly architectural and civil engineering studies, and crowd simulations for movies and video games.

Chapter 3

Theoretical Framework

Reinforcement Learning (RL) is a well-founded field in the machine learning area devoted to solving sequential decision-making problems. The goal is to learn to control a system so as to maximize a numerical value which represents a long-term objective. Specifically, the idea consists of using experiences interacting with the environment to learn the optimal *value function*. This value function predicts the best long-term outcome an agent could receive from a given state when it applies an action and follows the optimal policy thereafter (Santamaria *et al.*, 1997). In a RL problem, the rewards define the task to learn. For example, in a RL game problem, an agent could learn to play a game by being told whether it wins or loses through a reward, but he/she is never given the correct action at any point in time. In a RL setting, there are two main entities: the controller and the environment. The controller learns by interacting with the environment following these dynamics: i) From the environment the controller receives the current state and a reward associated with the last state transition. ii) The controller calculates an action which is sent-back to the environment. iii) In response, the environment makes a transition to a new state and the cycle begins again. In general, the state transition is an stochastic function. The problem consists of learning a controller that maximizes the total reward, starting from any state. In RL theory, the controller is exclusively concerned with the learning process, leaving the rest of the processes (i.e.sensing) to the environment. In this work, the controller resides in the agent, setting a relationship between agent and environment instead of controller and environment. An schema of the process

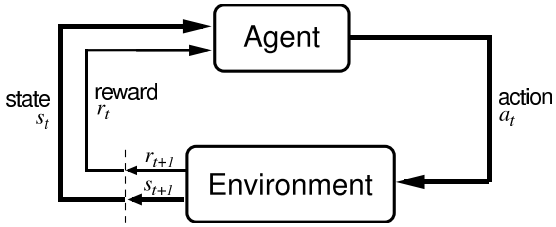


Figure 3.1: Scheme of a single-agent reinforcement learning process (Image from Sutton's book (Reinforcement Learning MIT Press)

can be seen at Figure 3.1. The following works Busoniu *et al.* (2010); Kaelbling *et al.* (1996); Sutton & Barto (1998); Szepesvári (2010); Wiering & van Otterlo (2012) focus on the RL foundations.

3.1 Single-agent reinforcement learning

A reinforcement learning problem can be modeled as a Markov Decision Process (MDP). In the 50's Bellman proposed a numerical method to solve sequential decision processes using a recursive functional equation (Bellman, 1957). The MDP concept was introduced by Howard (1960) in 1960 as a Markov chain in which each transition represents a criticized decision. The following definitions introduce the basic concepts:

Definition 1. *A Markov Decision Process is a 4-tuple constituted by a state space, S, an action space, A, a probabilistic transition function $P : S \times A \times S \rightarrow [0, 1]$ and a reward function $\rho : S \times A \times S \rightarrow \mathbb{R}$.*

The state signal s_t describes the environment at discrete time t. In a state s_t, the decision process can select an action from the action space $a_t \in A$. The execution of the action in the environment changes the state to $s_{t+1} \in S$ following the probabilistic transition function $P(s_t, a_t, s_{t+1}) = Pr\{s_{t+1} = s' \mid s_t = s, a_t = a\}$ that is, the conditional probability of going to state s' at time $t+1$ when being at time t in state s and performing action a. Each decision is accompanied by an immediate scalar reward given by the reward function $r_{t+1} = \rho(s_t, a_t, s_{t+1})$ that represents the value of the decision taken in state s_t. *The goal of the process is to find a* policy*, that is, a mapping between states and actions that maximizes a*

function of the reinforcement.

Definition 2. *A policy is a probability function* $\pi : S \times A \rightarrow [0, 1]$ *where* $\pi(s, a)$ *gives the probability of selecting action* a *being in state* s*. A policy is called stationary if it does not change over time.*

Definition 3. *A deterministic policy, also known as greedy policy is one that for each state of* S *has probability 1 of taking an action of* A *and probability 0 of taking the rest of actions. Formally:* $\forall s_i \in S \ \exists a_j \in A \mid \pi(s_i, a_j) = 1, \ \pi(s_i, a_k) = 0, \ k \neq j$

The behavior of the agent is determined by its policy which specifies how the agent chooses the action given the state. The agent's goal is to maximize at each time-step t the *expected discounted return* defined as:

$$R_t = E\{\sum_{j=0}^{\infty} \gamma^j r_{t+j+1}\} \tag{3.1}$$

where the $\gamma \in (0, 1]$ parameter is the *discount factor* and the expectation is taken over the probabilistic state transitions[1]. Figure R_t represents the rewards accumulated by the agent in the long run (Busoniu *et al.*, 2010). The discount factor can be interpreted as the increasing uncertainty of future rewards because the more distant the future reward is, the less importance it has in the return of instant t (note that factor γ^j tends to 0 in case of $\gamma < 1$). This discount factor has an important effect on the navigational problems (the class of problems that we are interested in). Figure 3.2 shows a transition graph commonly used to describe situations in RL problems. The colored circles represent actions and the large circles represent states. The values are the immediate rewards received to execute a specific action. To make it easier, imagine a deterministic model where the transitions do not follow a probabilistic function but they are deterministic transitions. The agent's learning process is at discrete time k in state S_k. If the agent chooses the action A_k, the agent goes to state S_{k+1} and after selecting action A_{k+1} arrives at the goal receiving an immediate reward of 1. The return

[1]In case where $\gamma = 1$ the return is named undiscounted and it is common in episodic problems

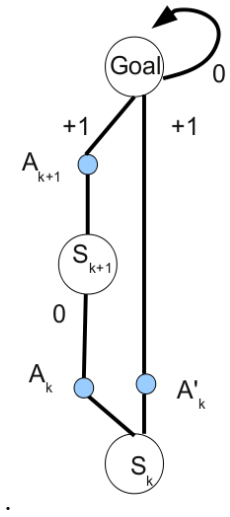


Figure 3.2: Transition graph to show the effect of the discount factor

that obtains state S_k is, according to the Equation 3.1, $R_k = 0 + \gamma < 1$. On the contrary, when action A'_k is selected, the return is $R'_k = 1$. Therefore state S_k will prefer to select A'_k action because the return is greater. The discount factor introduces a preference for the shortest path in terms of the number of actions selected which has a direct translation to the shortest distance path selection when the RL problem is a navigational problem.

In an MDP, the necessary information to take correct decision is in the state signal itself. This means that the history of the agent, that is, the sequence of states and actions that occurred before are not taken in consideration. It is said that MDPs have the *Markov property.*

Definition 4. *Markov property.*

$$Pr(s_{t+1} = s', r_{t+1} = r \mid s_t, a_t, r_t, s_{t-1} \ldots r_1, s_0, a_0) = \\ Pr(s_{t+1} = s', r_{t+1} = r \mid s_t, a_t, r_t) \tag{3.2}$$

This equality means that the environment's response at time $t+1$ only depends on the state and action chosen at the immediate previous time t.

If an environment has the Markov property, then its one-step dynamics enables us to predict the next state and expected next reward given the current state and

action. An example of a system with the Markov property is chess. In each moment, the player has all the necessary information to decide, by observing the current configuration of play without knowing how each piece came to its current position.

3.1.1 Action-value functions

Almost all RL algorithms *in control problems* are based on estimating action-value functions, functions of state and action pairs, that measure how good it is for the agent to perform a specific action in a given state. The notion of "how good" is defined in terms of the expected return that the agent can receive in this state (Sutton & Barto, 1998).

Definition 5. *The Action-value function (Q-function)* $Q^\pi : S \times A \rightarrow \mathbb{R}$ *is the expected return of a state-action pair given the policy* π*:*

$$Q^\pi(s,a) = E\{R_t \mid s_t = s, a_t = a, \pi\} = E\{\sum_{j=0}^{\infty} \gamma^j r_{t+j+1} \mid s_t = s, a_t = a, \pi\} \quad (3.3)$$

The action-value functions define a partial ordering relationship over policies (Sutton & Barto, 1998). A policy π is better than or equal to the policy π' ($\pi \geq \pi'$) if $Q^\pi(s,a) \geq Q^{\pi'}(s,a)\ \forall s \in S, a \in A$. There is at least one policy that is better than or equal to all other policies. This policy is an *optimal policy*, and we denote it as π^*. All optimal policies share the same optimal action-value function that is defined as $Q^*(s,a) = \max_\pi Q^\pi(s,a)\ \forall s \in S, a \in A$. An optimal policy is automatically derived from the optimal value-function:

$$\pi^*(s) = \operatorname*{argmax}_a Q^*(s,a) \quad (3.4)$$

.

One fundamental property of action-value functions is that they satisfy a recursive relationship:

$$
\begin{aligned}
&Q^{\pi}(s,a) = E_{\pi}\{\sum_{j=0}^{\infty}\gamma^{j} r_{t+j+1} \mid s_t = s, a_t = a\} = \\
&E_{\pi}\{r_{t+1} + \gamma\sum_{j=0}^{\infty}\gamma^{j} r_{t+j+2} \mid s_t = s, a_t = a\} = \\
&\sum_{s'} P(s,a,s')[\rho(s,a,s') + \gamma\sum_{a'}\pi(s',a')E_{\pi}\{\sum_{j=0}^{\infty}\gamma^{k} r_{t+j+2} \mid s_{t+1} = s', a_{t+1} = a'\}] = \\
&\sum_{s'} P(s,a,s')[\rho(s,a,s') + \gamma\sum_{a'}\pi(s',a')Q^{\pi}(s',a')]
\end{aligned}
\tag{3.5}
$$

This equation is the *Bellman equation* for Q^{π}, and expresses the relationship between the value of function Q^{π} for an state-action pair and the values of its successors pairs.

Definition 6. *The Bellman principle of optimality. From any point (an state-action pair) on an optimal policy, the remaining policy is optimal for the corresponding problem initiated at that point.*

From this definition, to apply the Bellman principle to the Bellman equation (Equation 3.5) means to select in each state s the action with maximum Q value. With this operation the equation for Q^{*} also known as *Q-function* is defined as:

$$
\begin{aligned}
Q^{*}(s,a) = \sum_{s'} P(s,a,s')[\rho(s,a,s') + \gamma\max_{a'} Q^{*}(s',a')] \\
\forall s \in S, a \in A
\end{aligned}
\tag{3.6}
$$

Equation 3.6 is a contraction mapping. It has been proved that the optimal solution of this dynamic programming equation is unique and can be calculated iteratively by successive approximation (Hernandez-Lerma, 1989). The agent can achieve the learning goal by first computing Q^{*} and then choosing actions by the greedy policy, which is optimal (i.e. maximizes the expected return) when applied to Q^{*} as expressed in Equation 3.4.

3.1.2 Solving Q^*

The calculus of the maximum action-value function Q^* (or Q-function) depends on the information available concerning the MDP. Specifically, if the transition function $P(s, a, s')$ and the reward function $\rho(s, a, s')$ are known, dynamic programming (DP) methods can be applied. If the transition function P is unknown, which means that there is not a model of the dynamics of the environment, RL methods can be applied[1]. In the field of pedestrian dynamics, many partial models have been developed to explain or simulate specific problems, like lane formations, oscillations, etc. These models generally take the form of abstract rules or complex dynamic equations that are very difficult to convert to an MDP problem. This is the main reason why my work will focus on using model-free, RL-family algorithms.

Basically, there are three classes of DP/RL algorithms: policy iteration algorithms, value iteration algorithms and policy search. They are characterized as follows (Busoniu *et al.*, 2010):

- Value iteration algorithms search for the optimal Q-function which consists of the maximal returns from every state-action pair. The optimal Q-function is used to compute an optimal policy.
- Policy iteration algorithms evaluate the policies by constructing their action value function (instead of the optimal value function) and use these action value functions to find new, improved policies. These two processes can be performed simultaneously with each new piece of data.
- Policy search algorithms use optimization techniques to directly search for an optimal policy.

In this chapter I will focus on the types of model-free algorithms that are going to be used in the experiments. Specifically, I will introduce a value iteration algorithm and a policy iteration algorithm.

[1] Another possibility exists which consists in interacting with the environment to build a model and then use DP methods to solve the MDP. This approach has been followed in algorithms like Dyna-Q (Sutton, 1990) and Prioritized Sweeping (Moore & Atkeson, 1993)

3.1.3 A model-free value iteration algorithm. Q-learning

Q-learning was first introduced by Watkins (1989). It has become the most widely used algorithm from the Value iteration class. It starts from an arbitrary initial Q-function Q_0 and updates it using pieces of data tuples collected by interacting with the environment. These tuples have the form $(s_t, a_t, s_{t+1}, r_{t+1})$. After each transition, the Q-function estimation is updated using the data tuple as follows:

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_{a'} Q_t(s_{t+1}, a') - Q_t(s_t, a_t)] \tag{3.7}$$

where $\alpha \in]0, 1]$ is the learning rate. The term in brackets is the *temporal difference* (TD), i.e. the difference between the updated estimate of the optimal Q-value of (s_t, a_t), $(r_{t+1} + \gamma \max_{a'} Q_t(s_{t+1}, a'))$ and the current estimate $Q(s_t, a_t)$. The Q-learning algorithm belongs to the family of the TD(0) control algorithms.

As the number of transitions t approaches infinity, Q-learning asymptotically converges to Q^* in a discrete state and action spaces under the following conditions (Watkins & Dayan, 1992):

- $\sum_{j=0}^{\infty} \alpha_j^2$ is finite and $\sum_{j=0}^{\infty} \alpha_j \to \infty$.
- All the state-action pairs are visited infinitely often.

The second condition is satisfied in theory if the controller has a non zero probability of selecting any action in every state (Busoniu *et al.*, 2010). But the controller also has to exploit the acquired knowledge to approximate to the optimal Q-function. This is the *exploration-exploitation trade-off* in RL. The agent has to explore the environment to find new better policies but also has to exploit the acquired knowledge to improve the expected return. The agent has to decide at each moment whether to act to gain new information (explore), or to act consistently with past experience to maximize reward (exploit). The solution to this dilemma is to use an exploratory strategy that balances the exploration with the exploitation inside the learning process. Actions are selected by using a probability distribution that changes over time, so that the policy explores more at the beginning of the learning process and exploits more at the end, becoming a greedy policy in the final stages. There are two commonly used exploratory strategies:

- The ϵ-greedy policy. This strategy selects with probability ϵ a random action and with probability $(1-\epsilon)$ the greedy action (the action that provides a higher expected return at the moment of the evaluation). In practice, the policy sets an initial value of ϵ that tends to 0 (greedy policy) at the end of the learning process.

- Softmax. This strategy follows a Boltzmann distribution where action a on the tth play is selected with probability

$$Pr_t(a) = \frac{e^{Q_t(a)/\tau}}{\sum_{b=1}^{n} e^{Q_t(b)/\tau}} \tag{3.8}$$

  where τ is a positive parameter called temperature. High temperatures cause an equiprobable choice of actions and at low temperatures the probabilities are different according to the differences in the values of Q_t. When the temperature is near 0, Softmax becomes the same as the greedy policy.

Aside from these approaches that need for *a priori* discretization of the state space, other proposals have arised without this restriction like the multi-resolution exploration (MRE) proposed by Nouri & Littman (2008). In MRE, the level of generalization of the state space is dynamically adjusted during the learning process using a hierarchical mapping to identify regions of the state space that would benefit from additional samples.

The trade-off between exploration and exploitation consists of choosing actions that are best according to the current state of knowledge, and actions that are not the current best but improve the state of knowledge and potentially yield higher payoffs in the future. It combines new experience with old value functions to produce new and statistically improved value functions in different ways

The Algorithm 1 shows the Q-learning algorithm. The final conditions are not specified because they depend on the implementation. An episode ends when a final state has arrived or when a fixed number of action executions have been performed. The final condition for the algorithm is specified in practice by setting a number of trials.

The update rule of the Q-learning algorithm uses a greedy policy ($\max_{a'} Q(s', a')$) while the algorithm uses another policy for interacting with the environment (an

Algorithm 1: Q-learning

Data: discount factor γ, exploration schedule, learning rate schedule, number of episodes
Result: Q^*

begin
  Initialize $Q(s,a) \quad \forall s \in S, a \in A$ arbitrarily;
  repeat
    Initialize s;
    repeat
      Choose a from s using a policy derived from Q (e.g. ϵ-greedy);
      Take action a, observe r, s';
      Update the Q-function:
      $Q(s,a) = Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$;
      $s \leftarrow s'$;
    until *end condition of the episode*;
  until *number of episodes completed*;

exploratory policy). For this reason it is said that Q-learning is an off-policy RL algorithm. Also note that the update rule is very similar to the Bellman optimization rule shown in Equation 3.6. This is a characteristic of the Value iteration algorithms. The convergence of the Q-learning algorithm as well as other off-policy variants were formally proved in Jaakkola *et al.* (1994); Watkins & Dayan (1992).

3.1.4 A model-free policy iteration algorithm. Sarsa(0)

The algorithm Sarsa(λ) is a model-free policy iteration algorithm proposed by Rummery & Niranjan (1994) as an alternative to Q-learning. Although initially its name was *modified Q-learning*, the name Sarsa was introduced by Sutton in 1996.

Sarsa starts with an arbitrary initialized Q-function and updates it at each step as follows:

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha \left[r_{t+1} + \gamma \, Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t) \right] \tag{3.9}$$

where α is the learning rate and γ the discount factor.

The term in brackets is the temporal difference. It is not the same as the temporal difference used in Q-learning. While the Q-learning temporal difference is similar to the Bellman optimization rule, the Sarsa includes the action a_{t+1} that is selected by the current policy that is being learned. This is why it is said that Sarsa is an on-policy temporal difference method. On-policy algorithms cannot separate exploration from learning and therefore must confront the exploration problem directly. The update rule of Sarsa (Equation 3.9) uses the 5-Tuple $(s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1})$ that makes up the transition from one state-action pair to the next. This 5-Tuple gives rise to the name Sarsa for the algorithm. Sarsa is continuously estimating Q^π using the policy that is being learned π. At the same time, policy π is changing because the exploratory policy uses the new updated Q^π. Sarsa converges with probability 1 to an optimal policy and an optimum action-value function as long as all state-action pairs are visited an infinite number of times and the policy converges in the limit to the greedy policy (Sutton & Barto, 1998). A version of Sarsa(0) is presented in the Algorithm 2. It has the same ending conditions as those for the Q-learning algorithm.

Algorithm 2: The Sarsa(0) Algorithm.

Data: discount factor γ, exploration schedule, learning rate schedule, number of episodes
Result: Q^*
begin
  Initialize $Q(s, a) \quad \forall s \in S, a \in A$ arbitrarily;
  repeat for each episode
    Initialize s;
    Choose a from s using a policy π derived from Q (e.g. ϵ-greedy);
    repeat for each step of the episode
      Take action a, observe r, s';
      Choose a' from s' using a policy π derived from Q (e.g. ϵ-greedy);
      Update the Q-function:
      $Q(s, a) = Q(s, a) + \alpha \left[r_{t+1} + \gamma \, Q(s', a') - Q(s, a) \right]$;
      $s \leftarrow s'$; $a \leftarrow a'$;
    until *end condition of the episode*;
  until *number of episodes*;

The convergence of the Sarsa(0) algorithm was formally proved in Singh *et al.* (2000).

3.1.5 Eligibility traces

As it was indicated above, the goal of the learning process is to find a policy that maximizes in each state-action pair the expected return defined in Equation 3.1. The value of this equation is calculated at the end of each episode, when all the immediate rewards have been collected. This is what the Monte Carlo methods do (Sutton & Barto, 1998). Given a state-action pair in step t, the value is updated with the collected rewards from this step till the end of the episode at step T as shown in Equation 3.10.

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots + \gamma^{T-(t+1)} r_T \tag{3.10}$$

This type of updating is slow because the immediate rewards have to be stored and the return R_t cannot be calculated until the episode ends. On the contrary, the TD(λ) methods use a part of the sequence of immediate rewards and substitutes the rest for a estimation using the correspondent value of the value function. This strategy is called *bootstrap*. Therefore, the R_t value could be estimated using different expressions depending on the step in which it is truncated. We refer the different expressions as a one-step target $R_t^{(1)}$, a two-step target $R_t^{(2)}$ and so on:

$$\begin{aligned} R_t^{(1)} &= r_{t+1} + \gamma Q(s_{t+2}, a_{t+2}) \\ R_t^{(2)} &= r_{t+1} + \gamma r_{t+2} + \gamma^2 Q(s_{t+3}, a_{t+3}) \\ &\cdots \\ R_t^{(k)} &= r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots + \gamma^k Q(s_{t+k+1}, a_{t+k+1}) \end{aligned} \tag{3.11}$$

In the TD(λ) family of control algorithms, the parameter λ gives weights to the set of n-step targets to get a weighted average (where the terms decay

exponentially in importance) following the expression called the λ-return:

$$R_t^\lambda = (1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1}R_t^{(n)} \tag{3.12}$$

The term $(1-\lambda)$ is a normalization factor of weights. When $\lambda = 0$, R_t^λ becomes $R_t^{(1)}$ that is the expression used in Algorithm 2.

This approach to the problem is known as the *forward* view. The forward view itself is not directly implementable because it is acausal, using at each step knowledge of what will happen many steps later. The *backward* view provides a causal, incremental mechanism equivalent to the forward view. In the backward view of the TD(λ) family of algorithms there is an additional memory variable which is associated with each state, the *eligibility trace* (Sutton & Barto, 1998). The elegibility trace of the state-action pair (s,a) at time t is denoted as $e_t(s,a) \in \mathbb{R}^+$. It is defined as:

$$e_t(s,a) = \begin{cases} \gamma\lambda e_{t-1}(s,a) + 1 & \text{if } s = s_t,\ a = a_t, \\ \gamma\lambda e_{t-1}(s,a) & \text{otherwise,} \end{cases} \tag{3.13}$$

At each step, the eligibility traces for all state-action pairs decay by $\gamma\lambda$ and the eligibility trace of the visited state-action pair (s_t, a_t) is incremented by 1. The γ parameter is the discount rate of the return (Equation 3.1). The TD-error for one decision at time t being at the state-action pair (s_t, a_t) is:

$$\delta_t = r_{t+1} + \gamma Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t) \tag{3.14}$$

This TD error is propagated towards all the state-action pairs according to their eligibility value which decays with time. Thus, the update rule of the TD family algorithms is:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha\delta_t e(s_t, a_t) \tag{3.15}$$

In Algorithm 3, the tabular version of the Sarsa(λ) algorithm is displayed.

In Sutton & Barto (1998), it is demonstrated that both, the forward view and the backward view, are equivalent. There are versions using eligibility traces for

Algorithm 3: The tabular Sarsa(λ) Algorithm.

Data: discount factor γ, λ, exploration schedule, learning rate schedule, number of episodes
Result: table Q^*

begin
  Initialize $Q(s,a) \;\; \forall s \in S, a \in A$ arbitrarily;
  Initialize $e(s,a) = 0 \;\; \forall s \in S, a \in A$;
  repeat for each episode
    Initialize s, a;
    repeat for each step of the episode
      Take action a, observe r, s';
      Choose a' from s' using a policy π derived from Q (e.g. ϵ-greedy);
      $\delta \leftarrow r + \gamma Q(s', a') - Q(s, a)$;
      $e(s,a) \leftarrow e(s,a) + 1$;
      forall the $s_i \in S$ *and* $a_j \in A$ **do**
        $Q(s_i, a_j) \leftarrow Q(s_i, a_j) + \alpha \delta e(s_i, a_j)$;
        $e(s_i, a_j) \leftarrow \gamma \lambda e(s_i, a_j)$;
      $s \leftarrow s'$;
      $a \leftarrow a'$;
    until *end condition of the episode*;
  until *number of episodes*;

the two algorithms presented in this chapter: Q(λ), the TD(λ) version of Watkin's Q-learning algorithm, and Sarsa(λ), the TD(λ) version of the Sarsa(0) algorithm. The algorithms using eligibility traces require more computation that one-step methods, but they offer significantly faster learning. Thus, it often makes sense to use these versions when data cannot be repeatedly processed as occurs in on-line applications such as in the framework presented in this thesis.

3.1.6 Complexity of the RL process

There has been a great deal of theoretical work on analyzing RL algorithms. The convergence proofs cited above for the Q-learning and Sarsa(0) algorithms with discrete states and actions spaces, guarantee that under certain conditions, these algorithms can compute optimal value functions in the limit. But these convergence results make no performance guarantee after only a finite amount of experience (Strehl *et al.*, 2006). The work by Even-Dar & Mansour (2003) showed that the tabular Q-learning algorithm converges to a near-optimal value function in a polynomial number of timesteps [1]. This result requires an exploratory policy that, with high probability, tries every state-action pair every L timesteps (for some polynomial L). As convergence in the limit is not a realistic situation, it is necessary to know the convergence properties of the RL algorithms under real conditions, that is, with a limited number of interactions with the environment. The *sample complexity* measures the amount of timesteps for which the algorithm does not behave near optimally or, in other words, the amount of experience it takes to learn to behave well. All the algorithms whose sample complexity can be bounded by a polynomial in the environment size and approximation parameters with high probability are called PAC-MDP (Probably Approximately Correct in Markov Decision Processes). The first family of algorithms proved to be PAC-MDP were those that maintain an internal MDP model: (R_{max} (Brafman & Tennenholtz, 2002), E^3 (Kearns & Singh, 2002) and MBIE (Strehl & Littman, 2005) are examples of these kinds of model-based algorithms. The first model-free algorithm proved to be PAC-MDP was Delayed Q-learning (Strehl *et al.*, 2006)

[1] A *timestep* is defined as a single interaction with the environment that produces a tuple of experience $(s_t, a_t, r_{t+1}, s'_{t+1})$

for a finite state and action spaces.

Although the above statements indicate that the MDPs are solvable in polynomial time, another interesting question arises: is the MDP problem parallelizable? Some optimization problems like minimum spanning tree or the shortest path problem can be solved by algorithms that use cooperating processors. The work by Papadimitriou & Tsitsiklis (1987) demonstrates that MDPs problems are P-complete (this means intuitively that such problems are as hard as any problem in P). P-completeness is taken as evidence that a problem is not satisfactorily parallelized and is likely to be inherently sequential (Greenlaw *et al.*, 1995). However, in cases of deterministic MDPs with finite horizon and infinite discounted horizon, Papadimitriou & Tsitsiklis (1987) demonstrated that they can be solved very fast in parallel. This means that episodic MDPs and non-episodic with discounted rewards MDPs which have a step probability function (probability 1 for one transition specific (s,a,s') and probability 0 for the rest) are parallelizable. The drawback is that these problems are less interesting that the stochastic MDP problems.

The extension of the results for finite state spaces to a richer world models with an infinite number of states and/or actions is an open issue. In this thesis, the learning processes are carried out in environments with continuous state spaces and complex stochastic dynamics. Therefore, I will empirically evaluate the learning processes using quality indicators such as performance instead of dealing with the problem of analytically evaluating the convergence to near-optimal policies.

3.2 Multi-agent Reinforcement Learning

Multi-agent systems (MAS) is the sub-field of AI that aims to provide both principles for construction of complex systems involving multiple agents and mechanisms for the coordination of independent agent's behaviors. The design of a Multi-agent system is a complex task from the point of view of the agent's behavior definition. Although the more extended approach consists of programming these behaviors in advance, it is convenient that the agents have the capability of learning totally or partially the behaviors such that the performance of the agents and of the whole system gradually improves (Stone & Veloso, 2008).

Markov games (also known as stochastic games) are the natural extension of the single RL problem for Multi-agent RL systems (MARL). This framework allows the whole range of collective situations from fully-cooperative to non-cooperative games including general-sum games to be defined (Busoniu *et al.*, 2008). In fully-cooperative games, all the agents share the same reward function and assume an unique maximum Q-function. Amongst the more representative are Team Q-learning (Littman, 2001) and Distributed Q-learning (Lauer & Riedmiller, 2000). In competitive or non-cooperative games, each agent assumes that the rest of the agents want to maximize their loss following a Minimax strategy. A representative example for two agents is Littman's Minimax-Q algorithm (Littman, 1994). Markov games use the joint actions (the Cartesian product of the agents' actions) as a part of the definition of the state-action space. Unfortunately, the exponential dependence in the number of agents and the need to converge to equilibrium as a basic stability requirement of these games, considerably increases the computational cost of the learning process.

Another different approach can be taken if the interactions between the agents are sparse. This approach considers that the agents can learn independently of each other if the task is independent (the dynamics and rewards for each agent are independent of the state/action of the other). In the work by Melo & Veloso (2009) this sparse approach takes into account the fact that the agents can coordinate at specific, local bounded areas of the state space. In their approach, each agent has two independent Q-learning algorithms. When there is not interaction with the other agents, a value function Q is learned. Whereas, when a coordination with another agent is necessary, another Q^C value function is learned that considers the joint actions and/or states. This last table is sparse and requires a computational sampling effort of the same order as that which is necessary to learn Q.

Multi-agent systems, where the agents are totally independent learners, have been studied in several works. For instance, Matarić (1994) and Sen *et al.* (1996) showed that independent reinforcement learning processes are associated with robots in a group for grasping and navigational problems. A similar cooperative multi-robot domain is studied with this independent learning approach in (Fernández *et al.*, 2005). The work by Claus & Boutilier (1998) empirically shows

that convergence is possible in cooperative settings for a Multi-agent system with independent RL processes. Another domain which has been addressed with this independent-learning approach is the Keepaway Soccer task (García *et al.*, 2010; Stone *et al.*, 2005). Inside the problem domain of crowd simulation, a small case study that applies independent learning in a Multi-agent RL framework has been presented by Torrey (2010). Our work assumes that our pedestrian navigational problems can be solved using this sparse approach and, therefore, the agents will be considered as independent learners.

3.3 Generalization of the state space

The tabular Q-learning and Sarsa(λ) algorithms are used in discrete state and action spaces, where each state-action pair corresponds with an entry in the tabular value function. The problems where a discrete state space exists are problems of high abstraction level, or toy problems. In real problems, it is common that, at least, the state space is described by real-valued features, which means that the value function should be able to represent infinitely many state-action pairs. Therefore, techniques that represent the value of continuous state-action pairs using finite resources are needed.

From the point of view of an MDP, the generalization of the state space is known as the structural credit assignment problem. This is defined as the task of figuring out the distribution of rewards across the state space (Connell & Mahadevan, 1997). The idea is to share the reward received in a state s with all the states in which, following the same action sequence that from s, will result in a similar outcome. All the methods for solving this problem are essentially function approximators of one sort or another (Connell & Mahadevan, 1997).

In general, the convergence to an optimal policy using generalization techniques is not guaranteed. For instance, the great success with learning to play the game of backgammon (Tesauro, 1992), that used a neural network as a state generalizer, has not been repeated in other problems. This indicates that using function approximation in RL does not always work well (Boyan & Moore, 1995). Convergence of TD(λ) algorithms using linear function approximators (i.e. function approximators that use linear combinations of fixed basis functions) was

studied by Dayan & Sejnowski (1994). They proved the convergence in the mean for this class of approximators, although the rates of convergence can be too slow for real-world problems. In a re-visiting of the problem, the work developed by Tsitsiklis & Roy (1997) proves stronger convergence results identifying desired characteristics in the TD algorithms such as the on-policy updating, but also undesired ones such as the use of non-linear function approximators. These convergence issues, makes the use of RL techniques for real problems a challenging task whose results depend, among other factors, on the problem domain and the used representation of the state space.

In this work I focus on the problem of state spaces with real-valued features, leaving out of the discussion the problem of continuous action spaces. In order to address the problem of continuous state space I will consider two different approaches: Vector Quantization (VQ), a state-aggregation-based method, and tile coding, which is a linear function approximator.

3.3.1 Vector quantization

Vector quantization (Gray, 1984) discretizes the continuous state space into a finite set of states named prototypes, centers or codewords (depending of the literature) such that the real-valued states are aggregated within each prototype. All the states aggregated to a prototype have the same value, and the region of state space that is covered by a prototype is a *cell*. There are not intersections between the cells, therefore the quantization generates a partition of the state space. Formally, a vector quantizer V_Q of dimension K and size N is a mapping from a vector space (in our context, the state space) in a k-dimensional space, R^k, to a finite set C containing N states, that is, $V_Q : R^k \rightarrow C$.

One important class of vector quantizers are those called Voronoi quantizer or nearest neighbor quantizer. In this class of quantizers, a metric is defined. A sensed state is aggregated to its nearest state in C, also called its prototype. Thus, given C and a state $x \in R^k$, then

$$V_Q(x) = \arg\min_{y \in C} \{dist(x, y)\} \tag{3.16}$$

where *dist* is the metric used to determine the distance between vectors. In this work the Euclidean metric will be used. This metric has been proven successful in RL problems with other domains (García & Fernández, 2012).

An important concept associated to Voronoi quantizers is distortion. The distortion of a vector $\vec{v} \in R^k$ over a prototype $\vec{y} \in C$ is a measure of the cost of assigning the vector $\vec{x}$ to the cell characterized by $\vec{y}$. It is common to use the squared distance as the distortion, also known as the squared-error distortion.

$$D(\vec{x}, \vec{y}) = \sum_{i=1}^{k} (x_i - y_i)^2 \tag{3.17}$$

The goal is to determine a set of N points in R^k (the prototypes or codewords) so as to minimize the distortion from each data point to its nearest center. There is no efficient solution to this problem and some formulations are NP-hard (Garey & Johnson, 1979). One of the most popular heuristics for solving this problem is based on an iterative scheme to find a locally minimal solution (Kanungo *et al.*, 2002). Given a data set, first generate an initial set of prototypes and iteratively: i) classify each data sample assigning it to the nearest prototype; ii) update each prototype $z \in C$ as the center of mass of all data samples which are assigned to the cluster represented by z (that is, calculate the centroid of the Voronoi cell of z). This procedure is known as the Generalized Lloyd Algorithm (GLA) (Linde *et al.*, 1980; Lloyd, 1982). The convergence can be slow and, in practice, a final condition related to the distortion is used (generally that the loss of distortion between two consecutive iterations is less than a quantity θ). In Algorithm 4, a version of the GLA algorithm with an additional stop condition is displayed.

Since the update of the prototypes is a local procedure (it only performs local variations to get the new prototypes), given an initial codebook C_0, the algorithm will find the nearest local minimum in the space of all possible codebooks. Therefore, the initial set of prototypes decisively influence the final result of GLA. Several initialization strategies have been developed to address this problem. I will follow a schema that uses the GLAWFI algorithm with random restarts which appears to be the *de facto* method (Duda & Hart, 1973). This algorithm builds an initial set of prototypes taking them randomly from the data sampled set. Next, the GLAWFI algorithm is executed. If the distortion loss between the last two

Algorithm 4: Generalized Lloyd Algorithm with fixed number of iterations (GLAWFI)

Data: Initial codebook with N prototypes C_0, threshold of distortion loss θ, max number of iterations MaxIt, data set H

Result: C_t, D_t, $GoodCandidate$

begin

t $\leftarrow 1$;

`// Last mean distortion`

$D_{t-1} \leftarrow 0$;

`// Current Mean distortion`

$D_t \leftarrow 0$;

repeat

$D_{t-1} \leftarrow D_t$;

$t \leftarrow t + 1$;

Given $C_{t-1} = \{y_j,\ j = 1, ..N\}$ find a partition of H in cells R_j such that:
$\forall x \in H, x \in R_j \Leftrightarrow dist(y_j, x) \leq dist(y_k, x) \quad k \neq j$

Calculate the new prototype of the cell R_j averaging the data that belong to this cell $y'_j = \frac{1}{\|R\|} \sum_{\vec{x} \in R_j} \vec{x}$ and obtain the C_t prototypes set.

Calculate the mean distortion:

$$D_t = \frac{1}{N} \sum_{l=1}^{N} \frac{1}{\|R_l\|} \sum_{\vec{x} \in R_l} D(\vec{x}, \vec{y_l})$$

until $(D_t - D_{t-1} < \theta)$ *OR* $(t > MaxIt)$;

if $t < MaxIt$ **then**

$GoodCandidate \leftarrow true$;

else

$GoodCandidate \leftarrow false$;

consecutive iterations is greater than a figure $MinLossDistortion$, the algorithm saves the configuration and executes the same instance of the GLAWFI algorithm again to try to get closer to the local minimum. If not, the current final distortion is compared with that of the stored candidate named $CandidateDistortion$ and it is replaced by the new configuration if the current final distortion is smaller than the stored one. Then, another set of initial prototypes is selected from random samples and the process begins from scratch. This set of operations is repeated until a total number of iterations of the GLA, specified in the value $TotalNumIt$, is exceeded. Thus, the total number of iterations used in an instance of the GLA algorithm depends on the how fast the initial set of prototypes converge to the solution. The version of the GLA with random restarts used in this work is described in Algorithm 5:

Algorithm 5: The GLA with random restarts algorithm.

Data: TotalNumIt, MinLossDistortion, MaxIt, the data base H.
Result: $CandidateCentroids$, $CandidateDistortion$

```
1  begin
2      CandidateDistortion ← MAXFLOAT ;
       // assume that CurrentIteration is actualized by the
          GLAWFI algorithm
3      CurrentIteration ← 0 ;
4      Select randomly from H a set of initial prototypes C_0;
5      while CurrentIteration < TotalNumIt do
6          { C_t, D_t, GoodCandidate } ← GLAWFI (C_0,
           MinLossDistortion, MaxIt, H);
7          if GoodCandidate = true then
8              C_0 ← C_t;
9              if CandidateDistortion = MAXFLOAT then
                   // First Time
10                 CandidateDistortion ← D_t;
11                 CandidateCentroids ← C_t;
12         else
13             Initialize C_0 with randomly selected samples of H
               (restart);
14             if CandidateDistortion > D_t then
15                 CandidateDistortion ← D_t;
16                 CandidateCentroids ← C_t;
```

By using vector quantization, the value function can be represented as a table with each entry composed of a prototype-action pair. The main advantage of the method is clear: the tabular RL algorithms can be directly applicable, avoiding convergence problems. In this line, Vector Quantization for Q Learning(VQQL) (Fernández & Borrajo, 2008; García *et al.*, 2010) is a learning schema that uses VQ as the generalization method for the state space and the tabular

version of Q-Learning for the learning process. In VQQL, given a sensed state, s_t, and a selected action, a_t, the Q table entry to be updated is $(V_Q(s_t), a_t)$. The exploration–exploitation trade-off is modeled using an ε-greedy policy, although different strategies could be implemented. The complete algorithm is described in Algorithm 6.

A trade off is necessary between accuracy and efficiency using VQ with GLA. In order to achieve accuracy, the cell size of the generalizer should be small to provide enough resolution to approximate the value function. But, as the cell gets smaller, the number of cells to cover the space grows exponentially and, therefore, more data is necessary to estimate each cell value Santamaria *et al.* (1997).

Algorithm 6: Single-agent VQQL Algorithm

Data: end condition (number of iterations or error threshold)
Result: Q^* and VQ

begin
  Generate the set T of samples of the state space S interacting with the environment using an exploratory policy;
  `/* Discretize the state space */`
  Use GLA to obtain a state space discretization $C \in S$ from the sample set T;
  Let $VQ : S \rightarrow C$ be the function that, given any state in S, returns the discretized value in C;
  `/* Learn the Q-Table */`
  while *end condition not reached* **do**
    Get an experience tuple $< s_1, a, s_2, r >$ by interacting with the environment;
    Map the states of the experience tuple using VQ. Each acquired tuple of experience $< s_1, a, s_2, r >$ is mapped to $< VQ(s_1), a, VQ(s_2), r >$;
    Apply the Q-Learning update function to learn a tabular value function Q: $C \times A \rightarrow \Re$, using the mapped experience tuple;

The final condition is defined in our case as a fixed number of iterations determined empirically.

The VQQL algorithm assumes that dataset T is significant, that is, the dataset contains the relevant information to represent the whole state space. The VQQL algorithm and similar approaches, have been used with good results in different domains (Fernández & Borrajo, 2008; García *et al.*, 2010; Mahadevan & J.Connell, 1991).

3.3.2 Function approximation and Tile coding

In an RL process, an agent interacts with its environment to collect experience tuples (s_t, a_t, r_{t+1}) that will be used to approximate a value function. We can understand each tuple specifying an example of a desired input-output behavior of the estimated value function. In this sense, each tuple indicates that the value of the estimated function for the state (s_t, a_t) should be more like r_{t+1}. Viewing the set of experiences as a conventional training set enables us, in principle, to use any of a wide range of function approximation methods for value prediction including neural networks, decision trees or multivariate regression. However, many approximation methods assume a static training set over which many passes occur. This makes them difficult to use in an RL problem, where the data is acquired on-line while the agent interacts with the environment. Besides, the values r_t of the training data are non stationary (they evolve over time) because they have been generated using bootstrap strategies as commented previously in the discussion of the TD methods. Methods that cannot easily handle nonstationarity are less suitable for RL problems (Sutton & Barto, 1998).

A measure of the performance of the approximation function method is the mean-squared error (MSE) over one distribution P of the inputs. In a learning problem, the inputs are the state-action pairs (s, a) and the target function to approximate is the true value function $Q^*(s, a)$, so the MSE for an approximation Q_t using a parameter vector $\vec{\theta}$ is:

$$MSE(\vec{\theta}) = \sum_{s \in S} P(s)[Q^*(s, a) - Q(s, a)]^2 \tag{3.18}$$

where $P(s)$ is a distribution weighting the errors of different states, and the $\vec{\theta}$ vector has a fixed number of real-valued components $\vec{\theta_t} = (\theta_t^1, \theta_t^2, \ldots, \theta_t^N)$. In an

on-line RL process, this distribution P is the distribution from which the states in the training examples appeared in the interaction with the environment. A goal in terms of the MSE would be to find a global optimum, that is, a parameter vector $\vec{\theta^*}$ for which $MSE(\vec{\theta^*}) \leq MSE(\vec{\theta})\ \forall\vec{\theta}$.

One main strategy to approximate the $\vec{\theta^*}$ vector is to minimize the MSE error of the observed examples. The gradient descent method achieves this by adjusting the parameter vector after each example by a small amount in the direction that would most reduce the error of that example:

$$\vec{\theta_{t+1}} = \vec{\theta_t} - \frac{1}{2}\alpha\nabla_{\vec{\theta_t}}[Q^*(s_t,a_t) - Q(s_t,a_t]^2 = \vec{\theta_t} + \alpha[Q^*(s_t,a_t) - Q(s_t,a_t)]\nabla_{\vec{\theta_t}}Q(s_t,a_t) \tag{3.19}$$

where α is a positive step-size parameter.

Although during the learning process we do not know the true value $Q^*(s,a)$, we can approximate it in each interaction t by the value $v_t = r_{t+1} + \gamma Q_t(s_{t+1}, a_{t+1})$. Therefore, the general gradient-descent method for action value function prediction is

$$\vec{\theta_{t+1}} = \vec{\theta_t} + \alpha[v_t - Q(s_t,a_t)]\nabla_{\vec{\theta_t}}Q(s_t,a_t) \tag{3.20}$$

If v_t is an unbiased estimate (i.e. $E[v_t] = Q^*(s_t,a_t)$ for each t, then $\vec{\theta_t}$ converges to a local optimum if α is a decreasing step size parameter.

When we use a function approximator based on a parameter vector $\vec{\theta}$, we assume that there are many more states than components of $\vec{\theta}$. Better approximations at some states are at the expense of worse approximation to other states. In this situation, distribution P specifies how these trade-offs should be made (Sutton & Barto, 1998).

Tile coding (Sutton & Barto, 1998; Szepesvári, 2010) is a linear function approximation with binary, sparse features. It is based on the Cerebellar Model Articulation Controller (CMAC) structure proposed by Albus (1975). It constitutes a specific case of the parametrized function approximators family where the functions are approximated with a linear combination of weighted binary-valued parameters.

In tile coding, the space is divided exhaustively in partitions named tilings.

Each tiling partitions all the space so there are as many partitions as tilings. Each element of a specific tiling is a tile and, given a point in the state space, there is only one active tile per tiling associated to this point (see the Figure 3.3).

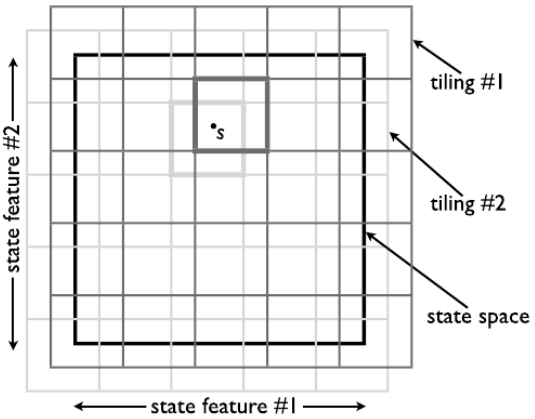


Figure 3.3: Tile coding with two tilings. Thicker lines indicate the active tiles for state s. Image from Whiteson et al.'s Technical Report AI-TR-07-339 of the University of Texas at Austin 2007

Given m tilings and k tiles per tiling, then $m \cdot k$ tiles exist. A binary vector $\vec{\phi}$ indicates the active tiles in each interaction at time t, and the vector $\vec{\theta}$ stores the value of the tiles. Therefore, for each tile i, $\phi_i(s)$ indicates if it is active (value 1) or not (value 0) for state s, and a weight stored in a table $\theta(i)$ indicates its value. If the set of actions is discrete as in my framework, the value function for each action Q^a and state s is represented as a lineal combination as described in the Equation 3.21.

$$Q_t^a(s) = \vec{\phi}^T \vec{\theta_t^a} = \sum_{i=1}^{m \cdot k} \theta_t^a(i)\phi_i(s) \tag{3.21}$$

The super index T means the matrix transpose.

The code of a point of the state space is given by the binary features $\phi(i)$ that have value 1, the rest remaining with value 0. Therefore, in practice the sum of Equation 3.21 is not over all tiles of all tilings since only one tile per tiling is active in the codification of a state.

Although tile coding was used initially as a a generalizator in a connectionist

context, it is easy to use it in a reinforcement learning process to form a piecewise-constant approximation of the value function for each action. Given an MDP, the value estimate of a state s_t in a tuple of experience $(s_t, a, r_{t+1}, s_{t+1})$ is updated computing the TD error

$$\Delta Q^a_{t+1}(s) = r_{t+1} + \gamma Q^{a'}_t(s_{t+1}) - Q^a_t(s_t) \tag{3.22}$$

and updating the particular expression of the gradient descent equation 3.20 for the case of the linear representation of each Q^a:

$$\theta^a_{t+1}(i) = \theta^a_t(i) + \frac{\alpha}{m}\phi_i(s)\Delta Q^a_{t+1}(s) \tag{3.23}$$

where α is the learning rate. Note that

$$\nabla_{\vec{\theta_t}} Q^a_t(s_t) = \nabla_{\vec{\theta_t}}(\sum_{i=1}^{m\cdot k} \theta^a_t(i)\phi_i(s)) = \vec{\phi} \tag{3.24}$$

in the particular case of a linear approximator. As stated before, it is not necessary to update the $m \cdot k$ weights, only the m associated with the active tiles.

The width of the tiles, that is, the number of divisions in each dimensional axis and the number of tilings determine the granularity of the function approximator. The denser the tiling, the finer and more accurately the desired function can be approximated, but the greater the computational costs (Sutton & Barto, 1998). Thus, there is a resolution/accuracy trade-off in the same way as with VQ.

The algorithm Sarsa(λ) described in a previous section can be easily adapted to use a linear function approximator (see the Algorithm 7). We note as $\wp^s_a$ the set of binary components of $\vec{\phi}$ that are active for the state s and the action a.

A critical problem of tile coding is the memory requirement, that is exponential in the number of dimensions. Using a direct implementation, a value to store each θ^a_i corresponding to tile i is necessary. In my case, the number of dimensions depends on the specific scenario, for example in some experiments more than 30 parameters are used to specify the state space. Supposing a uniform division of each dimension in k pieces, more than k^{30} tiles would be necessary. One trick to reduce memory requirements is to use a Hash function. With hashing, a pseudo-

Algorithm 7: Linear gradient-descent Sarsa(λ)

Data: number of episodes k, finite set of actions A

Initialize $\vec{\theta}$ arbitrarily and $\vec{e} = \vec{0}$;
repeat
  $s \leftarrow$ initial state of the episode;
  for *all* $a \in A$ **do**
    $\wp_a^s \leftarrow$ binary features present in(s, a);
    $Q^a \leftarrow \sum_{i \in \wp_a^s} \theta(i)$;
  $a \leftarrow \underset{a}{\operatorname{argmax}}\, Q^a$;
  With probability ϵ: $a \leftarrow$ a random action $\in A$;
  repeat
    $\vec{e} \leftarrow \gamma\lambda\vec{e}$;
    for *all* $a' \neq a$ **do**
      for *all* $i \in \wp_a^s$ **do**
        `// Replacing traces`
        $e(i) \leftarrow 0$;
    for *all* $i \in \wp_a^s$ **do**
      $e(i) \leftarrow 1$;
    Take a tupla of experience (s, a, r, s');
    $\delta \leftarrow r - Q^a$;
    `//` ϵ`-greedy policy implementation`
    for *all* $a \in A$ **do**
      $\wp_a^{s'} \leftarrow$ binary features present in (s', a);
      $Q^a \leftarrow \sum_{i \in \wp_a^{s'}} \theta(i)$;
    $a' \leftarrow \underset{a}{\operatorname{argmax}}\, Q^a$;
    With probability ϵ: $a' \leftarrow$ a random action $\in A$;
    `//` `end of the` ϵ`-greedy policy implementation`
    $\delta \leftarrow \delta + \gamma Q^{a'}$;
    $\theta \leftarrow \theta + \alpha\delta\vec{e}$;
    $a \leftarrow a'$;
  until *a terminal state of the episode arrives*;
until *complete* k *episodes*;

random large tiling collapses into a much smaller set of tiles. It produces new tiles consisting of non-contiguous, disjointed regions randomly spread throughout the state space, but that still form an exhaustive tiling (Sutton & Barto, 1998). The key property that makes the use of hashing feasible is that the state space is sparse in terms of its use by a policy. This means that high resolution of the method is needed only in a small fraction of the state space. Although the change of resolution is not covered in this explanation of tile coding, the study by Whiteson *et al.* (2007) proposes an adaptive tile coding where the density of tiles increases in the more visited regions of the state space.

The convergence of TD(λ) algorithms with linear approximators has been widely analyzed. Tile coding with hashing is a simple, computationally efficient function approximator that has demonstrated broad empirical success in the field of reinforcement learning (Stone *et al.*, 2005; Sutton, 1996).

3.4 Knowledge Transfer techniques in RL

The goal of knowledge transfer techniques is to transfer an inductive bias to the current learning task (named the *target task*) from previous learned tasks (named the *source task*), avoiding having to start the learning process from scratch. This has two main beneficial effects: i) it offsets initial performance in the target task (which is of great value when operating with robots or harmful/delicate processes) and ii) it achieves superior performance, faster than learning from scratch. The knowledge transfer can occur at the initialization of the learning process, during specific situations or along the whole process. Therefore, in knowledge transfer, we have three different research issues: i) what to transfer, ii) how to transfer and iii) when to transfer (Pan & Yang, 2010). Knowledge transfer is also known by other names such as transfer learning, inductive transfer, metalearning, multitask learning, life-long learning and incremental/cumulative learning amongst others (Thrun & Pratt, 1998).

Knowledge transfer was first studied in psychology with the works by Thorndike & Woodworth (1901) which focused on the improvement in the human mind of one mental skill based on the training of other (similar) mental functions. Also it has been studied in the machine learning field to transfer between machine

learning tasks (Caruana, 1995), for planning tasks (Fern *et al.*, 2006) and in the context of cognitive architectures (Laird *et al.*, 1986).

The taxonomy created by Lazaric (2012) classifies the transfer problems into three groups according to the problem setting: i) transfer from source to target with fixed domain ii) transfer across tasks with fixed domain iii) transfer across tasks with different domains. According with this taxonomy, most of the early literature on transfer in RL focused on the source-to-target setting, while the most popular scenario in recent research is the general problem of transfer from a set of source tasks. In his survey, Taylor & Stone (2009) classifies the existing transfer methods into five different groups: 1) Methods in which the source and the target task use the same state variables and actions, 2) Methods with the same property as group 1 but using a group of source tasks instead of only one source task. 3) Methods with different state pace or action space between the source task and the target task, but no explicit task mapping is used, 4) The same situation as group 3) but inter-task mapping is used 5) techniques that learn the inter-task mapping.

In group 1) works with different approaches are considered. Asada *et al.* (1994); Selfridge *et al.* (1985) use incrementally complex source tasks to approximate to the target task. The first one uses a source task that approximates to the transition function of the MDP of the target task. The second one approximates to the target task using incrementally harder source tasks in a "learning from easy missions" paradigm. Other set of works transfer partial policies instead of complete tasks. These works assume that the task to learn has a hierarchical structure with several sub-tasks that can be discovered using similarities in the state space (Andre & Russell, 2002) or a Bayesian estimator (Ravindran & Barto, 2003).

In group 2) all the methods use multiple source tasks. Some methods use all the experienced source tasks when learning a novel target tasks while others choose a specific subset of the source tasks. The use of one approach or the other depends on the assumptions about task distribution: if tasks are expected to be similar, there is no need to select a subset (Taylor & Stone, 2009). The type of knowledge transferred among the source tasks and the target task vary in the different approaches. In the work Perkins & Precup (1999), the transition

function $P : S \times A \times S \rightarrow [0, 1]$ may change after the agent reaches the goal. Upon reaching the goal, the agent is returned to the start state and another transition function is drawn randomly from a fixed distribution. The agent is provided with a set of hand-coded options which assists him in the task learning. The agent learns a single action-value function over these options allowing it to reach the goal more quickly in tasks with novel transition functions. In the work of Foster & Dayan (2004), no options are transferred, but there are sub-tasks that can differ in the placement of the goal state. The sub-tasks are identified in the learned source tasks using an expectation-maximization algorithm (EM) (Dempster *et al.*, 1977). As the sub-tasks represent unchanging problems, they can be used as blocks that completely solve a part of the new task.

The methods of group 3) allow the source task and target task to have different state variables and/or actions. There are no explicit mappings between the tasks, but the agent uses abstractions over the MDP that are invariant to changes in the state and/or action spaces. The work by Banerjee & Stone (2007) transfers identifying forks in tree-based games: states where the player could win on the subsequent turn regardless on the opponent's movement. The values of the features of the source tasks for these forks are used in a variety of target tasks. Another approach is Relational Reinforcement Learning in which the learner reasons about a state in propositional form by constructing first-order rules. Croonenborghs *et al.* (2007) uses learned values of specific state-action pairs to build a relational decision tree. This tree predicts which action will be executed by the agent in a given state. Lastly, the trees are mined to produce relational sub-tasks which are directly used in the target task assuming that the sources/target tasks are similar enough.

In group 4) the methods use inter-task mapping. There are approaches that assume that this mapping is provided to the learner (Taylor *et al.*, 2007). Another way is to transfer via learned advice or preferences as in the work of Torrey *et al.* (2005). In this work, the system automatically extract such advice from a source task by identifying actions with higher Q-values than others. This advice is transferred as relative preferences for different actions in different states.

The group 5 methods) deal with the problem of learning the mapping between tasks. There are two approaches depending on whether the transition function

is known or not. If it is fully known, a rule graph (i.e an abstract representation of a deterministic full behavior) can be obtained. In the work Kuhlmann & Stone (2007), the learner first trains on a series of source task games, storing not only the value functions but the rule graphs. When a novel task is presented, it first builds its rule graph and tries to make an isomorphism with the existing graphs of the source tasks. Once the relationship between a source task and the target task is found, the action-value function of the source task is mapped to the states of the target source. In the work by Liu & Stone (2006) knowledge of the transition function is not assumed. It uses qualitative dynamic Bayes networks to summarize the effects of actions on state variables. From this knowledge structure, a mapping between states and actions can be automatically found using graph-mapping techniques.

In this work I will use two methods for knowledge transfer which are described below: *complexification* and *probabilistic policy reuse*.

Complexification Taylor & Stone (2007) is a group 4 method that uses inter-task mapping to transfer the learned knowledge between tasks that differ in the function approximator (the state space generalizer), or in the learning algorithm. In a complexification process, the function approximator is changed over time to allow for more representational power. The agent can learn using a simple representation initially, and then switch to a more complex representation later. One case of complexification is represented in Algorithm 8 extracted from the work by Taylor & Stone (2007). In this case, the algorithm describes the process for transferring between value function approximators (FA) with different parametrizations of state variables. The weights (parameters) of a learned FA are used as needed when the agent learns a target value function representation. I will utilize this algorithm in the iterative schema for VQQL described in chapter 6. The performance analysis of the algorithms in that chapter will prove that our adaptation of the complexification algorithm significantly improves the learning processes.

Probabilistic policy reuse (PPR) (Fernández & Veloso, 2006; Fernández *et al.*, 2010) is a group 3 method which generates a library of learned policies that can differ in the state and the action spaces. The learner improves his/her exploration by probabilistically including the exploitation of the policies of the library. When

Algorithm 8: Complexification

```
1 Train with a source representation and save the learned FA_src;
2 while target agent trains on a task with FA_target do
3     if Q(s,a) needs to use at least one uninitialized weight in
      FA_target then
4         Find the set of weights W that would be used to
          calculate Q(s,a) with FA_src;
5         Set any remaining uninitialized weight(s) in FA_target
          needed to calculate Q(s,a) to the average of W;
```

the agent addresses a new task, at every timestep, it can choose to: exploit a learned source task policy, exploit the current best policy from the target task or randomly explore. Therefore, the RL process is able to probabilistically bias the exploration of the domain with a pre-defined past policy. The method introduces an exploration strategy (π-reuse) able to bias a new learning process with a past policy. The goal of π-reuse is to balance random exploration, exploitation of the past policy and exploitation of the new policy. Given a past learned policy π_{past}, the exploratory policy of the learning process for the new task will follow this schedule:

$$\begin{cases} \psi & \text{choose the } \pi_{past} \text{ policy} \\ (1-\psi)\epsilon & \text{choose an aleatory action} \\ (1-\psi)(1-\epsilon) & \text{choose the greedy policy} \end{cases} \tag{3.25}$$

PPR method also uses in its general form a similarity function that allows the estimation of the usefulness of past policies with respect to learning the new task. This function is important to discriminate useful policies from the created library for a specific problem. Another additional problem is the library building. In Fernández & Veloso (2006) the authors present PLPR algorithm, an incremental method to build a library of policies. When solving a new problem by policy reuse, PLPR algorithm determines how different the learned policy is from the past policies as a function of the effectiveness of the reuse. If the past and new policies are 'sufficiently' different, PLPR decides to add the new policy to the library of policies.

Algorithm 9 shows a Q-learning version which uses PPR. The v parameter decays the value of ψ inside each episode and, thus, so does the influence of the past policy π_{past} in the last steps of the episode.

Algorithm 9: Q-learning with PPR

Data: π_{past}, K, H, ψ, v
Result: W, Q, π_{new}
Initialize $Q(s,a) = 0 \forall s \; in S, a \in A$;
for *k=1 to K* **do**
  Set $\psi_1 \leftarrow \psi$;
  for *h=1 to H* **do**
    With probability of $\psi_h, a = \pi_{past}(s)$;
    With probability of $1 - \psi_h, a = \epsilon - greedy(\pi_{new}(s))$;
    Receive current state s', and reward $r_{k,h}$;
    $Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha[r + \gamma max_{a'} Q(s',a')]$;
    Set $\psi_{h+1} \leftarrow \psi_h v$;
    Set $s \leftarrow s'$;
  $W = \frac{1}{K} \sum_{k=0}^{K} \sum_{h=0}^{H} \gamma^h r_{k,h}$;

This algorithm also calculates the mean value of the sum of the discounted returns per episode W. This value can be used as an estimate of how similar the policy target π_{new}, is to the policy π_{past}.

PPR has been used in different types of problems. Fernández & Veloso (2013) demonstrates that PPR contributes to the learning of the structure of a domain. It can be used to identify classes of similar policies grouping them by their usefulness in solving a task. In Chevaleyre & Pamponet (2012), the authors define adaptive PPR. A new approach based on PPR that optimizes the transfer rate between tasks to avoid that transfer between dissimilar policies can slow down the learning. The work by Martinez-Plumed *et al.* (2013) uses PPR with success in the context of a learning framework for structured prediction using user-defined operators and functional programming language for the representation of rules and operations.

Before concluding this section it is necessary to mention the metrics used to measure the influence of knowledge transfer methods in the learning process. There are four main metrics (Taylor & Stone, 2009):

1. Initial jumpstart. This considers the agent's initial performance in a target task. This metric capture the influence of the transferred task in the performance before learning occurs.

2. Asymptotic performance. This compares the final performance of learners in the target task both, with and without transfer. It can also be understood as a measure of the acceleration of the learning process.

3. Total reward. This is the total reward accumulated during training. Improving the initial performance and achieving a faster learning rate will help the agent accumulate more on-line rewards. This metric is more appropriate for tasks that have a well-defined duration.

4. Time to threshold. This is the time (or the number of episodes) necessary to reach a specific performance. The threshold is dependent on the domain and the learning method.

3.5 Applications of RL

Reinforcement learning has a solid and extended tradition in the fields of robotics and machine learning. Inside these knowledge areas there are many research and practical applications that use RL as the core or part of their implementation. I present a non-exhaustive list of them in different areas that reveals the capability of RL to be applicable in a wide range of different problem domains. Among the successes of practical reinforcement learning applications in different fields I will highlight the work by Kohl and Stone in Robotics to learn quadrupedal locomotion (Kohl & Stone, 2004), the work by Abbeel *et al.* (2010) to control an autonomous helicopter flight and a control system to an autonomous robot for underwater cable tracking (El-Fakdi & Carreras, 2008). Also of note is the RL work of Maja Matarić in multi-robot domains (Matarić, 1997). In operations research noteworthy is the work by Rusmevichientong *et al.* (2006) that uses online preference data for vehicle pricing, the work on vehicle routing that presents the H-Learning algorithm (Moody & Saffell, 2001), and Proper & Tadepalli (2006) which presents a scalable system for product delivery. In games the success of

using RL to play Backgammon (Tesauro, 1995) is famous although it has been used in other games to improve performance such as chess (Baxter *et al.*, 1998) or Go (Ekker *et al.*, 2004). In Economics it has been used in the development of a trading system (Singh *et al.*, 2002) and in a business simulator (García *et al.*, 2012). In the field of multi-agent systems, RoboCup Soccer (Stone *et al.*, 2005) is a recognized forum to test RL in real and simulated autonomous robots with collaborative tasks. RL has also been used in communication to implement error-recovery strategies in spoken dialog systems (Frampton & Lemon, 2008), and in telecommunication networks to predict loss of information caused mainly by congestion (Esfahani & Analoui, 2008). Moreover, it has been used in mobile cellular communication systems to allocate the communication resource (bandwidth) so as to maximize the service provided to a set of mobile callers whose demand for service changes randomly (Ranjan & Phophalia, 2008). In Mechanical Engineering, an approximated dynamic programming based strategy has been designed for realtime energy control of parallel hybrid electric vehicles to develop a fuel-optimal control (Li *et al.*, 2008).

3.6 Reinforcement learning in animation and simulation

Although RL has been extensively applied in the fields of Robotics, Economics and Operations research (see Section 3.5), to the best of my knowledge, the use of RL in simulation and/or animation is scarce.

Several works use RL in Motion-graphs-based animation. Motion graphs (Kovar *et al.*, 2002) are commonly used to represent plausible transitions between motion segments to create animations. By traversing the graph, natural-looking and complex motions can be synthesized. Efficient goal-directed traversing of this data structure can be performed using learned controllers. The work by Treuille *et al.* (2007) is the main example of the use of RL in this context. In this work, a new approach to realtime character animation with interactive control is presented. Their system has two main components: a motion engine blends through captured motion clips to reproduce realtime human animation, while a control

policy determines the best sequence of clips to achieve some multivariate control objective. The controller is the component that uses RL to efficiently traverse the motion graph from a specific state. It must decide which sequence best achieves the motion goal from the initial situation. The proposed goals are 'navigation', 'spinning navigation' and 'obstacle avoidance' and a different controller exists for each one. The value functions are represented by a linear combination of low-dimensional basis functions that produce complex user navigation and obstacle avoidance tasks. This work was extended by Lee *et al.* (2009) who introduce weighted interpolation on the motion clips to enable more precise controls with a significantly reduced amount of data. The RL module constructs intelligent mechanisms that makes decisions on which sequence of clips to concatenate to produce effective long term behavior. Moreover, the authors use a basis refinement procedure to enhance the power of the value function near critical decision boundaries. The refinement can adapt very coarse initial basis functions to create effective controllers for highly complex tasks. In Wampler *et al.* (2010), the authors extend the previous works with motion graphs adding game theory to generate two-player adversarial games. The learned controllers take into account the interactions between the rules of the game and the motions generated from a parametric motion graph. Another extension of the same research group uses a space-time planner to determine the sequence in which controllers must be executed to reach the goal location (Levine *et al.*, 2011). By planning in space and time, the planner can discover paths through dynamically changing environments. The work by Ikemoto *et al.* (2005) tries to bridge the gap between animation control and high-level control logic. The proposed system consists of a motion graph and a value function to decide which transitions between animations to take. The value function is approximated using RL in a discretized state space. The controller learns a parametric value function for choosing transitions at the branch points in the motion graph. It is coupled with a global path planner to create realistic motion in changing environments.

A RL controller can also be used to select frames, or pieces, to construct the animation. The work by McCann & Pollard (2007) uses RL to pre-calculate a character animation controller that assembles a motion stream from short motion fragments. It chooses each fragment based on current player input and the previ-

ous used fragment. Other work uses value iteration to construct a sample-based value function that selects frames from a collection of motion capture data to animate and control avatars that represent boxers sparring (Lee & Lee, 2004).

Specific techniques used in RL have also been adapted to animation and/or simulation problems. The paper by Lo *et al.* (2012) proposes a method that facilitates the application of RL to character control generalizing the state space definition. Instead of using a specific parametric description of the state space, the agent perceives the environment directly and uses a hierarchical regression algorithm that adapts to the complexity of the scene. The value function is then represented by a set of regression trees (which represents the state space) associated with every action. The system uses the Fitted-Q algorithm to learn an action controller that permits an agent to navigate inside a virtual environment without colliding with the objects. The work by Lee & Popović (2010) uses inverse RL to determine the appropriate reward function of several behaviors inside an RL animation framework. The authors introduce a method for inferring the behavior styles of character controllers from a small set of examples. From these modeled controllers, they apply inverse RL to determine the reward functions that define the behavior. The authors show that the calculated reward function representing a behavior style can be applied to a variety of different tasks, while preserving the key features of the style present in the given examples. In addition, the author can refine, through the use of examples, the behavior so that it has better generalization properties.

One area of RL that has gained momentum by approaching the simulation field is Motivated Reinforcement Learning (MRL) (Singh *et al.*, 2004, 2010). Psychologists call behavior intrinsically motivated when it is engaged in for its own sake rather than being a step forward in solving a specific problem. This behavior, if implemented inside a learning system, can provide the capability of autonomously learning different skills, or designing adaptive agents that respond to unpredictable changes in their environment, as occur in interactive games. Singh *et al.* (2004), use intrinsically motivated RL aimed at allowing artificial agents to construct and extend hierarchies of reusable skills to give the agent competent autonomy. Hester & Stone (2012) presents an intrinsically motivated model-based RL algorithm that learns models of the transition dynamics of a

domain using decision trees. The experiments demonstrate that the algorithm learns an accurate model of a domain with no external rewards and that the learned model can be used to perform tasks in the domain.

Inside the field of games, simulation games such as 'The Sims', 'Black and White' and 'Creatures' give a human player control over the simulated world. Players can modify the game environment and interact with the non-player characters in certain ways. Although these games do not use learning techniques, they are the inspiration to use MRL in virtual reality, as explained in the work by Merrick & Maher (2006, 2007). The idea is to make games with more in-game modification capabilities. Specifically, this work focuses on the design of non-player characters (NPCs) which can respond autonomously to unpredictable open-ended virtual worlds. The approach uses MRL with context-free grammars (CFGs) to represent character reasoning in unpredictable, evolving worlds. The motivation component is based on task independent components such as novelty or interest. It can be used to direct learning towards different tasks without requiring prior knowledge of what those tasks may be or when they should be performed. The learning procedure is continuous because the character can change task learning, guided by motivation. The problem is how to represent environments which may change over time. The authors propose the use of CFGs to make this variable representation. They apply motivated RL to design adaptive characters for the Second Life virtual world. This initial work has been extended in the book by Merrick & Maher (2009), that illustrates the use of motivated reinforcement learning by applications in simulated game scenarios and in live, open-ended virtual worlds like 'Second Life'.

In the area of video games, of note is the Xbox 360 game entitled 'Project Gotham Racing 3'. It includes an RL algorithm called Adaptive Modeling and Planning Systems (AMPS) (Kochenderfer, 2006) developed by the Applied Games Group of the University of Cambridge (UK). This algorithm is used in the realtime game to learn from experience to drive simulated cars with the shortest possible lap times. The learning algorithm is efficient enough to permit frame rates of 60 fps.

The use of RL to create autonomous behavioral characters is anecdotal as far as I know. Apart from the mentioned works by Merrick *et al.*, Blumberg *et al.*

(2002) presents an autonomous animated dog that is trained using RL to react to acoustic patterns. The use of RL with autonomous agents is considered in Torrey (2010), who proposed a microscopic framework for learning behaviors that are a mix of socialization and goal-oriented traffic. Their approach uses a discrete space and focuses on the learning process but does not address the analysis of the learned behaviors.

3.7 RL tools

In recent years, the RL researchers community has developed a set of software applications for providing basic, general-purpose RL frameworks or functionalities. These applications are mainly conceived to provide an easy-to-manage platform for new researchers in RL or to experiment with new ideas using simple examples. The oldest example of these applications is Brian Tanner's RL-Glue (Tanner & White, 2009). It provides a standard interface to connect an RL agent with an environment even if they are written in different languages. It is a low level protocol to connect agents and environments and it is single-agent oriented. The user has to program the agent and the environment capabilities. A second example is the RL Toolkit provided by the RLAI group of the University of Alberta. It is mainly a collection of demos to initiate in the world of RL although it also provides utilities such as the tile coding generalizer and a graphic interface. Another tool available is Teaching-Box (Ertel *et al.*, 2009), a Java-based Machine Learning toolkit focused on robot learning with educational purposes that includes RL algorithms. The University of York has also recently introduced an RL library named YORLL. It is a research tool to quickly develop ideas in RL that support multi-agent environments and it is currently under development. RL-Park is another Java RL library developed by Thomas Degris oriented to robotics that uses Zephyr as a visual debugger.

3.8 Chapter highlights

1. Sarsa(λ) and Q-learning are Temporal Difference (TD), general purpose algorithms which have been used successfully in many different problem

domains.

2. Different variants of these algorithms (the tabular versions and the linear approximator versions) have theoretical convergence guarantees after infinite visits to the states (convergence in the limit). This fact implies that the learned solutions are sub-optimal.

3. The multiagent approach adopted in this thesis considers the agents as independent learners. This approach has been used with success in other problem domains by several authors.

4. The use of real-valued features to describe the state space implies the use of generalization techniques such as vector quantization (an aggregation method) or tile coding (a linear function approximator).

5. The use of knowledge transfer techniques can accelerate and improve the learning process. They can be classified according to the similarities between the donor task and the recipient task in terms of sharing, or not, the state and action spaces, performing, or not, the same task (although in different state spaces) part of the task (subtask transfer), or the capability of learning the mapping between the tasks.

6. The use of RL in animation and simulation is highly limited. Many works are dedicated to animation where RL is used to learn controllers capable of assembling frames, selecting pieces of motions or finding optimal paths inside a motion graph. Others extend specific RL techniques (such as inverse RL) or approaches (such as Motivated RL) to the animation field. The use of RL for pedestrian modeling and/or simulation is restricted, as far as I know, to the referenced work by Torrey and this thesis.

Chapter 4

Motivation and Objectives

Pedestrian simulation is an active research area in the simulation field. The simulation of pedestrians is needed in technical environments, entertainment and also for optimizing transport systems and public facilities. Training environments (known as serious games), advanced video games, and movies need the simulation of pedestrians, from individuals to crowds. Two important commercial examples are Legion (used in the Sydney Olympic games to assess pedestrian circulation through the Olympic Park) and Massive (used in The Lord of the Rings and Avatar to simulate crowds). However, achieving realistic pedestrian behaviors (individually and as a group) using an agent-based approach (that is, through the specification of local interactions) is a challenging task. First, because multi-agent architectures are intrinsically complex and specific restrictions have to be guaranteed (autonomy in sensing and actuation, decentralized control, message-passing protocol definition). Second, it is not easy to move a group of pedestrians without falling into mechanical or choreographic movements that produces a feeling of artificiality. Pedestrian groups have their own dynamic properties that differentiate them from other natural and artificial moving groups (i.e. the emergence of specific collective behaviors (Helbing, 2004), route-planning based on the pedestrian's own knowledge of the environment (Koh & Zhou, 2011)).

In this chapter, the use of RL for pedestrian simulation is put forward and Section 4.1 enumerates the main reasons. Then, Section 4.2 explains several considerations that lead to methodological decisions. Lastly, in Section 4.3, the objectives of this work are set out.

4.1 Motivation

The main interest for using RL in a pedestrian simulation framework is given because it is a biologically-inspired approach that provides a new perspective on the problem of the agents' behavior definition. In humans, one of the basic learning types, known in cognitive psychology as *operant conditioning*, is associative learning in which the learner associates behaviors with their consequences. In a later feedback phase, the learner modifies his/her conduct besed on the answers from the environment. In human beings, early learning tasks such as walking are based on operant conditioning theory. RL shares the assumptions of this type of learning. Thus, using RL, the user does not have to specify guidance rules or other models of behavior for pedestrians. Only high level restrictions over the behavior of the agents[1] are included in the framework as feedback signals as immediate rewards (e.g. reaching the goal is good and the agent gets a positive reward; going out of the borders is bad and produces a negative reward).

A second reason for using RL is the division between the process to acquire knowledge (the learning process) and the exploitation of this knowledge (the simulation process). The learning process can be performed off-line in a first step and the learned behaviors can be exploited on-line in a second step (simulation). The learned behavior (also known as the policy of the agent) constitutes the decision-making unit of each agent and is derived efficiently from the greedy traversal of the learned value function in simulation time. Therefore, the decision-making process is highly appropiate for *real-time simulations*.

The third reason is the capability of generating *heterogeneous behaviors*. The learned behaviors are different for each agent, providing variability in the simulation. This *heterogeneity* is intrinsic to the learned behaviors because each agent learns from its own experiences interacting with the environment, in contrast to other models in which the creation of variations is performed by the design of different sets of rules or stochastic variations of the parameters.

A fourth reason is the fact that policies obtained by RL are capable of operating at different levels, as demonstrated in many navigational problems in robotics.

[1]The words 'pedestrian' and 'agent' are equivalent when I refer to the entity that learns in the framework and will be exchanged throughout the text.

In Sections 1.1 and 2.3.7, mention was made of the strategies that crowd simulators use to model behavior at different levels. In this work I will study if the pedestrians' learned behaviors are complex enough to develop characteristics of high level behaviors (tactical like route-choice and strategical like route planning) as well as characteristics of low level behaviors (operational tasks such as collision management).

Another reason is that RL seems promising to address the problem of generating emergent behaviors. As explained in Section 2.2.11, several works have successfully used with optimization processes to generate pedestrian simulations. Specifically, the work by Guy *et al.* (2012) recently demonstrated that a pedestrian model based on the principle of least effort is capable of generating emergent collective behaviors. Moreover, Helbing & Johansson (2009) suggests that in real pedestrians, a learning process exists to optimize the automatic response that minimizes collisions and delays. Being RL algorithms optimization-based processes, they might also be capable of generating emergent collective behaviors.

Finally, RL has an important technical background developed over the last three decades that provides auxiliary tools to confront the challenges of this work. One important step was the use of generalization techniques to represent value functions and/or policies, that allowed the application of RL to continuous state and action spaces. Many different techniques that come from the machine learning field can be used, such as clustering methods, memory-based coarse representations, fuzzy representations, randomized trees or kernel functions. They have demonstrated their usefulness in RL problems (García *et al.*, 2010), opening up the possibility of their application to real world problems. Another powerful RL tool is transfer learning. This field has been experiencing significant growth in recent years and focuses on defining methods that retain and reuse previously learned knowledge. Knowledge transfer would greatly accelerate the learning process if done successfully. Different techniques for transfer learning are used in this work.

Other RL areas could become sources of new ideas for solving different problems, although they do not fall into the scope of the present work. For example, the multiple objective problem is an interesting issue for the domain of this thesis. In the presence of multiple objectives, the usual RL approach is to consider many

single-objective control problems and, then, try to combine them. Recently, new proposals appeared that enable one to learn the control policies for all the linear combinations of preferences in a single learning process (Castelletti *et al.*, 2012). Another examples of useful RL techniques in this context are discussed in Future Work section.

4.2 Methodological positioning

Prior to the presentation of the objectives, a number of considerations that lead to a methodological positioning are introduced:

- The problem of defining the general dynamics of pedestrians is complex. The state-of-the-art in pedestrian automatic recognition and tracking technology does not yet permit the proper and speedy tracking of a crowd of unaware pedestrians in cluttered environments (Berrou *et al.*, 2007). A complete description of their highly developed and complicated motion sequence is rather difficult (Schadschneider *et al.*, 2008). Therefore, a general model that describes real pedestrian dynamics is not yet available. Thus, many works have focused on specific aspects and/or scenarios such as jamming or lane formation in normal or panic conditions.

- Real pedestrians organize inside the groups interacting with the neighborhood (avoiding collisions, adapting speed and direction to the local flow, etc.). In terms of software architecture, interactions between agents is the defining attribute of Multi-agent systems.

- In realistic problems, *ad hoc* interactions can not rely on assumptions regarding norms and protocols to reach coordination. In real pedestrian problems, individuals do not communicate their intentions to their neighbors as in vehicle traffic. Out of general conventions, such as the assumption of soft changes in the motion states of the individuals in normal conditions, prior knowledge about the intentions of the other pedestrians is scarce, and their communication unlikely. Therefore, the information that agents know about the others is only provided by the sensors.

Given the above considerations, in this work the problem is addressed with the following methodological assumptions:

- No model of the dynamics of pedestrians is available. The agents learn autonomously from scratch, using model-free RL algorithms.
- Fidelity in the reproduction of the local interactions among the agents is important. The physical interactions should be calibrated to approach the real world. Moreover, the agents' actions should also be calibrated to represent the modifications of the dynamic state that real pedestrians perform. These similarities with the real world come from the fact that the physics module, which manages the local interactions, is calibrated.
- There is no prior information about norms coded into the agents. The agents learn their individual behaviors without prior coordination [1].

4.3 Objectives

The main objective of this thesis is *to create autonomous embodied virtual agents capable of learning behaviors that produce plausible simulations of pedestrians navigation using a Multi-agent RL framework.*

This objective has the following considerations and restrictions:

- Pedestrian navigation is considered in this work as the autonomous movement of a pedestrian, alone or inside a group, modeled as a particle. Therefore, the kinematics problems associated with human locomotion (considered as the movement of a body with articulated elements) are not considered.
- A Multi-agent framework implies a microscopic approach to the problem. It means that the behavioral rules describing the dynamics and the flow representation are based on individual entities.

[1] To consider that sharing the same learning method or the same features for the state space description is a kind of coordination, should be studied in future research.

- The generation of plausible behaviors means the creation of visually realistic navigation situations, analogous to those observed in real human pedestrians, although the data (position and velocities) can be different to real pedestrians. The result must be a simulation that produces a believable visual result. Plausibility is not a minor goal in pedestrian simulation. Many other works are focused on creating realistic pedestrian behaviors such as (Bonneaud & Warren, 2012; Ennis *et al.*, 2008; Lemercier *et al.*, 2012; Levine & Popović, 2012; Peters & Ennis, 2009; Sakuma *et al.*, 2005). However, it is important to know how far (or near) are the results from the reality or from other well-known models. Therefore, comparisons with experiments with real pedestrians and other models using fundamental diagrams will be carried out.

- The study is focused on planar facilities. This means that facilities like stairs, ramps or elevators are not considered.

- Pedestrian behavior is only influenced by external factors (other agents or obstacles in the environment). Internal or personal factors (such as time-pressure or psychological aspects) are not considered.

In order to achieve this objective, several milestones are proposed:

1. *Design, implementation, calibration and validation of a Multi-agent learning framework.*

   This milestone confronts different challenges:

   - Model the pedestrian simulation problem as an MDP. This requires solving several representation problems: the feature selection for representing the continuous state space, the action definition and the value function representation in a continuous state space.
   - The multi-agent architecture also presents difficulties. A multi-agent learning environment is essentially non-stationary. This is a characteristic that must be addressed in the different problems.

- The design and implementation of the framework. The following characteristics indicated by Zhou *et al.* (2010) for crowd simulation systems, are desirable:
  - Flexibility: the ability to adapt to different situations. The framework should be adapted to the representation of different scenarios and, therefore, the possibility of configuring the MDP. This implies the possibility of modifying the agent's perception of the environment (that is, different representations of the state space) and an adaptable virtual environment.
  - Extensibility: it should accommodate new features without much difficulty. To favor this, a modular object-oriented software architecture will be defined. This design allows the learning algorithms and the modes of generalizing the state space to be easily exchanged, which can then be combined to perform comparative studies. Other modules , such as the physics engine, should also be easy to change.
  - Execution Efficiency: the system will work in two different modes. In learning mode, the calculus of the learned behaviors are performed. In simulation mode, the exploitation of the learned behaviors is carried out by looking up the learned value function which constitutes an efficient decision-making system compared with other ABS. Moreover, parallel programming techniques will be used in order to exploit the parallel architectures of current workstations.
  - Scalability: this is the capability to increase the size of the problem without losing significant performance. Here, studies of scalability in the number of agents will be performed in order to test the robustness of the learned behaviors.
- The calibration and validation of the system is another challenge. First, the physics engine devoted to simulate agent interactions will be calibrated with selected and justified real pedestrian values. Then, validation experiments will be carried out and the results compared

with similar experiments performed with real pedestrians.

2. *Two different learning strategies will be evaluated: one based on the Q-learning algorithm and another based on Sarsa(λ).* Specifically, the Q-learning based strategies will constitute a new proposal in RL to adapt the VQQL algorithm to necessary requirements for pedestrian simulation.

3. *Analysis of the adequacy of the dynamics generated by the learned behaviors to pedestrian dynamics.* This study will be conducted using specific tools of pedestrian dynamics analysis and also comparing with other well-known pedestrian model (Helbing's model).

4. *Study of the capability of the framework to: i) generate emergent collective behaviors and ii) to obtain behaviors capable of operating at a higher level (tactical level).* Several scenarios will be proposed in which the existence of these capabilities is necessary to solve the navigational problem. These scenarios are well-known examples in the field of the pedestrian modeling and simulation.

To summarize, the main contribution of this work is to propose a new multi-agent reinforcement learning approach to the problem of pedestrian simulation and to empirically demonstrate that this approach provides positive results in a set of paradigmatic pedestrian scenarios.

Chapter 5

System architecture, calibration and validation experiments

In this chapter I will present an overview of the system architecture, the calibration of the physics simulator and the experiments carried out to validate the implemented framework. In addition, a section is dedicated to explaining the main tools used to analyze the experimental results.

A web site has been designed to display the results obtained in the different experiments described in this and the following chapters. The reader can access it via `http://www.uv.es/agentes/RL/index.htm`.

5.1 System Architecture

A first question arises when assuming the main objective of this thesis: is it necessary to implement a new framework from scratch? In general, the RL tools and algorithm implementations available for students and researchers (described in Section 3.7) are mainly single-agent oriented and/or educational oriented. They are built with the main goal of presenting a simple and fast way of obtaining an RL application to test new ideas or for academic purposes, and they are not prepared for large implementations of complex systems such as the problem proposed in this dissertation. Besides, the desired framework must be capable of selecting the learning algorithm and the generalization module in order to experiment with

different configurations. For these reasons, the decision to build a multi-agent learning framework from scratch is taken.

The architecture presented in this work assumes a continuous environment. In a preliminary study, a system with a *discrete grid-like environment* was developed and used in two different scenarios (specifically, the scenarios used in the experiments described in the next chapter). The first scenario (closed room with an exit) gave good results, as was reported in Martinez-Gil *et al.* (2010), and demonstrated empirically the possibilities of that multi-agent learning discrete framework. However, the results in the second scenario (a narrow corridor where a crossing of two groups of agents took place), revealed that a discrete environment reduced the navigational problem to a gap allocation problem where the agents could not develop adequate navigation strategies to solve complex spatial-organization problems. After these preliminary experiments, a real-valued sensing and navigation system was developed. The rest of the chapter will explain the architecture of the proposed system and the calibration and validation tests carried out in order to prepare the framework for further experiments that will be introduced in the next chapters.

5.1.1 Framework overview

The Multi-agent framework has two kinds of agents: the *learning agents* who are embodied agents, and the *environment* agent, without physical representation. While the number of learning agents is defined by the experiments, there is only one environment agent. The framework has two working modes: the learning mode and the simulation mode. In the learning mode, a learning agent uses RL to learn a near-optimal value function $Q(s, a)$, able to control at each moment the velocity of the physical representation of the virtual pedestrian. It constitutes the core of the agent's decision-making module. In the simulation mode, the learning agents follow the near-optimal policy $\pi(s)$ derived from $Q(s, a)$ using Equation 3.4. The environment agent works in the same way in both modes. It is in charge of the 3D virtual environment, where each learning agent is represented by an embodied virtual pedestrian.

The elements of the MDP that constitute the foundation of the learning algo-

rithms are spread in the different modules defined in the system's architecture. Thus, the transformation of the sensed signal to the state space will be allocated in one of the learning agent's modules. The action space is defined in both, the learning agents and the environment agent and the reward function is implemented inside an environment module. The transition function is unknown and depends on the physics module and the evolution of the learning agents' behaviors.

The dynamics of the framework is time-step based. The conceptual architecture of Weyns *et al.* (2004) is used with a cycle perception-interpretation-decision-actuation. At each time slot, t, all the learning agents interact with the environment following these steps:

1. Step 1: Each learning agent [1] receives individual raw data from the environment that describe his current state, s_t, and a reward, r_{t-1}, that evaluates the previous decision a_{t-1} at step $t-1$. The reward value will be zero if the environment does not have information to judge the adequacy of the action.

2. Step 2: Each agent converts the received raw data into a generalized state space s_t.

3. Step 3: Each agent selects an action a_t to be carried out. This is known as a *decision*. In the learning mode, the state s_t, the reward r_{t-1} and the previous action a_{t-1} are used by the learning algorithm.

4. Step 4: The environment gets the actions of the agents and executes them. The new actions modify the dynamics of the embodied virtual pedestrians and, therefore, the general dynamics of the environment. Then, the scene is simulated with the new dynamics for the rest of the time slot.

In both modes (learning and simulation), the execution time is divided into *episodes*. They have a maximum number of time slots (or decisions), but an agent can finish an episode without exhausting the number of decisions (e.g. because

[1] For economy I will refer to the learning agent as 'agent' and sometimes 'pedestrian'. The environment agent will be referred as the 'environment'. In an abuse of notation, sometimes the environment is only the physical vicinity of an agent (or pedestrian).

the agent has reached the goal). The duration of the execution time is defined by fixing a specific number of episodes.

In simulation mode, the environment generates an output file (divided into frames) with temporal information about positions and velocities of the embodied virtual pedestrians, which constitutes the input for the graphics engine and the analysis tools of the pedestrians dynamics.

5.2 Framework description

In Figure 5.1, a functional diagram of the two classes of agents is displayed. The modules have been enumerated with labels (M_i) to be more easily identified.

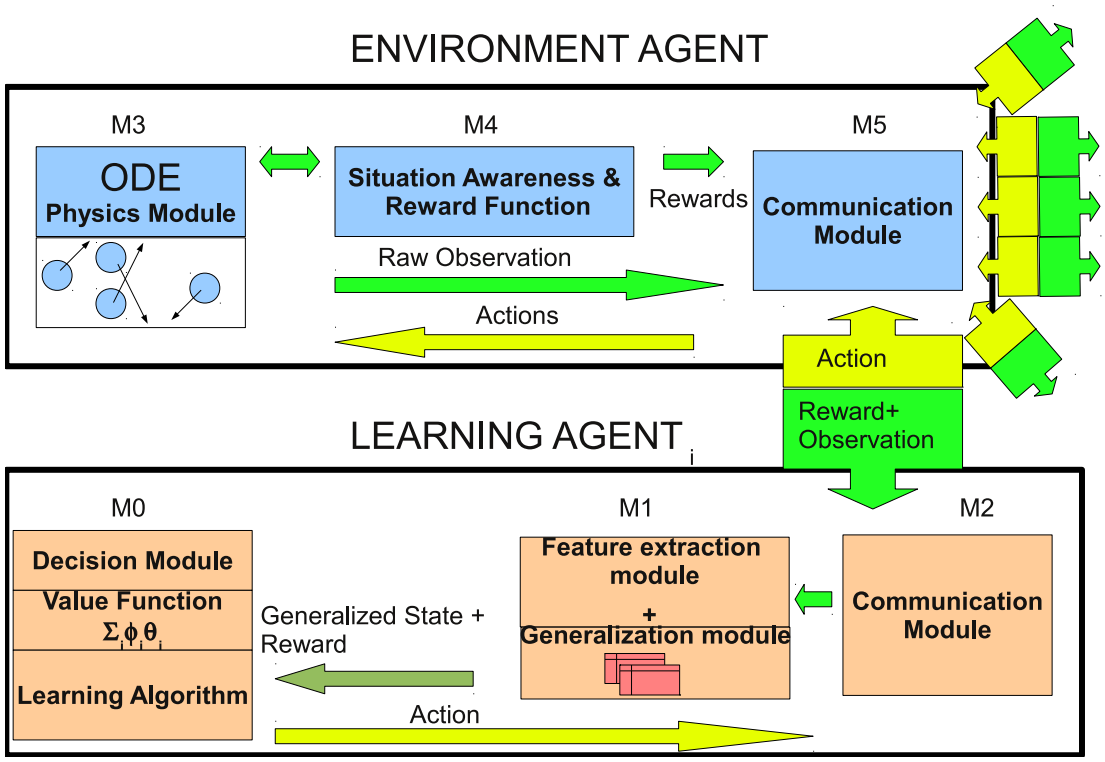


Figure 5.1: Functional diagram of the two classes of agents defined in the framework

5.2.1 Description of learning agent's modules

There are two abstract tasks in an agent: the calculation of the generalized state space and the decision-making process. When the learning mode is active, the decision-making task improves from scratch using the active RL algorithm. When the simulation mode is active, the decision-making task consists of using the learned near-optimal policy π. The cost of calculating $\pi(s)$ using Equation 3.4 is constant, thus, it is an efficient decision-making module appropriate for real-time simulations or interactive environments.

5.2.1.1 Feature extraction and generalization modules (M_1). State space definition

Each agent receives raw information from the environment sensed by the assigned embodied virtual pedestrian. This information is transformed into real features that describe the state of the agent.

The state space for each agent is modeled with the features shown in Figure 5.2. The states follow a deictic representation. The central premise underlying a deictic representation is that the agent only registers information about objects that are relevant to the task at hand (Agre & Chapman, 1987; Whitehead & Ballard, 1991). The selection of features that represent the state for the agent is critical for successful learning. Although several automatic feature selection methods have been developed for the MDPs (Kroon & Whiteson, 2009; Nguyen *et al.*, 2013), there are previous works in local pedestrian navigation that provide feature sets that have demonstrated their utility in navigational problems. The work of Lane *et al.* (2007) proposes a deictic representation of the state space, based on local information, that is particularly suitable for RL navigational tasks. In this thesis, a selection of features that provide local information about the agent's kinematic state, the neighboring agents, and the nearest walls is proposed. This features set models the observation of a real pedestrian inside a group. Similar features have been used previously in pedestrian models and they are considered as relevant for the kinematic description of the pedestrian (Robin *et al.*, 2009) or to characterize the imminence of the collision (Bierlaire & Robin, 2009). The work by Ondrej *et al.* (2010) also uses relative positions and orienta-

tions of neighbor walkers to calculate the conditions that determine a collision.

In the proposed RL framework, the state space representation is fixed but the number of features that describes the state space is configurable. In each specific experiment, a subset of these features is selected. For instance, in an environment without walls, the features related with obstacles sensing are disabled.

The features that represent angular magnitudes are measured relative to the direction that joins the agent with the goal. Note that the features that describe the neighbor agents and objects are relative with respect to the agent. Avoiding absolute measures is advisable to make the description of the states independent of the specific scenario.

The features that describe the state are real valued, therefore, a generalization process is needed to represent a usable value function. The framework allows the generalization module of its agents to be selected. The two types of generalization methods explained in sections 3.3.1 (vector quantization) and 3.3.2 (tile coding) have been implemented.

5.2.1.2 The learning module (M_0)

The RL algorithms implemented are Q-learning (described in Section 3.1.3 and referenced as Algorithm 1), and the linear gradient-descent Sarsa(λ) with eligibility traces (described in Section 3.1.4 and Section 3.1.5 and referenced as Algorithm 7). The modular architecture of the framework allows the selection of the techniques used in both the generalization and the learning modules in different experiments. Additionally, the learning algorithms include knowledge transfer techniques that will be explained in the specific experiments in the following chapters.

Furthermore, the user can define the exploratory policy for each experiment. For the experiments described in this thesis I have used a ϵ-greedy policy with an exponential decreasing value respect to the number of episodes carried out by the agent (see Section 3.1.3 for a detailed explanation). Although a Softmax-type exploratory policy has also been tested, it is difficult to tune for a high number of actions. The results, possibly due to this fact, have shown a poor performance. On the contrary, the ϵ-greedy exploratory policies are easy to define and have

Sag	Speed of the agent.
Av	Angle of the velocity vector relative to the reference line.
$Dgoal$	Distance to the goal.
$Srel_i$	Relative speed of the i-th nearest neighbor.
Dag_i	Distance to the i-th nearest neighbor.
Aag_i	Angle of the position of the i-th nearest neighbor relative to the reference line.
Dob_j	Distance to the j-th nearest static object (walls).
Aob_j	Angle of the position of the j-th nearest static object relative to the reference line.

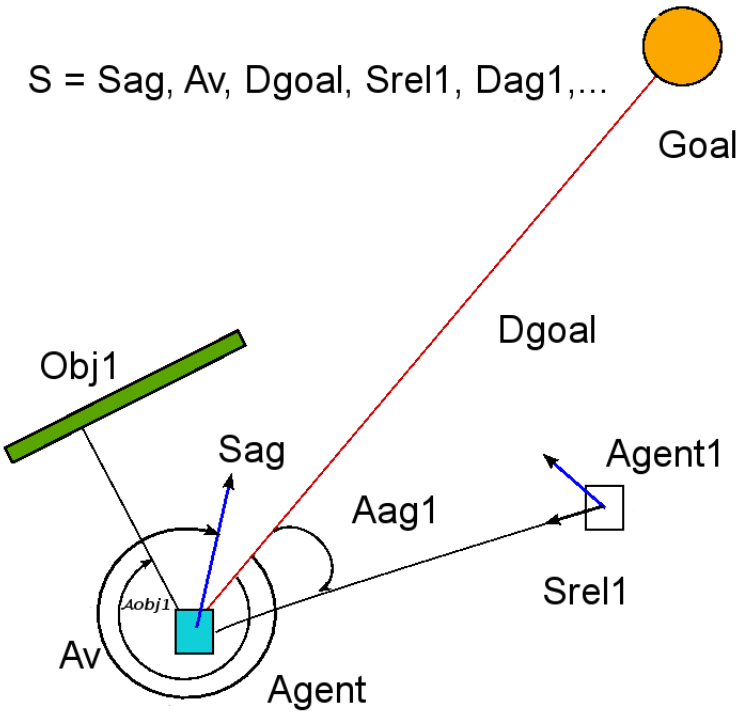


Figure 5.2: State space attributes. The reference line joins the agent with its goal.

provided good results in all the tested scenarios.

5.2.2 Environment agent's modules

The environment agent has several roles in the framework, such as: sensing the parameters that define the raw state of each learning agent, processing the action instructions coming from the agents, criticizing the results of these actions (through the reward signal), and managing the physics module.

5.2.2.1 Situation awareness and reward function module (M_4)

The function of this module is double. First, it senses the virtual environment from the point of view of each embodied virtual pedestrian to collect the parameters that describe the raw state of the corresponding agent. Each embodied virtual pedestrian is attached in the learning working mode, to a specific learning agent. The sensed raw state is processed by the feature extraction module of the correspondent learning agent as explained in subsection 5.2.1.1. Second, after observing the physical consequences of the actions carried out by the agents at time $t+1$, it makes a judgment of the suitability of the action selected at time t for each agent. The immediate rewards provided at each time slot by the rewards module are used in the calculation of the return (see Equation 3.1). The reward function is designed by the user and defines the behavior of the agent. It is not necessary to model all the possible cases. When there is no information about the adequacy of an action taken in a specific state, a value of 0 is returned as the immediate reward.

5.2.2.2 Physics module (M_3)

The physics module is a calibrated version of the Open Dynamic Engine (ODE), which is a physics software library implemented by Russell L. Smith (Drumwright *et al.*, 2010; Smith, 2001). ODE uses a first order semi-implicit Euler integrator which uses, to calculate position q and velocity v, the following equations:

$$q(t+1) = q(t) + hv(t+1)$$
$$v(t+1) = v(t) + ah \tag{5.1}$$

where h is the time step of the integrator and a the constant acceleration in a timestep. The difference with respect to other integrators is that position q is a function of the velocity at the end of the time step $v(t+1)$. This type of integrator is also used in other physics engines such as Bullet Physics Library or Box2D because it is stable and fast.

The physics module also implements the execution of the agents' actions. An action can be understood as a behavioral force that the agent applies to itself to modify its velocity vector just as real pedestrians do. The variation of this velocity vector has been used to control the trajectories in other pedestrian models (Bierlaire & Robin, 2009) and it is carried out with two types of actions that actuate simultaneously. One type varies the speed; the other varies the direction. The agent has to choose a pair of actions (one of each type) in each decision of an episode.

5.2.3 The communication modules

The communication modules ($M2$ and $M5$) are the interface between each agent and the environment. The communication is bidirectional and occurs every time step. There is no communication among the learning agents. The information flow can be observed in Figure 5.1. The information is included inside a list data structure that is transmitted through the MPI (The MPI Forum, 1993) protocol (described in the next subsection). Figure 5.3 displays the order in which the information is transmitted in a time step t. First, the environment communicates the reward which corresponds to the action taken at time $t-1$ and the sensing of the reached state s_t. In learning mode, this information is used by the learning algorithm to update the value function. Then, the agent selects an action a_t and sends it back to the environment which executes the action in the physics simulator. In the simulation mode, the reward information is not necessary and it

is not calculated by the environment. In this case, a default value is transmitted, and it is ignored by the agents.

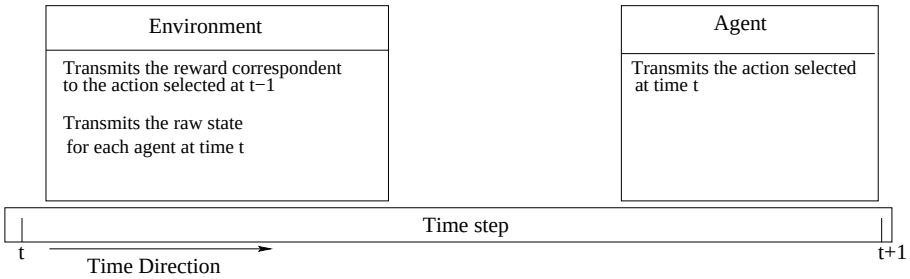


Figure 5.3: Data communication process between an agent and the environment in a time step.

5.2.4 Software implementation

Each agent has been designed as an independent computational process which follows a distributed memory model of parallel architecture that uses MPI. In the MPI programming model, a computation comprises one or more processes that communicate by calling library routines to send and receive messages to other processes. In most MPI implementations, a fixed set of processes is created at program initialization, and one process is created per processor. Processes can use point-to-point communication operations to send a message from one named process to another; these operations can be used to implement local and unstructured communications (Foster, 1995). The MPI programming paradigm considers local memory with private memory addresses for each process and communication between processes using message passing.

The output of the learning mode consists of several files with al data about the learning process. Specifically, data about mean and standard deviation of performance, immediate rewards and number of actions used in the episode. In the simulation mode, the output is currently a text file divided into frames where the position and velocity of each agent is stored. This file is the input for the 3D graphics module which reproduces the simulation. In learning mode, the 3D visualization module is not active. The coupling of the 3D module to the simulation output is immediate. However, the decision to dump the information

into a file was taken in order to have available data to analyze the dynamics of the simulated pedestrians.

The programming language is C++ with the additional modules of MPI for parallelization, the ODE physics engine library and Open GL library for the 3D visualization module. A total of 39 C++ classes were defined and implemented plus a few more files that contain GNU licensed code. These GNU files implements a random generator, a K-means algorithm and the tile coding generalization system. The reader can access the web page `http://www.uv.es/agentes/RL/code/files.html` where the files of the implemented classes are listed and the header files with the class definition are available.

5.2.5 The graphics visualizing tool

The visualizing tool takes as input the file with the per-frame positions and velocities of the agents and visualizes them with the required frame rate. This tool represents the scene with the same spatial information used by the ODE physics module, displaying an accurate representation, although not visually appealing. Two scenarios are represented with this visualization tool in Figure 5.4. The walls are represented by blue prisms and the embodied agents are represented by spheres, as discussed in Section 5.3.1. However, any tool that takes this format as input (which is common in simulation), can display the simulation. In fact, several simulations have been performed using the Unity Game Engine ® which can be seen on the web site. Figure 5.5 shows images of the same scenarios renderized with Unity.

5.3 Model of the physics and calibration

In this section I will explain the models of the different physical elements that make up the virtual environment, as well as the decisions taken in order to calibrate them. Calibrating a system means providing correct or justified values to the parameters of the system according to the problem to be solved. In our case, the parameters of the physic module have to be adjusted to represent a real pedestrian environment. The parameters discussed in this section are directly

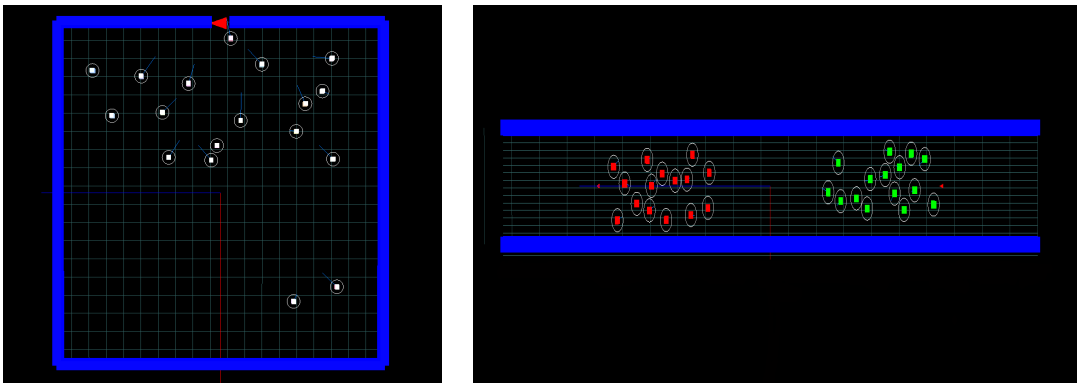


Figure 5.4: Different scenarios. Left: agents inside a room. Right: crossing of two groups inside a corridor. The agent's body is represented as a cube with a bounding circle that simulates the agent's boundary. The walls are represented in blue and the red triangles indicate the goals.

Figure 5.5: Renderized simulations of the above scenarios using Unity Game Engine ®. Left: two views of the agents inside a room scenario. Right: view of the corridor scenario.

related to internal ODE parameters making ODE calibration a straightforward process.

5.3.1 Model of the Body

The first considerations for designing magnitudes for pedestrians appear in the book by Fruin (1971b), who suggests that the plane view of the human body can be approximated as an ellipse defined by the body depth and shoulder breadth measurements. Human factors studies have shown that the fully clothed dimensions of the 95th percentile of the population (95% are less than this) are 330 mm in body depth and 580 mm in shoulder breadth. The plan view of the average male human body occupies an area of approximately 0.14 m^2. A similar body ellipse equivalent to a standing area of 0.21 m^2 has been used by the New York City Transit Authority to determine the standee capacity of its subway cars. The study by Rouphail *et al.* (1998) recommends designing a simplified body ellipse of 50 $cm \times 60$ cm for standing areas, with a total area of 0.3 m^2 (roughly the 108% of the ellipse suggested by Fruin. Figure 5.6 displays these body measurements. This study was used for the USA Federal Administration in its Highway Capacity Manual.

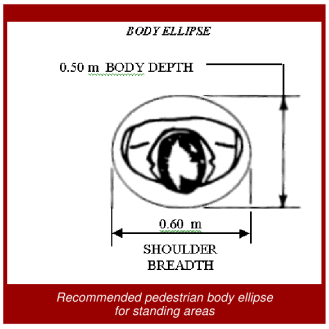


Figure 5.6: Pedestrian body model dimensions in the Rouphail's study

In this work, the pedestrians will be modeled as spheres in the ODE physics engine (with a circular plane section instead of elliptic one) to make collision detection faster and computationally cheaper than the elliptic configuration. However, the dimensions of the circular plane section are similar to those of the elliptic

model. Specifically the radius will be 0.3 m giving an area of 0.28 m^2, approximating the dimensions recommended in Rouphail's study. Other studies have also adopted the circular plane section for modeling the shape of pedestrians for similar reasons (Bourgois *et al.*, 2012; Johansson *et al.*, 2007).

5.3.2 Kinematics Model

The kinematic model is included inside self-driven many-particle systems. In these systems, the particles have an internal energy reservoir that feeds an individual driving force (Helbing, 2004; Schreckenberg & Wolf, 1998). Examples of these systems are animal herds, flocks of birds or traffic. Although these systems are within the class of non-equilibrium systems (open systems that exchanges energy and mass with the environment), the transport properties of the particles can be described with simple modifications of Newton's equation of motion (Helbing, 2004). The net force that actuates in a particle i is the sum of the external forces over it:

$$m_i\vec{a} = \sum_j \vec{F}_j \tag{5.2}$$

In self-driven particles, another internal force exists to generate the individual driving motion, therefore:

$$m_i\vec{a} = \vec{F}^i_{driv} + \sum_j \vec{F}_j \tag{5.3}$$

Two different dynamic cases exist. In the first, the particle moves freely without interacting with other objects. Then, the forces actuating over the particles are:

$$m_i\vec{a} = \vec{F}^i_{driv} - \mu_f|\vec{N}|\vec{u_v} \tag{5.4}$$

Where $\vec{u_v}$ is the unitary vector with the direction of the velocity and $|\vec{N}|$ is the module of the normal force. The second term of the equation represents the friction forces with the floor, where μ_f is the friction coefficient. It is easy to model the walk of a pedestrian in the absence of obstacles and without the

intention of changing the velocity. In this case, the dynamic situation is:

$$\vec{F}^i_{driv} = \mu_f |\vec{N}| \vec{u_v} \tag{5.5}$$

The total force of the agent is zero. That is, the agent is creating an internal force equal and contrary to the friction force to maintain a constant velocity in the walk, conceptually similar to what real pedestrians do.

In the second case, a collision happens between the agent and another object. Then the total force over an agent is:

$$m_i \vec{a} = \vec{F}^i_{driv} - \mu_f |\vec{N}| \vec{u_v} + \vec{F_c} + \vec{F_{fr}} \tag{5.6}$$

Where $\vec{F_c}$ is the collision force between the two objects and $\vec{F_{fr}}$ is the force of friction between the two objects. The solution to this second order differential movement equation is solved numerically in the ODE integration module.

The most interesting component of this kinematic model is constituted by the internal driving force $\vec{F}^i_{driv}$. This force has a dual role in the simulation. First, it is in charge of compensating the friction force with the floor as stated above. Second, it is a *behavioral force* that translates to forces the action that the agent decides to perform in each step.

$$\vec{F}^i_{driv} = m_i \frac{d\vec{u_v}}{dt} \tag{5.7}$$

Behavioral forces are commonly used in social forces models and, in these contexts, they implement different influences simultaneously affecting the behaviors of pedestrians (Helbing *et al.*, 2001). For instance, the repulsion forces to avoid obstacles or agents, the attraction forces to simulate intentions (for example to see window displays or meet other agent) and the need to reach a desired velocity.

5.3.3 Collision models and calibration

Modeling collisions between agents is an important task because many of the microscopic interactions (local interactions among agents) consist of avoiding or reacting to the collision of other agents. Of special interest is to model the

mechanical response of skin because it is the first contact element that participates in the collision of two embodied agents. Models of the human skin have varied depending of the study the authors are focused on. The work by Magnenat-Thalmann *et al.* (2002) considers fold and wrinkle formation and models the skin as layers of tissues, each one controlled by an elastic deformation using the Hooke's Law:

$$\vec{F} = -k\vec{x} \tag{5.8}$$

where k is the elasticity constant of the material and x is the elongation (distance to the equilibrium state) of the object. The work by Waters & Terzopoulos (1990) is oriented to the computer graphics representation of the face and models the skin using deformable meshes in a three layers model.

If the graphic representation is not the main goal, the mechanical response of the skin (taking in account that the skin is a composite of three kinds of materials: the epidermis, the dermis and the hypo-dermis) can be modeled as viscoelastic materials (Herman, 2007; Wijn, 1980). A viscoelastic media can be modeled using the combination of two types of elements: springs and dashpots. The basic combinations are inspired in parallel and series electrical configurations (Herman, 2007). Specifically, the *Maxwell body* is a spring and a dashpot in series. The *Voigt body* is a dashpot and a spring in parallel and the *Kelvin body* is a Maxwell body in parallel with another spring. Each model is valid for different types of materials and situations. While the Maxwell model is preferred to model the fusion state of several metallic materials, the Voigt model is more appropiate for collagen or silicon materials. The Kelvin model, and other more complex combinations of springs and dashpots, are more accurate if they are well configured but they are difficult to solve analytically and hard to compute numerically. The work of Heïgeas *et al.* (2003) local interactions between pedestrians were modeled using a mass-spring-damper system. In that work, stiffness and viscosity terms change with respect to relative distance between walkers.

For the collision model in this work, the Voigt model represented in Figure 5.7 will be selected due to the mentioned similarity of skin with the visco-elastic materials.

In the visco-elasticity model, the force associated with the spring is given by

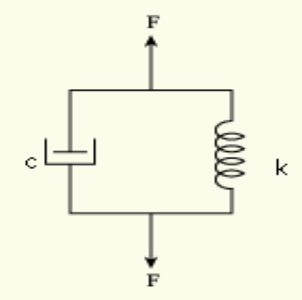


Figure 5.7: The Voigt Model for a visco-elastic media

Hook's Law (see equation 5.8) and the dashpot follows the equation

$$\vec{F} = -c\vec{v} \tag{5.9}$$

Where v is the velocity in response to the force and c is the viscosity damping constant. The total force of Voigt system is the sum of the partial forces of each element.

$$\vec{F} = c\frac{d\vec{x}}{dt} + k\vec{x} \tag{5.10}$$

In a dimension and considering the dotted variable as a derivated of time and the double dot as the second derivate of time:

$$m\ddot{x} = -kx - c\dot{x} \tag{5.11}$$

$$\ddot{x} + \frac{c}{m}\dot{x} + \frac{k}{m}x = 0 \tag{5.12}$$

Defining $w_0 = \sqrt{\frac{k}{m}}$ y $\zeta = \frac{c}{2\sqrt{mk}}$ Equation 5.12 is written as:

$$\ddot{x} + 2\zeta w_0\dot{x} + w_0^2 x = 0 \tag{5.13}$$

The initial condition is $x(t = 0) = 0$ because the dashpot prevents any immediate deformation.

Equation 5.13 is the differential equation of the motion of an harmonic oscillator and it is solved numerically by the ODE integrator module.

The parameter ζ configures three different types of damping (see Figure 5.8):

- If $0 < \zeta < 1$ the system is underdamped and it oscillates with a reducing amplitude (dampening)
- If $\zeta > 1$ the system is overdamped and it returns to the equilibrium without oscillations.
- If $\zeta = 1$ The system is critically damped and it back to the equilibrium as fast as possible without oscillations.

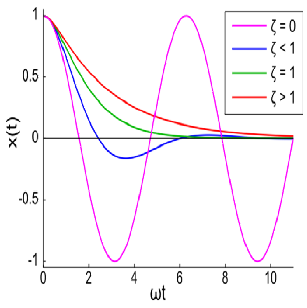


Figure 5.8: The damping ratio parameter

In this work, I will consider an overdamped system in similarity with a collision between real pedestrians. ODE can model a viscoelastic media through the configuration of two system parameters (ERP and CFM) that need the value of the coefficients k and c. Knowing only k, the coefficient c can be calculated from the expression $\zeta = \frac{c}{2\sqrt{mk}}$.

The Young module for the elasticity of the human skin varies in the literature. The book by Herman (2007) sets a value of $0.3\,MPa$ (Megapascals) for the face skin while the work by Magnenat-Thalmann *et al.* (2002) gives a value of $0.6\,MPa$ for the deep dermis. Skin thickness is variable depending on sex and age. Medical studies (Escoffier *et al.*, 1989) work with values between 0.7 and $0.8\,cm$ for women, or between 0.8 and $1.0\,cm$ for men aged between 25 and 65. The paper by Magnenat-Thalmann *et al.* (2002) sets a value of $1.3\,cm$. I will consider a value of $L_0 = 1.0\,cm$ as a compromise between the different values. The Young module is defined as:

$$Y = \frac{kL_0}{A} \tag{5.14}$$

where A is the cross-sectional area and L_0 is the length when the material does no suffer any force. Assuming a Young coefficient for the skin of $0.5\,MPa$, and a value of $A = 10\,cm^2$, we can calculate the values for k and c used in the configuration of the ODE library (See Table 5.1). The relative mass of the modeled pedestrians has been set to $50\,Kg$. The crashes between pedestrians occur in a standing position and not all the mass of the pedestrian is involved, specifically the torso is the most important part in real pedestrian collisions. The ground compensates part of the weight of the pedestrian, specifically the foot, or feet, on the ground and part of the leg. Therefore, the kinetic energy involved in the crash has to be considered with less mass than the real mass of the pedestrian.

Y	$0.5\ MPa$
A	$0.001\ m^2$
L_0	$0.01\ m$
ζ	1.5
Relative mass	$50\ Kg$
k	$50000\ Nw/m$
c	$4743\ Kg/s$

Table 5.1: Values used in the calibration of ODE physical simulator for collisions between agents

5.3.4 Friction model and calibration

The ODE library automatically calculates the friction between two bodies in contact. The contact-point-based model of friction in the ODE library is a simplification of the Coulomb model:

$$|\vec{F_t}| \leq \mu |\vec{N}| \tag{5.15}$$

Where $\vec{F_t}$ is the tangential force proportional to the force $\vec{N}$ normal to the surface where the contact point is applied and μ is the friction coefficient. The resultant tangential force is parallel to the friction surface and is in the opposite direction to the movement.

The experiments described by Kwiatkowska *et al.* (2009) using aluminum as the contact surface with the skin, provides values of $0.5 < \mu < 1.5$ in a wide range

of skin hydration conditions. Other experiments (Masen, 2011; Ramalho *et al.*, 2007) also give values in the range assigned by Kwiatkowska. I will consider a value of $\mu = 1.0$, as an approximate value for the kinetic friction coefficient of the human skin[1]. The friction against the floor will be modeled with the friction coefficient of rubber against the concrete, imitating the contact of the shoes with the floor. The chosen values are displayed in Table 5.2.

	μ
Friction coefficient for the skin	1.0
Friction coefficient of rubber against concrete	0.8

Table 5.2: Values of the kinetic friction coefficient used in the model of friction forces

5.3.5 Actions model and calibration

The agents' actions modify its velocity vector to control the trajectories in pedestrian models (Bierlaire & Robin, 2009) and mimics the control that real pedestrians operate in their trajectory. The actions are taken in pairs. The first component of the pair modifies the velocity module (increasing or reducing it) and the second, the orientation of the velocity vector (clockwise or counterclockwise). There are 8 different ratios plus the 'no operation' option for both the speed and the angle, resulting in 81 possible combined actions. In this model, each action is understood as the configuration of the physic impulse $\vec{I}$ of the behavioral force from time t_1 to t_2.

$$\vec{I} = \int_{t_1}^{t2} \vec{F_{driv}} dt = \int_{t1}^{t2} (d\vec{P}/dt)dt = m_i(\vec{v_{t2}} - \vec{v_{t1}}) \tag{5.16}$$

In the case of the velocity module (speed), adding or subtracting an absolute quantity is the recommended method for simulating real pedestrians' speed variations Bierlaire & Robin (2009). The possible additions or subtractions to the current velocity are the fractions 1/8, 1/4, 1/2, 1/1 of a reference value a_{ref}

[1]Other situations can be modeled (i.e. considering the friction between two clothes) with adequate values of the parameter μ

that has been set to $1.75\,m/s$ following the work by Teknomo (2002). In that work, this value was obtained from video tracking sequences of real pedestrians on a walkway. With respect to the angle of the velocity, the work by Steiner *et al.* (2007), that describes the calibration of the SimWalk pedestrian simulation system, states that in normal conditions (free or congested) pedestrians walk in a smooth trajectory. Therefore, the angle of the velocity vectors between subsequent steps is assumed to be small. The authors point out that the maximum change for walking direction in the SimWalk pedestrian simulator is of $\pi/4$ radians. Following that study, I also assign a maximum value of $\pi/4$ radians to the inter-step direction of velocity variation. Therefore, the possible actions are the addition or subtractions of the fractions $1/8,\ 1/4,\ 1/2,\ 1/1$ of the reference value $\pi/4$.

The agent cannot be assigned with a negative speed. When the deceleration produces this situation, the final speed is always $0\,m/s$. Besides, the speed cannot increase more than the maximum allowed ($1.8\,m/s$), determined empirically in Teknomo (2002). Therefore, the actions over the speed can produce a positive and negative value overflow (Figure 5.9) that is managed by setting the velocity to the end value.

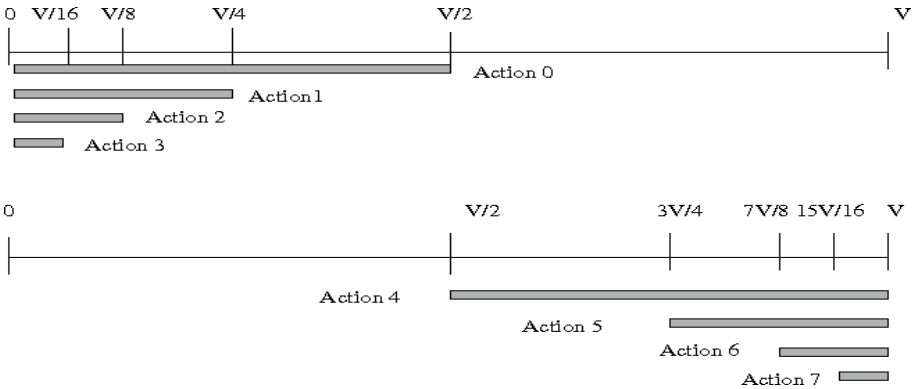


Figure 5.9: The range of operation for the actions that modify the speed with a reference value fixed to half of the maximum speed V. The gray rectangles show the range of the speed where the action has the same effect (overflow). Above: operation range for the actions that decelerate. Below: operation range for the actions that accelerate.

The kinematic module of the environment moves the agents across the plane

using the velocity vector of each agent. The simulation is discretized in Simulation Time Slots (STS) of the same size assigned to the agent's decisions. During one STS, the agent's velocity remains constant unless a crash occurs, because the driving force $\vec{F_{driv}}$ compensates the friction force against the floor. The STS is the inverse of the number of decisions per second that the agent must take, and it is a configurable parameter inside this framework. The reaction time is the time that a pedestrian takes to change his/her kinematic condition when an event occurs (for example to avoid another pedestrian that suddenly appears in its field of vision). I will configure the STS parameter (where the agent executes a decision) taking into account the real pedestrian range of response. The work by Hoogerndoorn & Daamen (2009) states that the reaction time range of a real pedestrian varies from 0.1 seconds to 0.8 seconds. We set the STS parameter to 0.5 seconds (therefore, the agent takes two decisions per second).

The values for calibrating the actions are summarized in Table 5.3.

Maximum speed	$1.8\,m/s$
Maximum change of direction	$\pi/4\,rad$
Maximum acceleration module (a_{ref})	$1.75\,m/s^2$
Reaction time (STS parameter)	0.5 seconds

Table 5.3: Values used in the calibration of actions

5.3.6 Calibration of the time step for integration

The accuracy of the Euler semi-implicit integration procedure depends critically on the selected integration time step. Ideally, it should be near 0 to provide an accurate algebraic-like calculation. However, it is not efficient because the integration step is a bottleneck for the simulation process. Although paralellization can alleviate the problem, in a scalable framework, the problem does not disappear. Therefore, a trade off between accuracy and efficiency is necessary.

In order to decide the range of operation of the integration time step, I have set a simple test where two agents suffer a frontal crash with constant velocity $1.8\,m/s$ (see Figure 5.10).

Figure 5.11 shows the mean values of the speed calculated by ODE after the

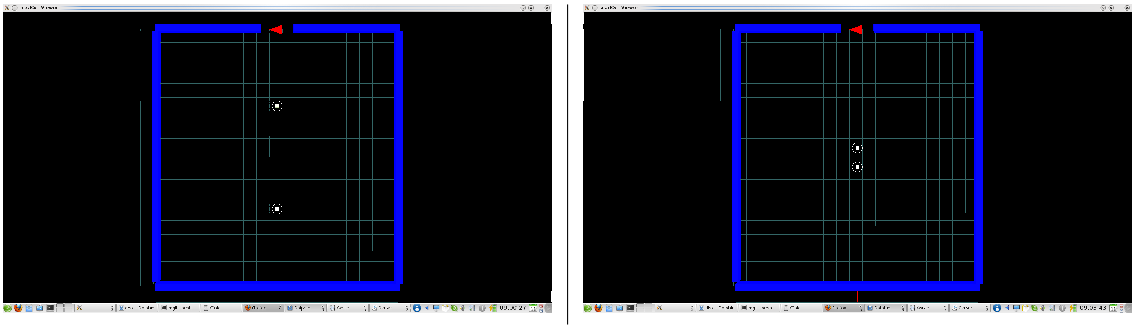


Figure 5.10: Collision scenario. Left: start positions. Right: positions after the collision.

collision (named the bounce speed in the graphics). The logarithmic scale in the abscissa shows the number of timesteps per second. The means have been calculated in an interval centered in a power of 10 of the same relative width, that is, values of 7, 8, 9, 11, 12 around the ten, values of 70, 80, 90, 110, 120 around the hundred, etc. The values tend to converge to the value of abscissa 10000. There is a significant difference in the standard deviation of values around ten and the rest of the values. However, the mean values are similar for 100, 1000 and 10000 time-steps per second. I will consider a time-step of order 2 (about 100 time-steps per second), a good trade off between accuracy and efficiency.

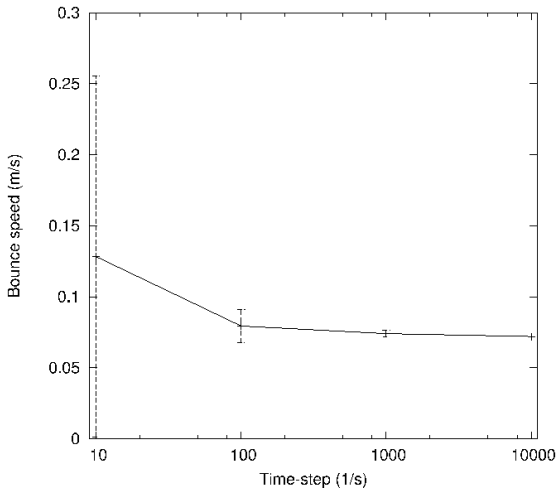


Figure 5.11: Averaged values of bouncing speed after frontal collision between two agents

5.4 Analysis tools

In this section, I will introduce the tools used to analyze the results of the framework's main processes: the learning process and the simulation. These tools will be used, among others, to indicate the level of quality achieved in all the experiments described in this work.

5.4.1 Tools for analyzing the learning processes

The learning process is divided into episodes. In an episode, each agent tries to achieve their goal (specifically reaching a point by means of his/her velocity control) by taking decisions. Each decision is evaluated by the environment, and this evaluation is used to learn another better control. Therefore, the learning process is gradual and incremental, being developed along a number of episodes (thousands of episodes in the experiments considered in this work). The natural way of representing the evolution of the learning process in an agent is to represent a performance curve. It displays the performance reached by the agent along the episodes. If the learning process works, the performance increments its value with the number of episodes. The typical shape of the curve increases at the beginning and is asymptotic when the agent has learned enough. The concept of 'performance' can be specific in each experiment. In general, it indicates the number or percentage of successful episodes, whatever this means in each experiment. Other indicators, which are equivalent to the performance, are those that represent the length of the episode (in number of decisions) and the mean of the rewards obtained in an episode.

In Figure 5.12, different types of curves that describe a learning process are displayed. On the top left, a graphic with three curves represents the normalized percentage of performance obtained. The curve entitled "Mean of performance in the last 100 episodes" calculates the mean of the successful episodes in the last 100. Note that the curve oscillates with a frequency caused by the exploratory policy. Although the exploratory process decreases exponentially, exploration is active in all the episodes because it is part of the learning process. The curve entitled "Mean of performance from the beginning" represents the same data as the previous curve, but the mean is calculated using all the data from the beginning. Thus, the

final performance is lower than the other curve. The curve entitled "Mean curve of performance from the beginning for all agents" represents the mean curve of the mean performance from the beginning *for all the agents involved in the learning process*. In this curve, the standard deviation is displayed in several points of the curve indicating the variability of the different agents' learning processes. It is higher at the beginning because the learning curve oscillates notably at the early stages and reduces at the end. The shape of the curves is typical in a learning process. In a first stage, exploration prevails over exploitation and learning improves fast. In a second stage, the curve is increasing at a very slow pace which indicates that the agent has learned. In the case that the learning process fails, the shape of the curve would decrease from a point.

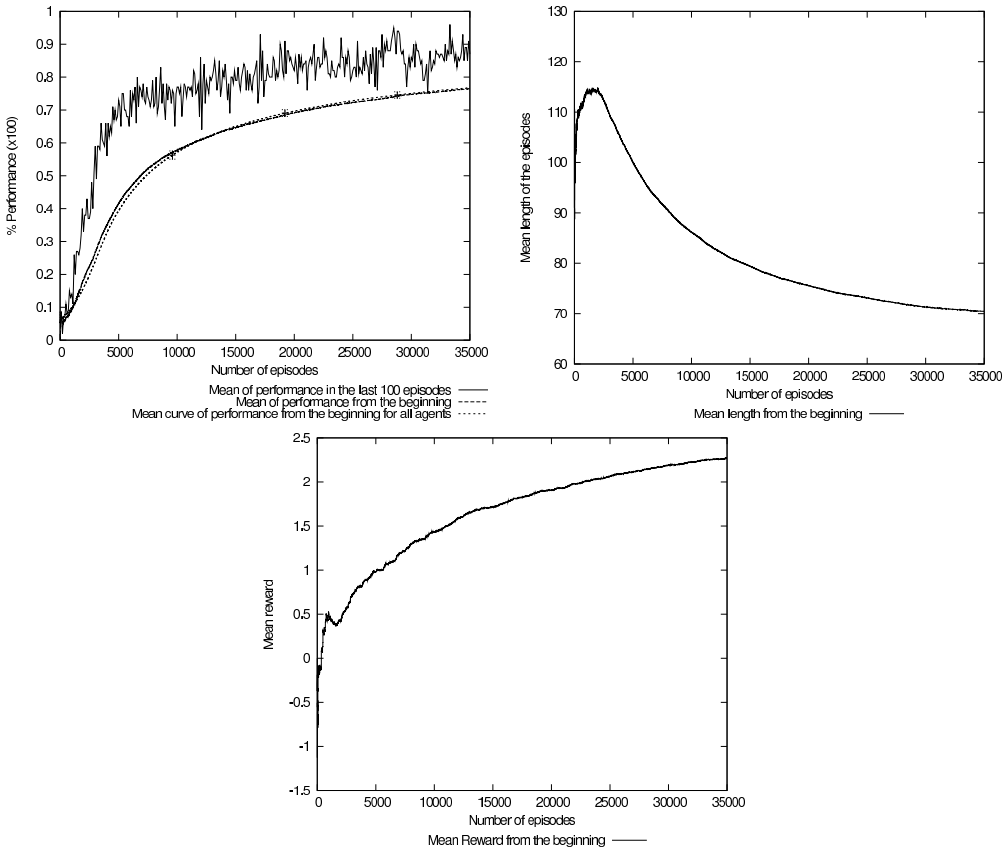


Figure 5.12: Different ways of monitoring the evolution of a learning process. Top and left: mean performance. Top and right: mean length of the episodes. Down: mean reward of episodes.

The curve on the top right of Figure 5.12 represents the mean length of the episodes from the beginning of the learning process for an agent. As the learning process progresses, the curve decrease because the policy improves directing the agent to the goal. The curve at the bottom of Figure 5.12 represents the mean of the return achieved from the beginning in the successive episodes for an agent:

$$\bar{R}_N = \frac{1}{N}\sum_{i=1}^{N}\gamma^{l_i} r_{l_i} \tag{5.17}$$

where N is the number of the episodes considered, γ is the discount factor, l_i is the length of the episode i and r_{l_i} is the reward at the final state of the episode i. This curve increases because the episodes are shorter and the probability of being successful increases with the number of episodes.

5.4.2 Tools for analyzing the pedestrian dynamics. The fundamental diagram and the density map.

The fundamental diagram has already been introduced in Section 2.2.1.2. It is a tool that uses the relation between two macroscopic quantitative parameters to validate and calibrate the pedestrian models: the flow J and density ρ (Schadschneider & Seyfried, 2009b). Due to the hydrodynamic relation $J = \rho v b$, where v is the average velocity, there are three equivalent forms: $J(\rho), v(\rho), v(J)$. In this work, the second one will be used because the velocity is primary data obtained from the experiments.

The definition of density in this field is pedestrians per area unit. In a certain area A, usually a rectangle, the number of people inside it, N, is counted. The density is then

$$D = \frac{N}{A} \tag{5.18}$$

This classic definition of density is appropiate for fluids with $> 10^{18}$ particles per mm^3. However, pedestrian streams are not infinitely many particle systems and the individuals are not adimensional points but they are discrete and extended in space. The calculus of density has problems of definition especially at the borders of the measured space. There are two types of problems

with the measurement of density caused by the borders: spatial and temporal. The spatial problem consists of defining when a pedestrian is inside the measured surface. Counting or not the presence of a pedestrian who has half of their body outside the measurement area can significantly alter the density value. The temporal problem is related with the fact that pedestrians near the border can oscillate between being inside and outside the measurement area, generating great deviations of the mean density values. Other definitions of local density have been proposed to manage with finite particles. A way to define values of density in a point is to consider that every individual i produces a density distribution $p_i(\vec{x}) \mid \int p_i(\vec{x})d\vec{x} = 1$. Examples of these kinds of functions are the step function $p_i(\vec{x}) = \frac{1}{2\pi r^2}$ for $\|\vec{x} - \vec{x_i}\| < r$, a linear function of distance $p_i(\vec{x}) = \max(0, h(r - \|\vec{x} - \vec{x_i}\|))$, or a gaussian. Recently, the work by Steffen & Seyfried (2010) proposes assigning a personal space to every pedestrian using a Voronoi diagram. This last approach minimizes the density scatter and allows a resolution down to individual level. In this work, I will follow the definitions given in Helbing's work (Helbing *et al.*, 2007). Previous studies of this author carried out in the real scenario of the Jamaraat Bridge (Mina, Saudi Arabia), where a massive ritual happens as part of the pilgrimage to Mecca, showed that the densities calculated with the following definitions agree with the actual average density. The local density is obtained by averaging over a circular region of radius R. The local density at place $\vec{r} = (x, y)$ and time t was measured as:

$$\rho(\vec{r}, t) = \sum_j f(\vec{r_j}(t) - \vec{r}) \tag{5.19}$$

where $\vec{r_j}(t)$ are the positions of the pedestrians j in the surrounding of $\vec{r}$ and

$$f(\vec{r_j}(t) - \vec{r}) = \frac{1}{\pi R^2} exp[-||\vec{r_j} - \vec{r}||^2/R^2] \tag{5.20}$$

is a gaussian, distance-dependent weight function.

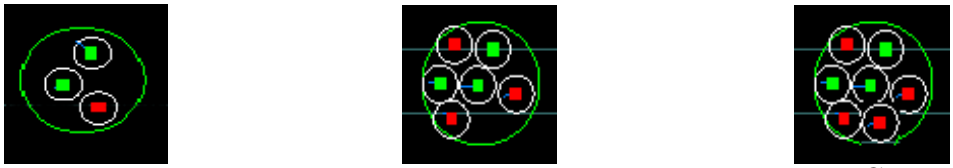
The local speeds have been defined via the weighted average:

$$\vec{S}(\vec{r}, t) = \frac{\sum_j \vec{v_j} f(\vec{r_j}(t) - \vec{r})}{\sum_j f(\vec{r_j}(t) - \vec{r})} \tag{5.21}$$

while the flow has been determined according to the fluid-dynamic formula

$$\vec{Q}(\vec{r},t) = \rho(\vec{r},t)\vec{S}(\vec{r},t) \tag{5.22}$$

The capacity of a surface to allocate pedestrians, and therefore the maximum density, depends, among other factors, on the shapes and sizes of the pedestrians. In similar circumstances, groups and crowds made up of pedestrians with different morphology (such as groups formed by people of different sexes and different ages) generate higher densities due to the best fitting of the body volumes. The body model of the agents in this work is a sphere of the same radius for all the agents (see Section 5.3.1). This fact generates densities reached by the system which are lower than in real pedestrian studies. Figure 5.13 represents different density situations generated with the agent shapes used in this work (using Equations 5.19 and 5.21). It is difficult to find densities with values higher than 2.2, although it is possible because the effective radius is greater than the geometric radius.



A B C

Figure 5.13: Groups of embodied agents with different densities inside a measurement area of radius 1 m. A: density = 0.95 Ped/m^2 B: density = 1.9 Ped/m^2 C: density = 2.2 Ped/m^2

A density map indicates the spatial distribution of the agents during the experiment. It is a histogram that shows the occupancy of the physical space. The space is divided into tiles and the number of times that the tile is occupied along time is counted. The density map reveals the patterns of movement present in the navigational problems (paths, cloggings, etc.) showing the zones frequently occupied by the pedestrians.

5.5 Validation Experiments

Two experiments have been designed to validate the proposed framework with existing empirical studies in real pedestrian dynamics. The first is a 1-Dimensional

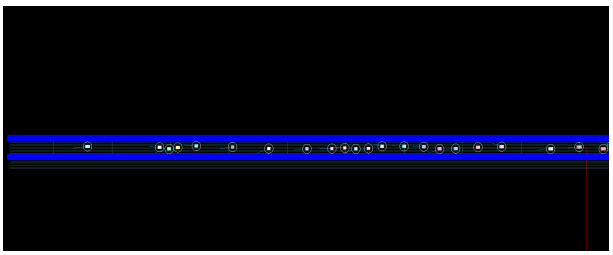
pedestrian motion that is compared with a similar experiment described by Seyfried (Seyfried *et al.*, 2007). The second is a 2-Dimensional open field scenario that is compared with two studies Weidmann (1993) and Mori & Tsukaguchi (1987). The fundamental diagram was used as the primary analysis tool to check whether the calibrated framework generates results comparable with those of real pedestrian dynamics. These experiments have been selected because they represent the simplest collective scenarios. However, the validation process can be extended to the rest of the experiments explained in the following chapters, where evidence of the similarities with the dynamics of other pedestrian models are presented.

5.5.1 Line walking

The work by Seyfried *et al.* (2007) uses sets of real pedestrians (students and administration staff of the Jûlich Research Centre, Germany) placed inside a closed oval circuit with a total length of 17.3 m and being 0.8 m wide. There is a measure area 2 m long, placed in a straight corridor of the circuit. The pedestrians are placed uniformly inside the closed circuit and are instructed not to pass each other and not to hurry. The study takes measurements of sets with 15, 20, 25 and 30 pedestrians as they repeat the circuit several times.

To reproduce this experiment, I have designed a straight corridor the width of which is equal to Seyfried's section straight corridor measurement. The agents are a single group of 30 individuals placed in a line that begins at the middle point of the corridor to avoid border effects. The goal, that is, the place where the agents have to learn to arrive, is situated near the left end of the corridor. The dimensions of the measure section are equal to those in Seyfried's experiment and it is placed near the goal. A screen shot of the proposed environment set up is displayed in Table 5.4.

The behaviors of the learning agents have been modeled using the rewards shown in Table 5.4. The agents are able to move forward and backward and they are rewarded negatively if they go backward. Also, the property of moving straight to the goal without veering between the walls is rewarded positively measuring the value of the angle of the agent's velocity. The agent is rewarded with



(a) The environment

Goal reached	100
Crash against another agent	-10
Crash against a wall	-1
Rearward Movement	-15
Velocity with small angle respect to the goal	0.1

(b) Values of the immediate rewards

Table 5.4: Virtual scenario and reward values of the line walking experiment

a value of 0 when the selected action does not have the immediate consequences reported in the table.

The features of the state space in this experiment include the kinematic description of the two nearest neighbors (the agent in front and behind) and the features that describe one obstacle. The left graphic of Figure 5.14 shows the learning curve of the learning processes for 30 agents. The performance counts the times that an agent has reached the goal from the beginning. The shape of the curve indicates that the learning process has converged.

In the right graphic of Figure 5.14 the fundamental diagrams are displayed for both the Seyfried experiment[1] and our results. The density is calculated by counting in each frame the number of agents or fraction of agents present in the measured section (2 m long). The speed values of both experiments are means of the data that fall in the same range of density values. The values in our experiment have been provided by the simulation of 100 trials.

Considering the number of pedestrians, our graphic should be restricted to the range of values of the Seyfried experiment with 30 pedestrians. However, our results cover all the range of values of density and velocity that appears in the experiments by Seyfried with a different number of pedestrians. There are two main reasons for this situation. First, the experiments are not exactly the same. The fact that the pedestrians in the Seyfried experiment walk inside a closed circuit provides a stationary mode of walking that is not possible in an

[1]The data for the Seyfried's experiment are available at the URL://http.ped-net.org

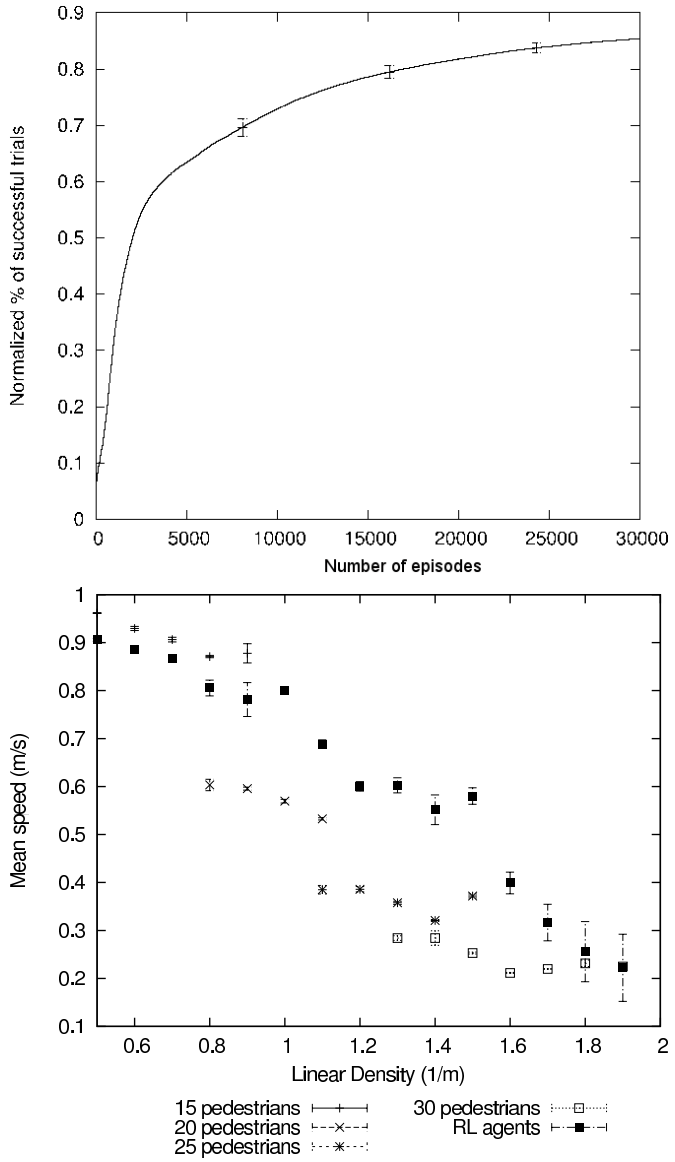


Figure 5.14: Line walking results. Top: Mean performance in the learning process. Data are the average of the learning performances of all agents. Performance counts the number of episodes in which an agent reaches the goal. Bottom: Fundamental diagram. The points are averages provided by Seyfried's public database and averages of 100 simulations in case of the RL framework. The standard error of each point is displayed.

open circuit like our proposed corridor. This facilitates the existence of gaps in the line which in turn generates low densities. Second, there are psychological

aspects that influence the behavior of the human pedestrians. Specifically, the fact that the test subjects in the Seyfried experiment have a global knowledge of the scenario means that their behavior was regular and predictable. A human tester in a closed line that has no possibility of passing will adapt his/her velocity to keep a reasonable space so as not to invade the vital space of other neighbors. The agents in our experiment, although negatively reinforced when crashing into a neighbor, do not have this global information of the whole situation and lack psychological restrictions. As positive indicators, we can observe that similar mean speeds are obtained for both, our agents and the real pedestrians in low and high densities. Moreover, the mean velocity decreases with the increment of the density in both experiments which is a characteristic of the fundamental diagrams of pedestrians (Schadschneider & Seyfried, 2009b).

The graphic visualization of the simulations in a 3D virtual environment also shows good behavior by the agents who perform a credible simulation of pedestrians through the corridor. The reader can see an example at URL `http://www.uv.es/agentes/RL/line.htm`.

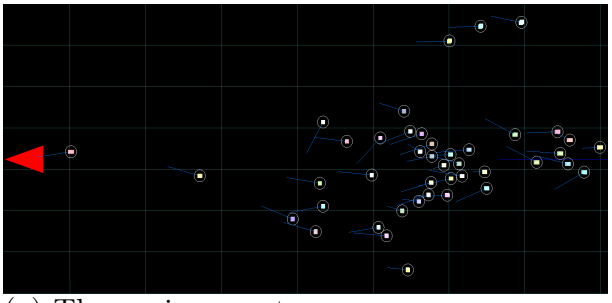
5.5.2 Walking without restrictions (open field walking)

In this experiment, walking on a bidimensional surface without obstacles is considered. I will compare the results with two works with real pedestrians. Weidmann's diagram (Weidmann, 1993) summarizes the data of 25 different pedestrian configurations without obstacles. This diagram is a reference for many other studies of pedestrian dynamics in planar facilities. We have also considered for comparison the experiment by Mori-Tsukaguchi (Mori & Tsukaguchi, 1987), performed in an open space (without obstacles) with data from photographs of flows of commuters in downtown Osaka City[1].

In our experiment, the bidimensional environment consists of a plane measuring 40 x 40 m (see Table 5.5 for a picture of the actual environment). The agents are placed far from a point that constitutes the goal. The only restriction to the free movement of an agent is the presence of other agents in the vicinity.

In this experiment, a group of 45 agents have learned to reach the goal in

[1]Data available at the URL http://www.ped-net.org



(a) The environment

Goal reached	100
Crash against another agent	-10
Velocity with small angle respect to the goal	0.1

(b) Values of the immediate rewards

Table 5.5: Environment and reward values of the open field experiment

presence of other agents who have the same goal. The features that describe the state space includes the description of the kinematic characteristics of the 7 nearest neighbors, and do not include obstacle descriptions. The learning curve displayed in Figure 5.15 (left) is a mean of the 45 learning curves of the agents. The shape of the curve indicates that the learning process has been successful.

The values of the immediate rewards that model the behavior of the agents are introduced in Table 5.5. Specifically, the third value of the table models the tendency to a smooth trajectory reported in Steiner *et al.* (2007).

In Figure 5.15 (right), the databases of Weidmann and Mori-Tsukaguchi together with our data in the range of densities achieved in our experiment are displayed. The values of our agents fit with those of Weidmann's and Mori-Tsukaguchi databases, with velocities slightly higher. A similar tendency of data is observed and the curves have comparable shapes with a decreasing trend as stated for all fundamental diagrams of pedestrians.

The visualization of the simulations gives, as in the previous experiment, a reasonable grade of plausibility. The reader can see a video at URL page `http://www.uv.es/agentes/RL/plane.htm`.

5.5.3 Conclusions

The results indicate similarities in the learned dynamics of the agents with those of the real pedestrians. The fundamental diagrams, used for comparing our results with the others described, share the characteristic of decreasing speed with

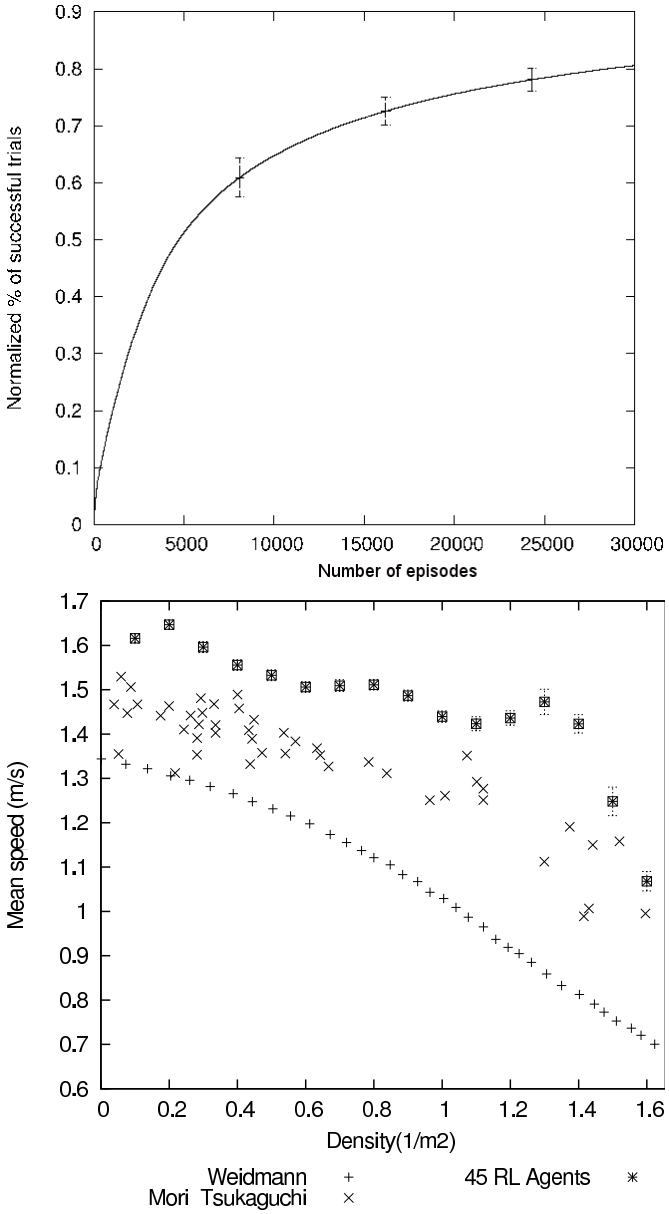


Figure 5.15: Top: learning process with 45 agents in the open field experiment. The data are the average of the learning performance for all the agents. Down: Fundamental diagram for the open field experiment in comparison with Weidmann and Mori-Tsukaguchi's databases. The points are averages of 100 simulations in our experiment with 45 agents. The standard error is displayed.

increasing density that is an important property the pedestrian dynamics. The visualization tests also support the results of the fundamental diagrams, showing realistic motion by the agents.

5.6 Chapter Highlights

- The framework has two working modes: learning mode and simulation mode. In learning mode, the policies of the agents are calculated. In simulation mode, the agents exploit these policies to generate the agents' behaviors.
- Each agent is an independent computational process that exchanges information with the environment. There is not data communication between agents.
- The calibration of the physics module consists of the specification of the dynamic parameters involved in collisions and interactions. These values come from studies with real pedestrians.
- The validation has been carried out using two scenarios: a line walking scenario and an open field scenario.
- There are several important concepts to remember:
  - Decision: this is the selection of an action that a learning agent takes in a time step.
  - Episode: this is the execution time defined by a number of decisions given to an agent to perform a task. Sometimes it is also named 'trial'.
  - Performance: this normally counts the times that an agent has reached the goal given a number of trials, that is, the number of successful episodes from the beginning to this stage. The definition can vary with the experiment.
- The results have been compared with available real pedestrian data from similar experiments. The comparisons indicate that the generated fundamental diagrams have the basic characteristic of the decrement of velocity

with the density. Moreover, in the second experiment similar curves have been obtained. The visualization of the simulations show plausible learned behaviors.

Chapter 6

Learning approaches based on VQQL

Applying RL to pedestrian simulation requires solving two important issues: facing the multi-agent problem and the generalization of the state space. In the works Fernández & Parker (2001); Fernández *et al.* (2005), it was shown that a cooperative task can be mapped to a single reinforcement learning problem in a multi-robot domain with a continuous state space. The authors used vector quantization (VQ) as the state space generalization system, and demonstrated that a single RL process using a state space description with relevant attributes to the task (that include information about the other robots), can develop cooperative capabilities. At the other end, our preliminary work (Martinez-Gil *et al.*, 2010), showed that a multi-agent system, where each agent carried out an independent RL process, was capable of converging in a navigational scenario without explicit coordination. In this work, the authors used a discrete state space which was inadequate in other scenarios, limiting the scope of application of this approach.

In this chapter a new methodology is introduced that addresses the two issues. It proposes the use of two algorithmic schemas, Iterative VQQL (ITVQQL) and Incremental VQQL (INVQQL), which differ in the way of addressing the problem of multi-agent learning. The algorithms use VQ as the generalization technique and, additionally, both use knowledge transfer techniques to accelerate the learning process. These algorithms are tested and compared with the VQQL

algorithm (see Section 3.3.1) as a baseline in two scenarios. In the first, agents in a closed room need to reach the single exit producing and solving a bottleneck. In the second, two groups of agents inside a corridor need to reach their goal that is situated on opposite sides (they need to solve the crossing).

The first scenario is suitable to study learned behaviors from the point of view of local interactions, because multiple and different interactions appear at high densities as happens inside a congestion produced by a bottleneck. Thus, I focus on the analysis of the dynamics of the learned behaviors, using the metrics described in Section 5.4.2, and comparing the results with Helbing's social forces model. Additionally, the problem of scalability in the number of agents is also considered. In the second scenario, an emergent collective behavior consisting of the formation of lanes has been obtained as a solution to the problem of crossing. This result shows that the agents' dynamics, which are the result of individual learning processes using local interactions, have acquired collective organizational capabilities.

6.1 Modeling the problems

This section describes the modeling options taken to define the considered navigation problems as MDPs.

6.1.1 The scenarios

The first scenario consists of a group of agents inside a closed room with a door. The agents have to learn how to reach the door and leave the room. Because of the reduced dimensions of the exit compared with the number of agents, a bottleneck is produced in front of the door. Moreover, a waiting crowd is formed upstream of the bottleneck forming a shell-shape distribution where several pedestrians compete for the same gap. This problem is described in several studies (e.g. (Helbing & Johansson, 2009)). The virtual environment is a $225\,m^2$ room with an aperture of $0.8\,m$ (which represents the door) in the center of one of the sides. The limits of the square are defined by walls, as shown in Figure 6.1.

The second scenario is a narrow corridor in which two groups of four agents

each, have to cross to reach to the opposite end. Lane formation is an emergent collective behavior that appears in real situations where the pedestrians' movements are constrained by borders (real or not) such as on urban sidewalks or corridors (Helbing *et al.*, 2005). In real-world situations, pedestrians with opposite directions of motion do not equally distribute over the cross section of a congested walkway. On the contrary, pedestrians organize into lanes with a uniform walking direction. This phenomenon maximizes the averaged velocity in the desired direction of motion, reducing the number of encounters with pedestrians moving in the opposite direction. The virtual environment is a corridor $15\,m$ long by $2\,m$ wide. Each group of agents is placed at one end of the corridor and their goal is to reach the opposite end. Thus, a crossing must be produced. (see Figure 6.1).

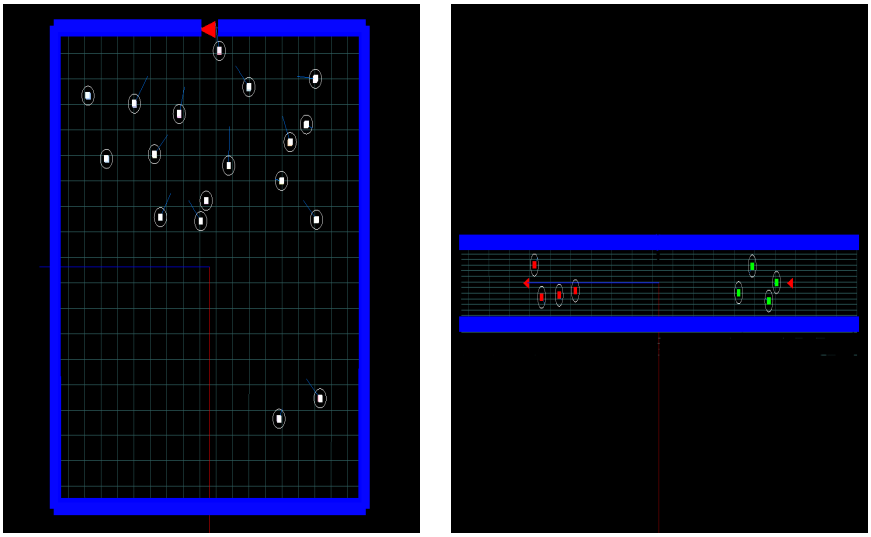


Figure 6.1: Images of the different scenarios. Left, first scenario: closed room with an exit. Right, second scenario: crossing inside a narrow corridor. Red triangles represent the goal.

6.1.2 State space

When representing the state space of a scenario, there is always a trade-off between accurately representing the kinematical situation around the agent and the

burden of dimensionality. In the first scenario (closed room), the number of maximum sensorized neighbors is seven, and can be varied in one of the algorithmic schemas (see the next section). Seven neighbors is an upper bound for the number of people nearest to a pedestrian in situations of high density ($\approx 2\ ped/m^2$) that are expected in this scenario (see Figure 5.13)[1]. This value sets the number of different possible configurations of the state space to eight (sensorizing 0, 1, 2, 3, 4, 5, 6 and 7 neighbors). In the second scenario, the number of maximum sensorized neighbors is four. A narrow corridor is more limited in terms of movements than a wide room, therefore I have considered relevant a number of neighbors that could be in contact directly with the agent. Four neighbors homogeneously distributed in the vicinity of an agent occupy practically all his sides. The number of sensorized static objects (walls) is always fixed at two. The maximum number of features describing one state space is 28 in the first scenario and 23 in the second scenario.

6.1.3 Immediate rewards

The behaviors of the agents are modeled according to the immediate payoffs listed in Table 6.1. In the first scenario, the payoff function models the prevention of collisions as an important task which a navigation controller must take into account. However, crashing into another agent is punished less than crashing into a wall, to lessen the shading effect of the frequent crashes in the bottleneck over the rest of the immediate rewards. The ignorance about the effects of other interactions is modeled with an immediate reward of value 0. In the second scenario, the immediate reward is simpler. It ignores crashes into agents and walls. In this domain, empirical tests were carried out with the rewards of the first scenario giving worse performance than the test with the proposed reward.

[1]Other works set a variable active perception distance for a pedestrian depending on parameters like the instant local density and the angle of vision (Bourgois *et al.*, 2012; Ondrej, 2011). In the case of (Bourgois *et al.*, 2012), an empirical formula is proposed for this distance that depends on the total number of sensed agents. According to this formula, for 20 sensed agents (approximately the number of learning agents in our first scenario), an active distance of 1.25 m is given. With this active distance of perception, an agent in density conditions of $2\ ped/m^2$ would perceive less than 10 neighbors which is similar in magnitude to my proposal.

First Scenario (closed room)	
Crash into other agent	-0.1
Crash into a wall	-2.0
Reach the goal	+100.0
Default	0.0
Second Scenario (Crossing in a corridor)	
Reach the goal	+100.0
Default	0.0

Table 6.1: Description of the values of the immediate rewards for both scenarios.

6.2 State space generalization

The states are generalized using Vector Quantization (VQ), which has been demonstrated to be an accurate approach for state space generalization and transfer learning in RL frameworks Fernández & Borrajo (2008); García *et al.* (2010, 2012). The use of VQ requires the definition of the number of prototypes to use, that is, the resolution of the state space. Typically, a coarse discretization composed of a reduced number of prototypes does not have enough expressiveness to represent the optimal value function. On the other hand, too many states introduce the generalization problem, although with a finite number of states. In order to fix the resolution of the space in both scenarios, two sets of experiments were carried out, testing different numbers of prototypes to represent the state space. In each scenario, a set of six experiments is performed with a specific number of prototypes: $k = \{512, 1024, 2048, 4096, 8192, 16384\}$, with the same configuration. In each experiment, the agents perform a series of random walks in the environment to get unbiased sensorization data. Then, the dataset is used to build the vector quantizer with a fixed number of prototypes. The sizes of the datasets ranges from 20000 samples in the case of 512 prototypes to 70000 samples in the case of 16384 prototypes. The calculated vector quantizers are different for each agent because they gather their own dataset and perform independent learning processes. Therefore, each agent uses its own vector quantizer in the VQQL learning process. Figure 6.2 shows the results from the set of experiments performed with the first scenario (the results for the second scenario are omitted because they are equivalent). In these experiments, the agents have

to learn to leave the room in a number of episodes[1]. In each episode, the agents are placed randomly inside the room and a maximum number of steps (the same in all the episodes and for all the agents) is given to get to the goal. At each step, the agent must take a decision (action) to adjust its velocity to the current local situation in the environment. In Figure 6.2, the x axis shows the number of episodes executed, and the y axis represents the normalized percentage of successful episodes, that is, the percentage of agents that leave the room. The quality of the learned behaviors is given by the asymptotic value of the curves at the end of the process. Note that the performance increases with the number of prototypes used in the quantizer (the resolution of the quantizer). However when duplicating the number of prototypes, the time for the assignment of the dataset to the clusters in each iteration of GLA is also duplicated. Therefore, when few prototypes are used, the learning process is faster, although the results are far from optimality. On the contrary, when many prototypes are used, the learning process is slower but the asymptotic performance increases. This conclusion is reflected in Figure 6.2, where the curves with a low number of prototypes have a high initial slope (which means fast learning), but the asymptotic part of the curve is lower than the curves with a high number of prototypes (which means a worse learning performance). The best trade-off between performance and computational cost was set to 4096 prototypes for the first scenario (agents in closed room) and 8192 for the second scenario (crossing in a corridor).

6.3 Description of the algorithms

The application of VQQL-based schemas to pedestrian navigation faces three challenges. The first one arises because pedestrian simulation is a multi-agent environment. Therefore, we first need to decide how many agents should learn from scratch. It is easy to understand that if we set 100 agents to learn from scratch, rarely will they be able to learn an effective policy. This is due to the fact that the learning agents could modify their policies while learning, which makes the environment non-stationary. Furthemore, a study of whether they should be

[1]The term 'trial' situated in the abscissa of the graphics has the same meaning as the term 'episode' in the text

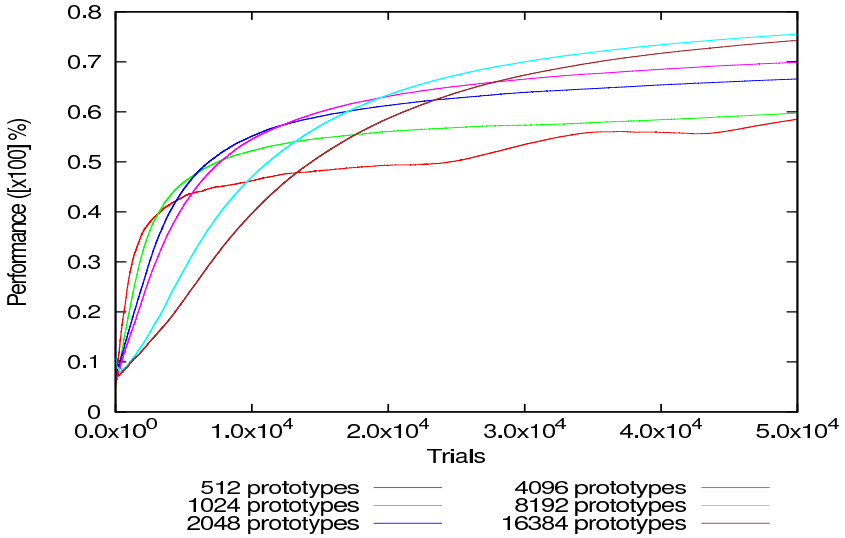


Figure 6.2: Analysis of the resolution for a VQ. Learning curves obtained in the first scenario using the VQQL algorithm and vector quantizers with different numbers of prototypes. The curves are the means of 18 learning processes (18 agents).

introduced into the environment from the beginning or gradually added is needed. However, this produces a second challenge: the agents can perceive a different number of agents in different situations, and therefore the space representation should be different. Therefore, two different situations can arise:

- Situation 1 (S1). The different descriptions of the state space are inherent in the incremental setting of the agents. For example, in a learning process for five agents, the space representation should consider the description of at most four neighboring agents. But in a learning process with six agents, the space representation should consider five neighboring agents.

- Situation 2 (S2). The evolution of the episode in time creates a variable perception of neighbors. When the agents are gradually reaching their goal, the rest of the agents can perceive fewer neighbors around.

The third challenge is that a random exploration of the environment does not produce a representative dataset with which to generate a correct set of prototypes for the VQ space generalization method. As stated in Section 3.3.1, the VQQL algorithm assumes that the dataset is significant, that is, the dataset

contains the relevant information to represent the whole state space. This is not the case with our domain, where a random walk can bias the exploration towards irrelevant states. Therefore, it is not interesting to model the whole state space uniformly, but rather the point is to get the states that are informative to solve the problem. Other domains, like a helicopter (García & Fernández, 2012; García-Polo & Fernández, 2011) have a similar problem, random exploration produces helicopter crashes, so it is unfeasible. Therefore, a process to select better datasets is necessary.

The decisions taken to address these challenges are included in the following explanation of the iterative schemas.

6.3.1 The iterative schemas

The iterative schemas consist of a pipeline of N learning processes carried out over the same problem, where the knowledge learned in an iteration is transferred to the next iteration. Two different approaches have been defined: Iterative Vector Quantization for Q-Learning (ITVQQL) and Incremental Vector Quantization for Q-Learning (INVQQL). Both perform different iterations of the VQQL algorithm's basic steps (computing the vector quantizer and learning the Q table) but in different ways, as described below. Additionally, in the two schemas, the learned policies in an iteration are used to gather a policy-biased dataset for building a new state space model that will be used in the next learning process. This process is referred to in this text as *policy transfer* and is carried out in Line 11 of Algorithm 10. However, the schemas differ in the way of placing the agents in the scenario and, hence, in the number of agents learning at the same time in each iteration and with the configuration of the vector quantizer. Specifically, regarding the number of agents per iteration:

1. The ITVQQL schema puts the same number of agents for every learning iteration. Moreover, it models the state space with one set of prototypes with a fixed number of features to represent the neighboring agents.

2. The INVQQL schema puts the agents into the stage beginning with one agent in the first iteration and incrementing by one more agent in each following iteration to reach the total number of agents in the last iteration.

Regarding the definition of the vector quantization:

1. The ITVQQL approach represents the state space using prototypes with a constant number of features, which means that different perceptions (in terms of sensorized neighbors) must be described with the same space vector and therefore with a fixed number of features (this problem appears in situations (S1) and (S2)).

2. The INVQQL approach uses different space vectors to represent the different perceptions in term of the neighborhood ($0, 1, 2, \ldots$ neighboring agents). Therefore a collection of vector quantizers with their respective sets of prototypes with different number of features is used.

The description of both schemas are grouped under Algorithm 10. From Line 2 to 6 of the algorithm, the counters, the quantizer, and the policies are initialized for each agent. The loop begins in Line 7. Each iteration is an entire learning process where the sub-steps described in Lines 9 to 12 correspond with the execution of VQQL. The step described in Line 12 is an entire Q-learning process. In this step, the initialization of the Q-tables, as well as the parameters of the exploratory policy, are carried out. Next, an iterative process with a fixed number of episodes is performed to get an approximation of the optimal policy for the problem. The number of iterations, N, should be configured differently for each schema. For the INVQQL schema, the number of iterations, N, agrees with the total number of agents with which the experiment was planned. For the ITVQQL schema, the number of iterations, N, are enough to achieve a good learning performance (see curves for ITVQQL schema in Figure 6.4, Figure 6.5 for the first scenario and Figure 6.7 for the second scenario). In the experiments presented in this chapter, the same number of iterations is used for both schemas to set similar experimental conditions and to compare the results.

In Algorithm 10, the decisions that differentiate the schemas have been underlined and are explained in the following enumeration:

1. ITVQQL. A fixed number of learning agents is set in Line 3. Therefore, no new agents are included in Line 8. There is no specific transfer between quantizers in the step of Line 11. The state space representation has a fixed

Algorithm 10: The ITVQQL/INVQQL schemas.

Data: The number of iterations N
Result: The sets Q_N and V_N (The final value function of Q-learning and the vector quantizer respectively).

```
1  begin
2      i ← 1;
3      Set p to the initial number of agents in the environment;
4      for k ← 1 to p do
5          V_0^k(s) = ∅  //initial vector quantizer of agent k;
6          π_0^k = random  //initial policy of agent k;
7      repeat
8          Decide whether or not to include new agents. Set p consequently.;
9          for k ← 1 to p do
10             Collect a dataset T_i^k for agent k using policy π_{i-1}^k with
               V_{i-1}^k;
11             Build V_i^k using T_i^k for agent k following a transfer learning strategy;
12         Learn Q_i^k ∀k, 1 ≤ k ≤ p and hence the policies π_i^k using
           Q-Learning (with the option of using transfer of value
           functions).;
13         i ← i + 1;
14     until i = N;
```

number of features. The problem discussed in (S2) is solved by setting unobserved features to random values[1].

2. INVQQL. The interactions between the agents are learned gradually by increasing the number of agents from one iteration to the next. This can be understood as a kind of shaping (Sutton & Barto, 1998), where a more complex task is achieved through the learning of easier tasks, which are oriented towards the solution of the final task of ultimate interest. In Line 3, there is no agent at the beginning ($p = 0$). In Line 8, one agent is added in each

[1]In Machine Learning, many different approaches are used to fill in unobserved features. We have studied some of them, specifically random imputation and mean imputation, obtaining similar performances.

iteration. The novelty here is that this schema calculates a different VQ for each state space of different dimensionality incrementally. The calculated VQs are transferred to the next iteration in the step described in Line 11, avoiding their recalculation. For instance, in the first scenario, a learning process of eight agents has inherited the previous VQs which models the space state sensorized by 7, 6, 5, 4, 3, 2, and 1 agents. In the first scenario, the number of prototypes increases in the first eight iterations (of a total of 18 iterations), modeling the eight possible different configurations of our experimental setting. In the second scenario, the number of prototypes increases during the first four iterations (of a total of 8 iterations).

An additional problem arises in Line 10 when using the policies learned in one state space configuration to collect data in a space with a newly added agent and, therefore, with another state space configuration. The policy transfer between spaces with a different number of features is carried out using a projection. A projection can be understood as a selection of features $\Gamma : R^m \rightarrow R^s$ where $m > s$. Therefore, the projection is from the higher dimensional state space to the lower dimensional one. The set of vector quantizer V_s and the value functions Q_s of the previous iteration can be used to collect biased experiences in the highest dimensional state space. The difference between two spaces of different dimensions in our scenarios is always constituted by the features that describe a new neighbor agent in the higher dimension space. In practice, the projection is carried out by deleting this new set of features. Such mapping between different state spaces has been evaluated successfully in previous transfer learning problems (Fernández *et al.*, 2010; García *et al.*, 2010) and state abstraction problems (Chiu & Soo, 2008).

As defined above, INVQQL goes a step further because the prototypes learned in previous iterations, where the state configurations are different in terms of the neighbors perceived, are transferred to the next iteration. In this way, the whole range of different state configurations are covered. In Table 6.2, the properties of each schema are summarized.

Feature	ITVQQL	INVQQL
Number of prototypes	Fixed	Variable
Number of features per prototype	Fixed	Variable
Number of agents per iteration	Fixed	Variable
Inter-iteration policy transfer	Yes	Yes
Inter-iteration prototype transfer	No	Yes
Inter-iteration value function transfer	Yes	Yes

Table 6.2: Summary of the settings of the schemas.

6.3.2 Value function transfer procedures

In this section, using the transfer of the value function learned in a previous iteration as the initial values for the value function of the next one is proposed. The source task and the target task are the same but with different state representations. This problem can be considered a simple case of those described in Taylor *et al.* (2007). In that paper, the authors focused on the problem of representation transfer, that is, how to transfer the learned knowledge between tasks that differ in the function approximator (the state space generalizer) or in the learning algorithm. Specifically, they call the transfer of an inter-task value function between different representations of the state space *complexification*. The transfer proposed here is a form of complexification and it is described in Algorithm 11. The idea is to initialize the target Q table (Q_{target}) with the learned values of the source Q table (Q_{source}). The transfer is performed using a metric for the similarity between the prototypes of the source and the target vector quantizers. Thus, for each prototype of the quantizer associated to Q_{target}, the nearest prototype of the quantizer associated to Q_{source} is calculated. Then the values of Q_{source} are transferred using this association. In the experiments, the algorithm uses the Euclidean metric to determine the similarity between the sets of prototypes. In this case, the transfer requires the two Q tables to have the same dimension and parametrization. Although the initial values are the same as those of the learned values of the previous iteration, the set of prototypes used in the new configuration is different to the previous ones and the epsilon-greedy policy begins the

Algorithm 11: Simple Complexification with a Q-table

begin
  Train with a source representation and save the learned Q-table and the vector quantizier Q_{source}, V_{source};
  forall the *prototypes* $q^i_{target} \in C_{target}$ **do**
    Find prototype $q^j_{source} \in C_{source} \mid \min_j \|q^i_{target} - q^j_{source}\|$;
    $Q_{target}(q^i_{target}, action_k) = Q_{source}(q^j_{source}, action_k)\ \forall action_k$;

new iteration with high rates of exploration. Therefore, the new process *is not in any case* a mere continuation of the previous learning process.

Several metrics to evaluate transfer learning methods have been proposed (Taylor & Stone, 2009). In the present paper, the following metrics are considered for testing in the performance curves: i) Jumpstart: the improvement in the initial performance of an agent. ii) Asymptotic Performance: the improvement in the final performance. iii) Time to Threshold: the learning steps needed by the agent to achieve a pre-specified performance level.

The results of the evaluation will be shown in the next section.

6.4 Experimental set-up and performance results for the learning process

This section describes the learning results in terms of mean performance.

6.4.1 Closed room with an exit scenario

In all the learning schemas described above, the number of iterations performed is 18 ($N = 18$). [1] Each iteration has been set empirically to $50,000$ episodes. An episode ends when all the agents reach the goal or when a maximum of 150 decisions have been taken. The learning algorithms have the same parameter configuration in all processes and this is summarized in Table 6.3 for all agents. The

[1] In the experiments, I will show that 18 iterations is a value large enough to ensure convergence in all the proposed scenarios

γ	0.9
α	0.4
ϵ	0.4 (initial value)

Table 6.3: Specific values for the learning parameters which are common to all the experiments in the first scenario.

value of ϵ is decremented exponentially with the number of episodes following the expression $\exp(\frac{-episodes}{k})$, where k is a constant to regulate the decay whose value depends on the total number of episodes. All the agents learn simultaneously. That means that all the agents are in the same step of the same episode. This configuration favors a similar progress in learning for all the agents, palliating the intrinsic non-stationarity nature of the environment.

The dataset gathered in each iteration to learn the vector quantizer (step described in Line 10 of Algorithm 10) is generated using a ε-greedy policy to avoid the overfitting of the training data to the previously learned policy: if the learned policy is used fully greedy, the state space visited may be excessively restricted by the policy, so adding a small amount of noise in the action selection can reduce this effect. Before using the GLA algorithm, the collected data are standardized (each feature has zero mean and standard deviation equal to 1). In Figure 6.3, the effect of the refinement of the VQs along an execution of the ITVQQL schema in the first scenario is shown. The graphics display the values of two features of the 4096 prototypes that constitute a VQ. Specifically, in the X-axis, the speed of the agent is represented while in the Y-axis the distance to the goal is displayed. Because the features are standardized values, negative values for distances and speeds appear. The prototypes of the first iteration (on the left) and those of the last iteration (on the right) are displayed. In the first, the prototypes are distributed quite homogeneously in the space because the data are collected using a random policy as indicated in Line 6 of Algorithm 10. In the graphic on the right, a high density of prototypes in the region of low speed and low distance to the goal is observed, showing the bias in the collected data generated for a learned policy. The prototypes located in the high density region are probably generalizing states of agents placed near the door, where low distances are associated to low speeds due to the bottleneck.

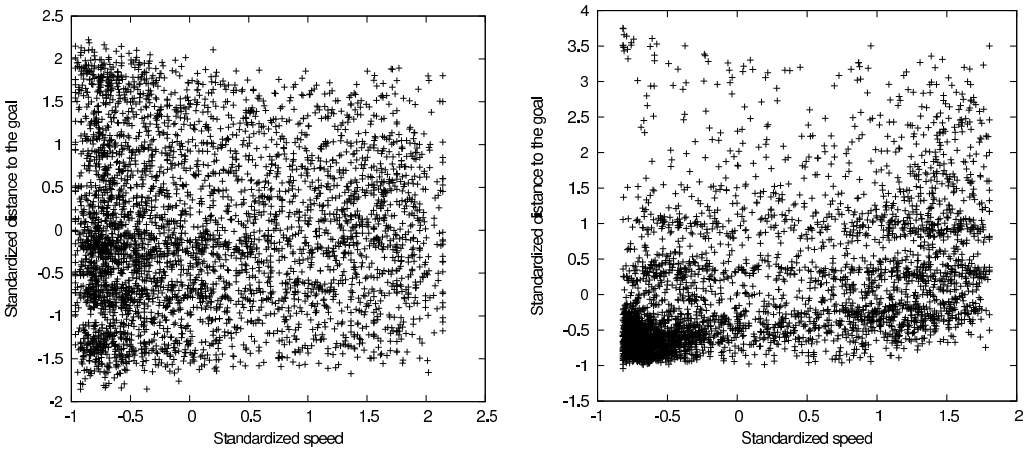


Figure 6.3: Influence of learning in the distribution of prototypes in the features space. Visualization of the prototypes calculated for first iteration (left) and last iteration (right) when ITVQQL schema is used in the room scenario. The speed (in the x-axis) and the distance of the agent to the goal (in the y-axis) features are displayed. Prototypes of the left graphic are calculated using data collected with a random policy. Prototypes of the right graphic are calculated using data collected with a learned policy correspondent to the penultimate iteration.

Figures 6.4 and 6.5 summarize the results of the learning processes for the schemas without and with transfer of value function respectively. Each point represents the averaged final performance reached by the agents when they use a greedy policy over the learned value functions in that iteration, that is, an iteration of the loop in Line 7 of Algorithm 10. For each iteration, a simulation of 100 episodes without learning calculates the performance of the value functions. The performance is measured in terms of the percentage of successful episodes in the simulation. In Figure 6.6, the shapes of the performance curves for one learning process along all the episodes can be seen. The mean performance given by the learned value functions calculated in this process will constitute a point displayed in the curves of Figures 6.4 and 6.5. As an example, the ITVQQL averaged curves displayed in Figure 6.6 correspond to iteration number three, and the averaged performance of the resulting value functions becomes the point of abscissa number 3 in Figures 6.4 and 6.5. The VQQL algorithm is included for comparison. VQQL is not iterative, and the graph displays a learning process of 900,000 episodes, which is the total number of episodes completed by the

other schemas in the 18 iterations. Each point in the VQQL curve is calculated using the state of the value functions in the corresponding number of episodes (first iteration equals 50,000 episodes, second equals 100,000 episodes, and so on). Table 6.4 shows the specific configuration of the parameters described in Table 6.2 for the experiments.

Key	ITVQQL	INVQQL	VQQL
Episodes	50000	50000	900000
Iterations	18	18	1
Prototypes	4096	From 4096 to 32768	4096
Features per prototype	28	From 7 to 28	28
Agents per iteration	18	From 1 to 18	18
Inter-iteration prototype transfer	0	4096	0

Table 6.4: Specific settings for the learning experiments in the closed room scenario.

In the ITVQQL schema, the first point in the curves (with and without transfer) shows lower performance than those in INVQQL because the second uses one agent while the first schema uses 18 agents. Note in Figure 6.4 that after a short number of iterations, the learned policies do not improve the performance. A similar curve appears in Figure 6.5, although a slight increment with fewer oscillations is seen.

The INVQQL schema works with a state space with eight possible feature configurations (from 0 to 7 sensorized neighbors), creating eight sets of 4096 prototypes to quantify each state space configuration. The first eight points in both graphs decrease in their performance because the dimensionality of the state space grows with each iteration (because a new neighbor is sensorized). At iteration eight, the highest dimensionality is reached and the process begins to improve. This improvement appears in INVQQL without transfer due to the fact that the new VQs stop growing in size and a process of refinement along the rest of the iterations begins. In INVQQL with transfer, the process of transfer of value function begins and adds its effect to that explained before. Note the jump between iterations eight and nine in Figure 6.5, where the transfer is applied for the first time. In this schema, the use of a transfer of value function positively

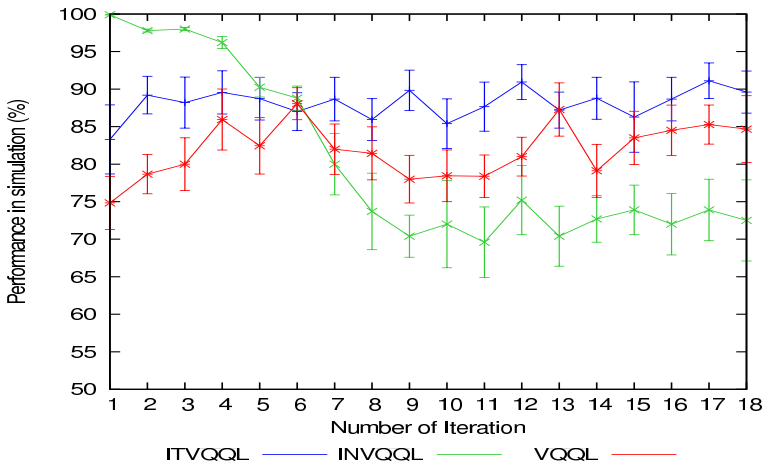


Figure 6.4: Performance of the learning processes for all schemas without transfer of value functions in the closed room scenario. Performance is calculated using a greedy policy over the learned value functions. A simulation with greedy policy without learning, with a total of 100 episodes with 18 agents per iteration was carried out to calculate each point. Points are sorted by iteration number inside the learning process.

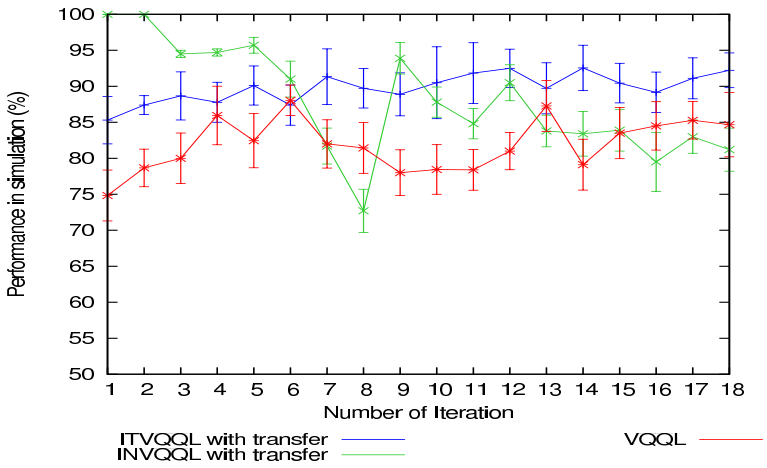


Figure 6.5: Performance of the learning processes for all the schemas with transfer of the value functions in the closed room scenario. Again, the performance is calculated using a greedy policy over the learned value functions. A simulation with greedy policy without learning, with a total of 100 episodes with 18 agents per iteration was done to calculate each point. The points are sorted by iteration number inside the learning process.

modifies the performance curve.

The VQQL curve in Figures 6.4 and 6.5 represents the performance of one iteration of the algorithm with a number of episodes equal to the addition of the episodes for all the iterations in any other schema. Obviously there is no transfer of value functions. The ε parameter for exploration is adjusted to the number of episodes as well as the learning ratio parameter α. The first point of the curve (iteration 1) is lower than in the rest of the schemas because at this point this algorithm has a high rate of exploration. In comparison with the curves in Figure 6.2 that also correspond to a VQQL learning process (although with different configurations), the performance results are higher here because we display the mean performance of a greedy policy which exploits the value function. On the other hand, the graph in Figure 6.2 displays the mean performance of a process that uses exploration, especially in the early episodes, which causes a decrease in the total performance. The same effect explains the lower values in Figure 6.6 of the iterative schemas compared to the values reached in Figures 6.4 and 6.5.

Figure 6.6 compares, for each schema, the performance curves of the learning processes with and without transfer in the last iteration. The three metrics introduced in Section 6.3.2 (i.e., jumpstart, asymptotic performance and time to threshold) are easily checked in the graphs. The jumpstart is very clear in both schemas. This means that the transferred knowledge accelerates the new learning process. This conclusion is supported by the fact that the curves with transfer are crescent[1]. The time to threshold is also highly influenced by the transfer. The threshold has been set to 0.7 (70% of success) as a limit where the good behavior begins. Note that the processes with transfer substantially advance the arrival at the threshold saving at least half the number of necessary episodes for the ITVQQL schema, this being much more dramatic in the INVQQL schema. The asymptotic performance is also improved with the transfer in the ITVQQL schema and more strongly in the INVQQL schema. Summarizing the transfer results, the acceleration of learning exists in the ITVQQL schema and it is especially important in the INVQQL schema. In the case of the INVQQL schema,

[1]If the transferred knowledge was not useful, the agent should unlearn it and the learning curves would have a decreasing zone.

it seems to be connected with the fact, discussed above, that it uses, in the displayed iteration, 32,768 prototypes, while the other schemas use 4096 prototypes. Therefore, the number of states to explore and evaluate in the INVQQL schema requires more episodes to get enough experience, and this problem is alleviated with the knowledge transferred.

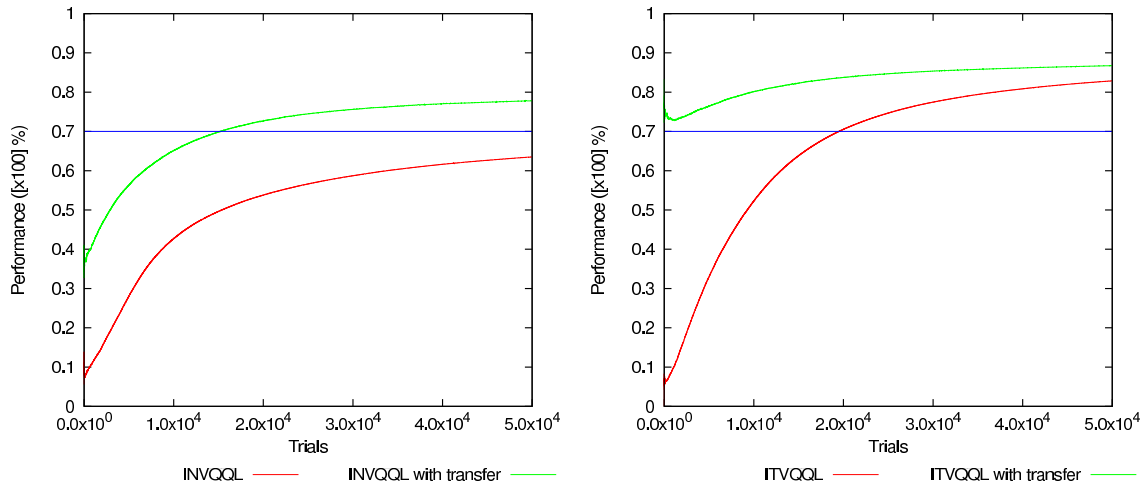


Figure 6.6: Comparison of performance curves for learning processes with and without transfer of the value function for INVQQL (left) and ITVQQL (right). These curves correspond to the last iteration and averages over 18 agents.

From the previous evaluation, two main conclusions can be drawn:

1. It has been empirically proved that the learning processes converge in both learning schemas, for the considered domain.

2. The transfer of knowledge especially benefits the INVQQL schema in the learning task. In this schema, this high benefit reveals that the chosen number of episodes per iteration were not enough to learn properly, probably due to the size of the Q table's being greater than the sizes of the other schemas. The transfer of knowledge alleviates this lack of episodes.

6.4.2 Crossing inside a corridor scenario

Eight agents ($N = 8$) are divided in two groups that move counterflow inside the corridor. Each iteration has 50000 episodes. The INVQQL schema needs 8

iterations to insert one by one the total number of learning agents, therefore the number of iterations has been set to 8 in both schemas so as to be comparable in number of learning processes, as in the first scenario.

The dynamics of the episode have been described in the previous subsection with the first experiment. Tables 6.5 and 6.6 summarise the configuration of learning parameters [1]. In this problem, a learning rate decreasing with the number of iterations was used. As indicated in Table 6.5, the value of α begins in the first iteration to 0.3 and decreases to 0.1 in the last iteration.

γ	0.9
α	from 0.3 to 0.1
ϵ	1.0 (initial value)
ψ	1.0 (initial value)

Table 6.5: Settings for the learning parameters common to all the experiments in the second scenario.

Key	ITVQQL	INVQQL	VQQL
Episodes	50000	50000	400000
Iterations	8	8	1
Prototypes	8192	From 8192 to 40960	8192
Features per prototype	24	From 8 to 24	24
Agents per iteration	8	From 1 to 8	8
Inter-iteration prototype transfer	0	8182	0

Table 6.6: Specific settings for the learning experiments in the second scenario.

In this scenario the experiments are carried out with transfer of value functions proved in the first scenario that equal or increase the performance of the schemas and accelerate the learning process. The graphics for the performance are displayed in Figure 6.7 (left). The INVQQL schema begins with a 100% performance because in the first iteration only one agent is learning. The shape

[1] Assuming that a soft variation in the values of the parameters produce a soft variation in the learning performance (the experiments agree with this assumption), the way to find the values for the learning parameters consists of a coarse search inside the allowed values followed by a refinement over the candidate with better performance.

is similar to that of the first experiment because the agents are introduced incrementally in the iterations. The first iteration of the VQQL schema has the lowest performance because, in this stage of the learning process, the exploratory policy is still dominant. As ocurred in the first experiment, the ITVQQL with transfer achieves a higher performance in the last iteration compared to the rest. The asymptotic shape indicates that not all the iterations were necessary for this schema.

In order to accelerate the learning process, the *probabilistic policy reuse (PPR) transfer technique* is used (see Section 3.4 for a description of the method, and specifically, Equation 3.25 and Algorithm 9). In this problem, the policy π_{past} always suggests the use of an action that drives the agent towards one side of the corridor [1]. Contrary to other works, the policy π_{past} does not come from a previous learning process carried out in this or another similar task, but it comes directly from real experience. In this case, PPR does not transfer learned knowledge strictly, but it is used as a way of introducing knowledge from experience. It is important to note that PPR is used as a way of efficiently exploring the space of policies to find a solution for the crossing problem. If the agent does not find it useful to follow the policy π_{past} in a state, it will learn a better policy because the exploratory policy is active throughout the whole learning process. In Section 7.4, a more exhaustive analysis of the benefits of using PPR in this task is performed.

6.5 Pedestrian simulation

In this section, the policies learned from the point of view of the pedestrian simulation are analyzed. The value function learned by each agent in the last iteration of the processes is used in the simulation tests for two reasons: i) They represent the culmination of their respective learning iterative schemas ii) In the last iteration, the number of agents is the same for both iterative schemas. The analysis of the simulations focuses on different aspects in both scenarios. In the first, the focus is on the dynamics. A macroscopic and a microscopic study of different metrics to characterize the learned dynamics is presented. In the second,

[1] Specifically, the policy π_{past} chooses randomly from the set of actions that turns the agent's velocity vector towards the right side of the corridor

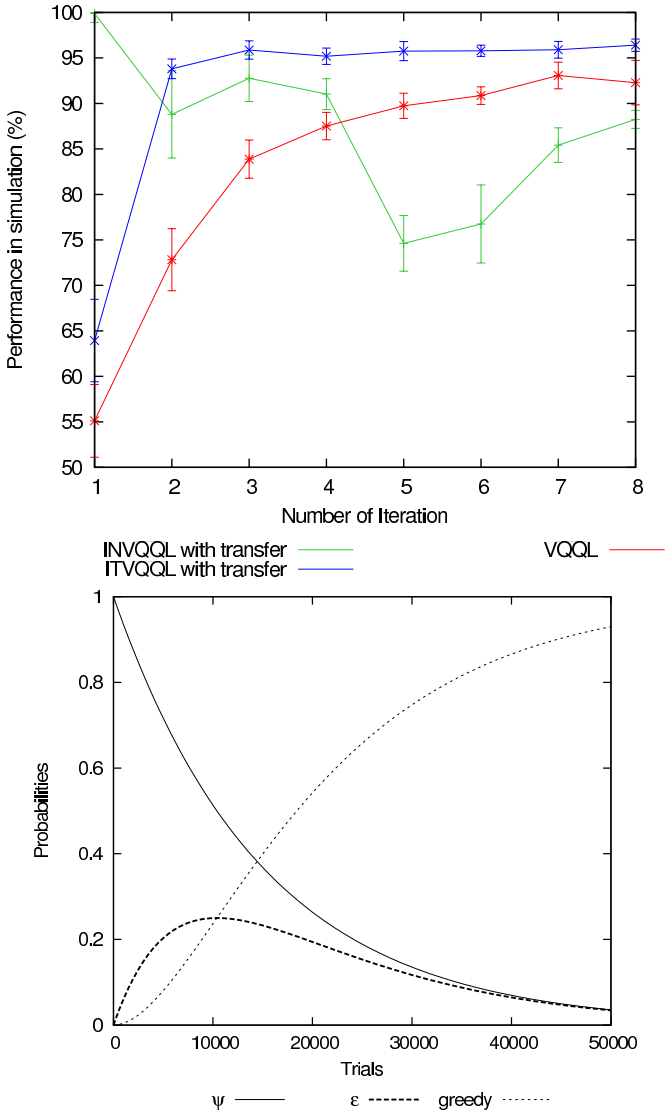


Figure 6.7: Top: Learning process performance for all the schemas with transfer in the crossing scenario. Performance is calculated using a greedy policy over the learned value functions. A simulation with greedy policy without learning, with a total of 100 episodes with 18 agents per iteration was performed to calculate each point. Points are sorted by iteration number inside the learning process. Down: Probability functions used in PPR transfer technique

we focus on the emergent collective behavior, therefore a macroscopic study is performed to determine the shape of the emergent lanes and its influence on the performance. In both, a comparison at the macro-dynamic level with the Helbing

social forces model is performed. The reader can complement the data provided in this analysis by viewing several videos at URL `http://www.uv.es/agentes/RL/index.htm`

6.5.1 Simulation metrics

To evaluate the quality obtained in our simulations, three different kind of metrics have been used:

1. Local interactions (microscopic level): Velocity *vs.* collision distance correlation.

2. Macro-dynamics (macroscopic level): Fundamental diagram of pedestrians (speed *vs.* density relationship) and density maps.

3. Performance: Path length, number of decisions per episode and number of fails (episodes where the agent did not reach the goal).

Firstly, the micro-dynamic metrics are used to analyze the local interactions of the pedestrians, by measuring the speed controller reactions under specific situations. The most interesting situation to analyze here is the collision response, so we have correlated the distance to the nearest neighbor with agent speed, in order to study this important relationship.

Secondly, macro-dynamics are also very interesting to evaluate the behavior of the simulated group, and it is also an easy way to contrast the proposed schemas. In this case, we use density maps to represent the space allocation during the simulation, and the fundamental diagram that summarizes the local interactions that agents have learned in a single diagram. Other works on pedestrian modeling that have used similar metrics are Daamen & Hoogendoorn (2003); Steiner *et al.* (2007); Still (2000).

Some scalability tests have been performed in the simulation experiments for the first scenario. To scale in the number of agents, each simulated agent uses one copy of one of the learned value functions with its corresponding vector quantizer. In simulation, there are as many different behaviors as agents in the learning process, therefore several agents use the same learned policy. Thus, in

this section, all the experiments for the first scenario with more than 18 agents use copies of the original set of value functions and vector quantizers. Before reassigning a copy of a behavior to an agent for the n-th time, the rest of the behaviors have had to be reassigned $n-1$ times.

Finally, different performance-oriented metrics have also been included, such as path lengths, the number of required decisions, and the number of fails, which are also interesting for comparing the schemas.

6.5.2 Local interaction analysis for the first scenario

In this section, I analyze the micro-behavior of the agents in the first scenario and focus specifically on the agents' learned capacity to regulate their speed when interacting with the environment. These interactions represent the normal navigation conflicts and situations produced in collision avoidance scenarios for embodied autonomous agents. For this analysis, two parameters of the agent's state space have been observed: the speed of the agent and the distance to the nearest neighbor. In a collision scenario, it seems reasonable that both parameters would have some correlation. That is, when the distance is shortened, the agent should reduce its velocity, and *viceversa*. Nevertheless, this is not always expected, since these parameters do not give a complete description of the local situation. For example, the nearest neighbor might be approaching from behind, therefore the agent could accelerate if no other agent is in front; or, in other situations, if the distance to the neighbor is large, the proximity of a wall could force a reduction in the agent's speed. Therefore a positive correlation is expected between these two parameters although it should not be a total correlation. In Figures 6.8 and 6.9, these parameters have been plotted for one agent in an episode and for different learning schemas. The graphics tell the story of an agent in a episode. The schemas have used the transfer of value functions in the learning phase. The graphs show representative cases of a simulation without scaling in the left column (18 agents) and scaling by five (90 agents) in the right column. Note that each graph represents an individual episode, therefore the number of steps is different in each one, depending on the random initial positions and the different local interactions that appear in it. For the same reason, the scales of the Y axis

for the velocity and the distance are different in each graph. For the purpose of comparison, two more experiments have been added, as shown in Figure 6.9. In that figure, the first row represents the results for the basic VQQL schema while the second row presents the results of the RANDOM experiment, where the agents take random actions in each decision step, that is, they use no learned knowledge. The VQQL and RANDOM experiments have the same number of agents and the same configuration as the other schemas.

The curves of the graphs of Figures 6.8 and 6.9 have two different zones in terms of the collision situation. The zones of minimum distance, with a value around $0.6\,m$, represents a situation where the neighboring agent is practically in contact with the observed agent. In this situation, a continuous acceleration (to avoid the neighboring agent) or deceleration (due to a new approach and possible crash) creates the speed oscillations in these zones of the curves. Note that the frequencies of the oscillations are not comparable in a simple observation of the graphs because of the different number of decisions in the presented episodes. In comparison, in the RANDOM experiment (Figure 6.9), the speed oscillations are present at any value of the distance. For example, in the graphic of the left column for the RANDOM experiment, between decisions number 0 and 20, the nearest neighbor is far from the agent but the speed oscillates from 0 to a maximum value of 1.0 with high frequency. This effect does not happen with an agent with a learned policy as observed in the VQQL graphic (Figure 6.9 above).

In the left VQQL graphic, a similar situation as described for the RANDOM experiment between decisions number 1 and 7, is observed. However, in the VQQL graphic the speed increases monotonically and decreases when the distance becomes small. On the other hand, the curves show that, in general, the speed is increased by the agent in the zones of the curves where the distance to the nearest neighbor increases, reaching the maximum allowed speed (1.8 m/s) when the distance is large enough. See for example the zones at the beginning of the curves in the left column for all the learning schemas. The horizontal sections in the region of maximum speed match the regions of the distance curve where the distance is large: this is considered to be a good indication of the stability of the learned behavior (for example, see this situation in the left column graph of the ITVQQL schema around decision 7, and also in the left column graph of

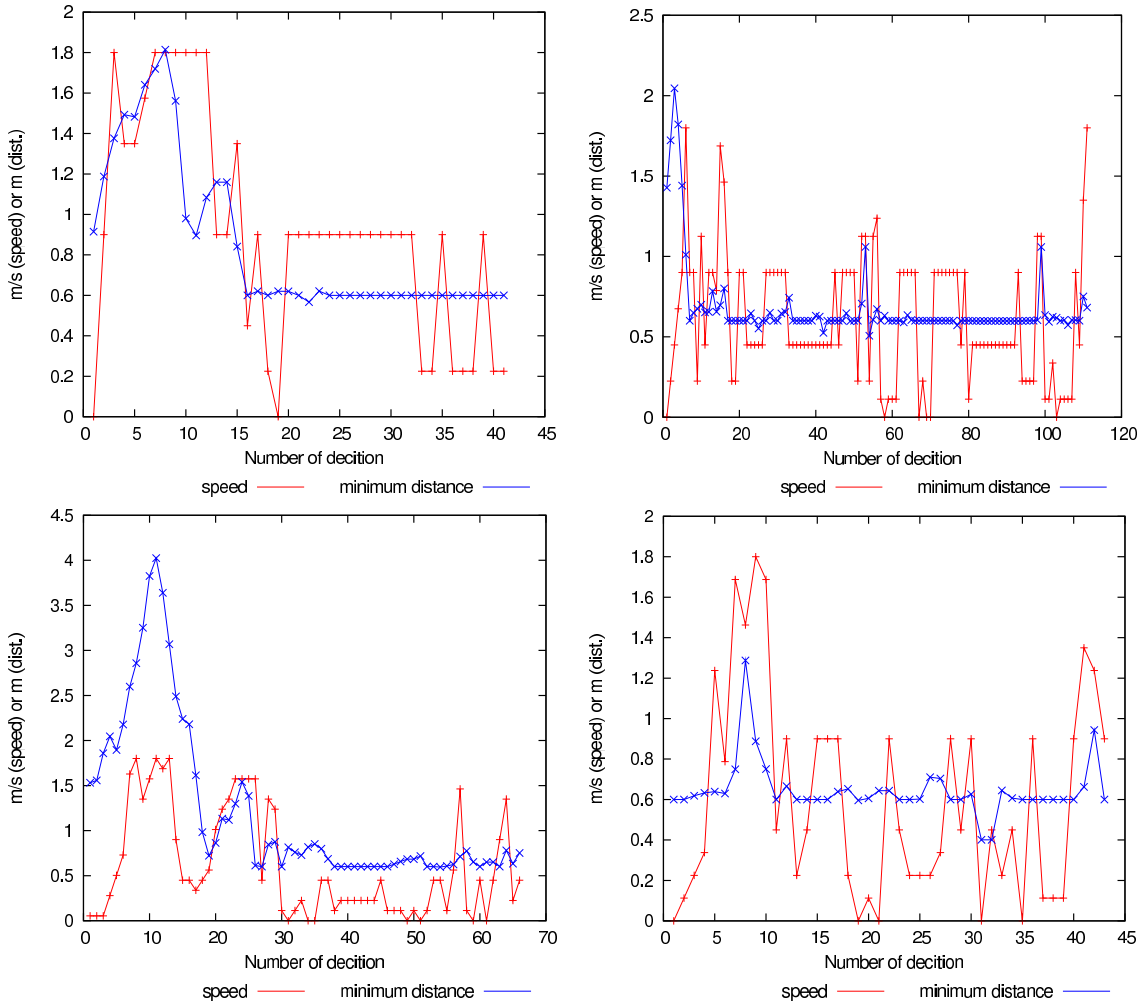


Figure 6.8: Local interaction results for an agent in an episode. All the schemas have used transfer in the learning phase. The order of the displayed schemas are: ITVQQL in the first row, INVQQL in the second row. First column is without scaling (18 agents), and second column, with scaling (90 agents). The blue curve displays the distance to the nearest neighbor and gives an idea of imminence of a collision. The red curve represents the velocity proposed by the learned controller. Along the abscissa, the number of decisions is a measure of time.

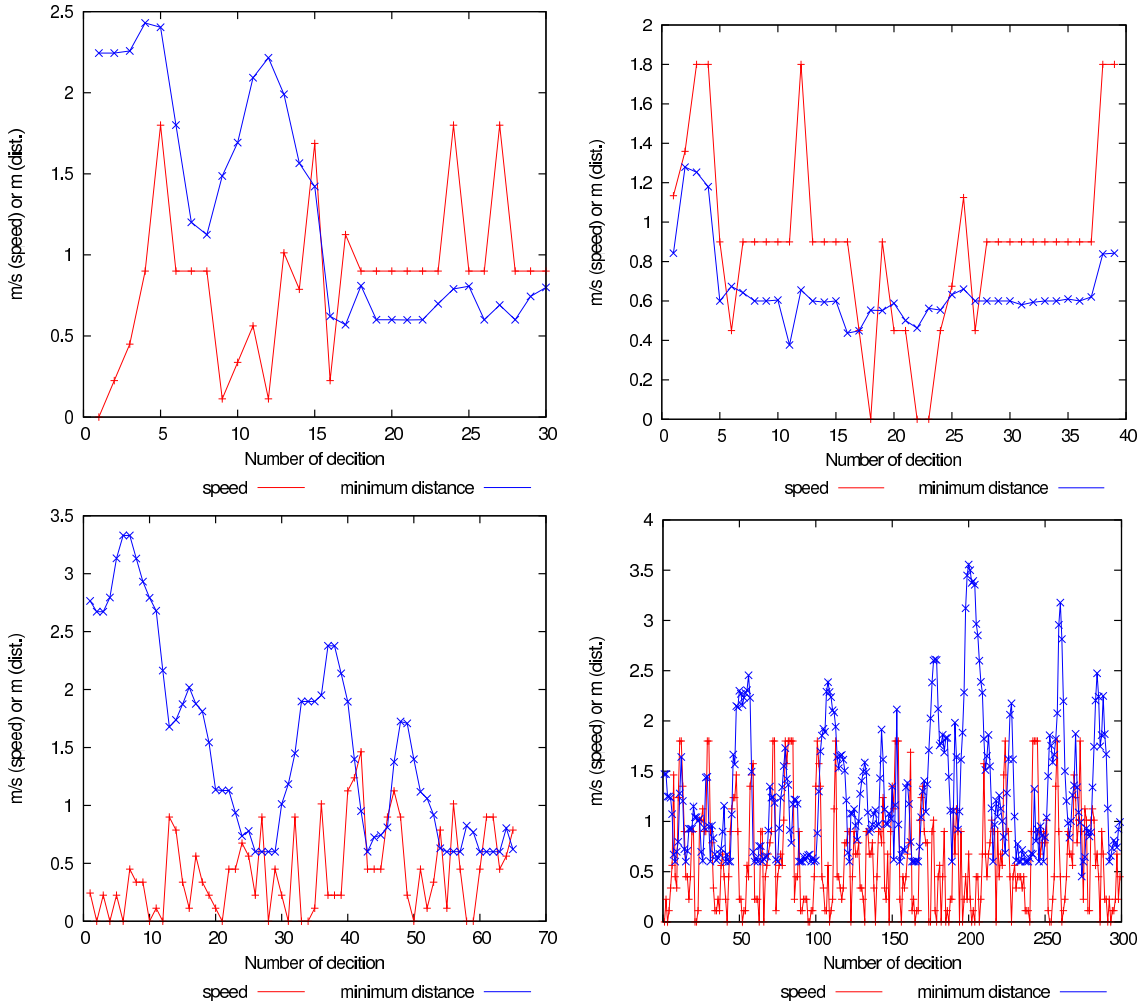


Figure 6.9: Baseline experiments for local interaction results for an agent in an episode. The order of the displayed schemas are: VQQL in the first row and RANDOM in the second row. First column without scaling (18 agents), and second column, with scaling (90 agents). The blue curve displays the distance to the nearest neighbor and gives an idea of the imminence of a collision (when the minimum distance is 0.6). The red curve represents the velocity proposed by the learned controller. Along abscissa, the number of decision is a measure of time.

the INVQQL schema around decision 26). Also, this property appears in the VQQL baseline curves (in the right column in decisions $3, 4$ and $37, 38$) although the length of the sections are shorter than in the other schemas. In contrast, the RANDOM graphs do not have this property (see for example the erratic behavior of the speed at the beginning of the episode in the left column although the nearest neighbor is far away). In the graph in the right column (for the RANDOM experiment), the same situation appears. Note the lack of correlation between the speed and the distance around decision number 50 and in number 200. Therefore, there is a high difference at the microscopic level between a random policy and a learned policy (independently of the learning schema).

A study of the correlation between the pairs of speed and distance is also given in Table 6.7. In order to fit the size of the table, the names of the schemas have been abbreviated in all the tables. They correspond as follows:

- IT means ITVQQL.
- IN means INVQQL.
- TF_IT means ITVQQL with knowledge transfer.
- TF_IN means INVQQL with knowledge transfer.

The study assumes that a correlation factor greater than $+0.5$ means a significant correspondence between the two parameters. The scenarios are very different, varying with the number of agents, because the local interactions are different in situations with high or low densities (see the comments to the fundamental diagrams of Figure 6.10 in the next subsection). Therefore the data in the two rows of Table 6.7 are not directly comparable, because it is to be expected that high density interactions are more frequent with 90 agents than with only 18. For 18 agents, the percentage of episodes with a correlation greater than $+0.5$ is higher in the experiments with learning than in the RANDOM control, with higher percentages in the IT and TF_IT schemas with respect to IN and TF_IN schemas and VQQL. When 90 agents are used (second row), the highest correlation is also observed in IT and TF_IT schemas. Because the neighbors are closer to the agent than in low density situations (the distance to the nearest neighbor is often the minimum value possible), speed oscillations often occur

as explained before in the graphs in Figure 6.8 and Figure 6.9. This justifies a decrease in the correlation index between the two parameters in the data of the second row (90 agents). Note that the learned behaviors have correlation percentages significantly higher than the RANDOM control in both rows.

# Ag.	IT	IN	TF_IT	TF_IN	VQQL	RANDOM
18	77.3%	37.1%	79%	46.5%	45.7%	1.72%
90	48.8%	12.4%	48.6%	12.1%	41.8%	1.1%

Table 6.7: Percentage of episodes in which the selected speed and distance to the nearest neighbor of an agent have a correlation coefficient greater than +0.5. The data comes from the simulation of 100 episodes with 18 agents each (a total of 1800 episodes) in the first row and with 90 agents (a total of 9000 episodes) in the second row.

6.5.3 Macro-dynamics

This section gives the fundamental diagram and the density maps obtained in the experiments and their comparison with the Helbing model for the same configuration. The used implementation of the Helbing model is described in Helbing *et al.* (2000), although the panic component has been disabled to resemble these experiments. The fundamental diagram summarizes the micro-behavior recently set out, showing the relation between the speed and the density of all the agents involved. They are important for evaluating a pedestrian model. The left column of Figure 6.10 shows the fundamental diagram for all the schemas, specifically the average and the standard deviation of the speed values measured at different densities. The data was measured in a position in front of the door within a circular area of radius $1\,m$.

In all the experiments, the fundamental diagrams show that the velocity decreases when the density increases, which is a fundamental property of pedestrian dynamics. The range of velocities in the INVQQL and ITVQQL schemas are higher than in the VQQL schema. Specifically, with high densities the velocity decreases to 0.2 in the INVQQL schema and to less than 0.4 in the ITVQQL, whereas at high densities the velocity is 0.6 aproximately for the VQQL. This may indicate different dynamics of the VQQL with respect to the other two schemas

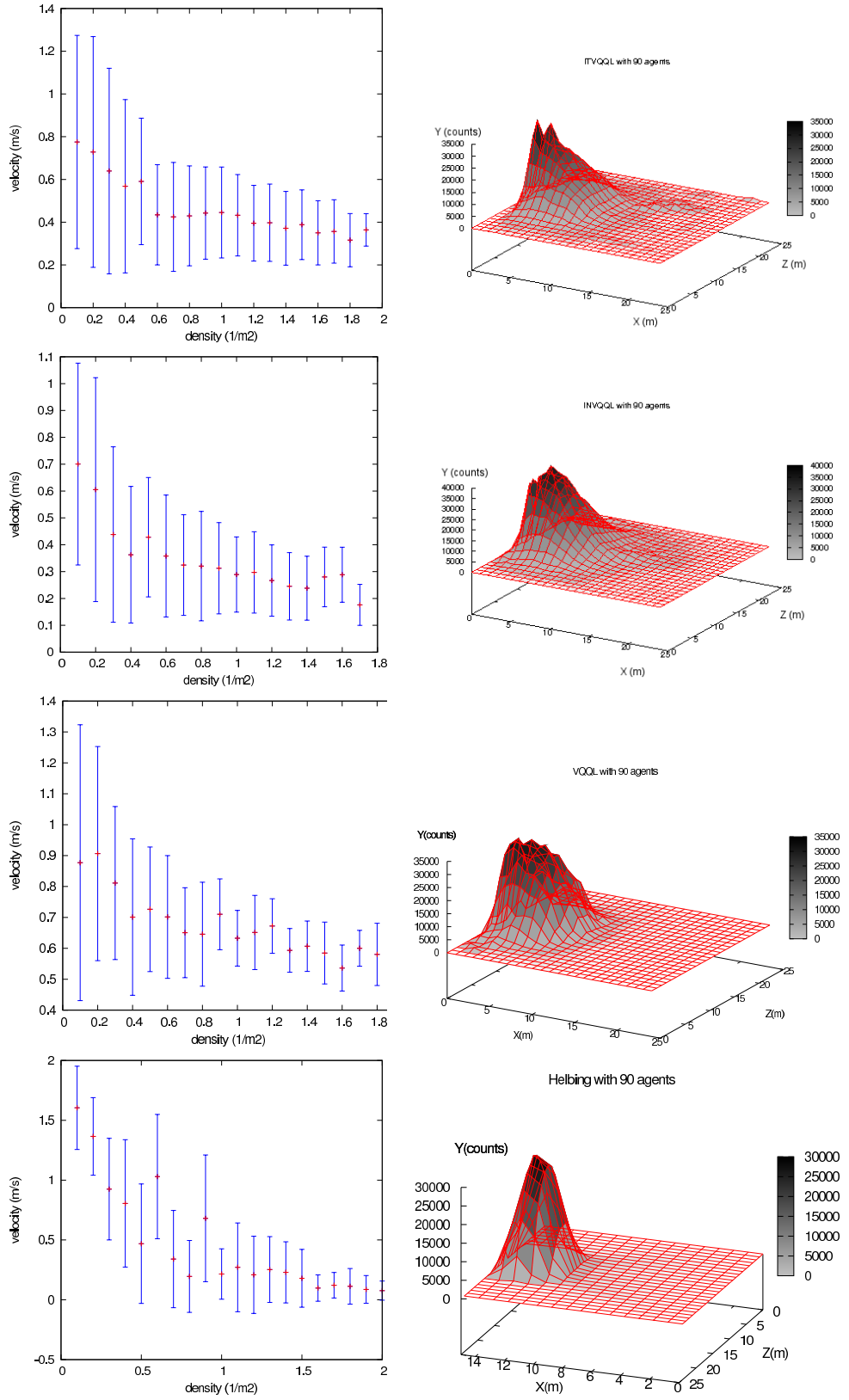


Figure 6.10: Fundamental diagrams and density maps. First row=ITVQQL, 2nd=INVQQL, 3rd=VQQL, 4th=Helbing model. The data are collected from 100 episodes with 90 agent each.

at high densities. Furthermore, in both, the schemas and VQQL, the standard deviation decreases with the density, indicating that the agents have learnt to choose small variations of velocity in high densities (that is, in a crowded situation). In a low density situation, an agent has more opportunity to modify the velocity vector, because collisions are less likely: therefore the standard deviation is high. This is also considered a positive characteristic in terms of the control of velocity. The Helbing experiment shows the same property: with increasing density, speed decreases.

The right column of Figure 6.10 shows the density maps in the selected configurations. A density map is a histogram that counts the number of agents occupying a spatial zone along the time of the experiment, and therefore reflects the use of this zone. In these graphs, the X and Z axes represent the spatial positions in the room. The square room has been divided by a 25×25 grid. The Y axis represents the occupation of the space in terms of the number of times that a unit square of the grid has been occupied in the 100 episodes of the experiment. The shell-shaped pattern in front of the door is a typical collective effect of clogging in the kind of bottlenecks studied in pedestrian dynamics (Helbing, 2004; Schadschneider *et al.*, 2008). Here, the bottleneck is clearly represented in the surface shape close to the door for all the experiments, including Helbing's experiment. Note the absence of significant heaps in other zones of the space, highlighting the absence of any wandering around the room. This is a common characteristic with Helbing's model.

With respect to the scalability, note that the mentioned characteristics of the results explained above are tested in an environment with 90 agents whereas the number of agents in the learning process was 18. This means that the approach is robust to scaling in the number of agents (note that the scaling is performed without additional learning, only sharing the learned value functions by the agents) and is capable of generalization in terms of the persistence of the learned dynamics. This statement is also corroborated by the results of Table 6.9 as discussed below.

A sequence of a simulation is presented in Figure 6.11. The agents learned using the ITVQQL schema in an environment with 18 agents. In the simulation, the number was scaled 5 times (90 agents).

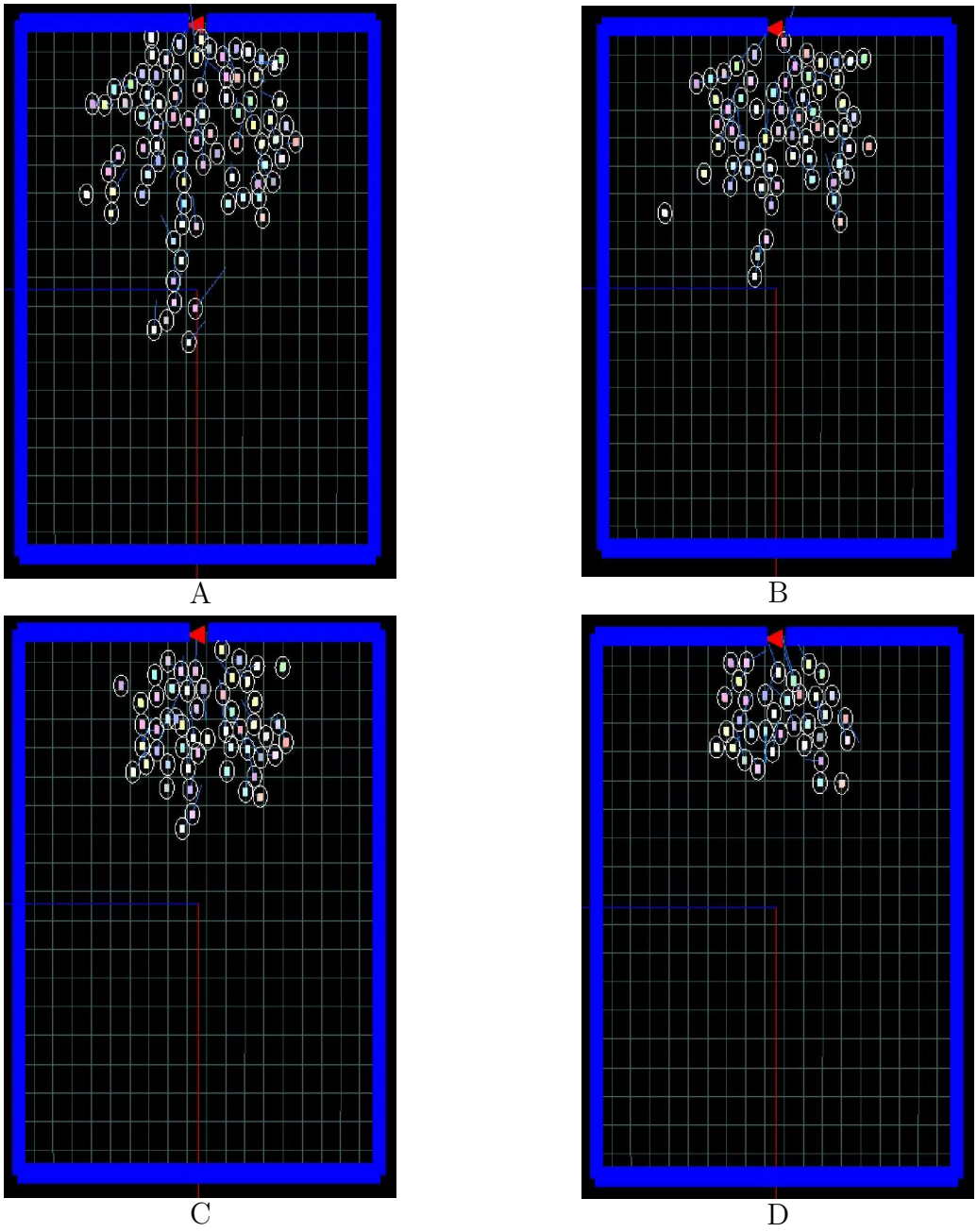


Figure 6.11: Four moments of a simulation for the closed room with the ITVQQL schema and 90 agents. The temporal sequence of stills is sorted alphabetically.

A similar sequence is represented using Unity rendering engine in Figure 6.12. Note the shell-shape of the group of agents in front of the exit, which minimizes

the distance to the goal.

6.5.4 Performance

The next tables characterize the different schemas with respect to the scalability in the number of agents. The learning tasks were performed with 18 agents.

Table 6.8 shows the length, in meters, of the agents' paths. The lengths are practically constant with the scaling (read the data by columns), suggesting that similar behaviors occur independently of the scaling factor. Specifically, the agents tend to avoid detours in their trajectories. This behavior is also observed in real pedestrians and has been reported in Helbing *et al.* (2001). There is little difference in the use or not of transfer for the ITVQQL and the INVQQL schemas.

#Ag.	IT	IN	TF_IT	TF_IN	VQQL
18	14 ± 7	20 ± 21	15 ± 7	18 ± 10	15 ± 11
36	13 ± 5	18 ± 15	15 ± 7	18 ± 8	15 ± 17
54	14 ± 6	18 ± 13	16 ± 8	19 ± 10	15 ± 11
72	15 ± 6	18 ± 12	16 ± 8	19 ± 11	15 ± 10
90	15 ± 7	18 ± 11	17 ± 9	20 ± 11	17 ± 10

Table 6.8: Averaged lengths and standard deviation for the paths in meters. The averages are over 100 episodes and for all the agents.

Table 6.9 displays the averaged number of agents who fail to exit by the door and its correspondent median. A statistical analysis was performed using the Statgraphics software package. A Kruskal-Wallis test using the medians was conducted to determine statistical significant differences among the experiments because neither the raw data nor the transformed data adjusted to a normal distribution. We can observe the following main results:

1. The means of all the schemas remain similar when a scaling with 54, 72 and 90 agents is performed. This fact suggest the possibility that the scalability results could be similar with a higher number of agents.

2. In the experiments with 18 agents (without scaling), the schemas with transfer are statistically different from the schemas without transfer and also from the VQQL. This shows that transfer has a significant impact when it is used

Figure 6.12: Rendered scenes of the simulation for the closed room with an exit experiment with the ITVQQL schema and 90 agents. The images are divided in two views of the same scene. The temporal sequence of stills is sorted alphabetically.

#Ag.	IT	IN	TF_IT	TF_IN	VQQL	P-value
18	1 b (2.4)	4 c (4.2)	0 a (1.0)	0 a (2.9)	1 b (2.8)	0
36	1 ab (6.5)	4 c (4.8)	0 a (2.8)	3.5 bc (6.2)	0 ab (6.8)	4×10^{-9}
54	1 a (6.4)	4 b (6.0)	0 a (3.6)	4 b (6.4)	10 ab (15.7)	7×10^{-10}
72	1 ab (9.4)	4 b (5.6)	1 a (3.9)	4 b (6.6)	4 b (22.1)	4×10^{-8}
90	1 ab (10.3)	4 b (6.0)	1 a (4.1)	3 b (6.5)	18.5 c (25.0)	4×10^{-10}

Table 6.9: Medians and means (in parenthesis) for the agents that do not reach the door (fails) when scaling up the number of agents. The means are averaged over 100 episodes (N=100) and are considered a measure of performance. Median values separated by different letters for the same number of agents (within a row), are significantly different ($P \leq 0.05$) according to Kruskal-Wallis test.

without scaling. When the number of agents is scaled up, the IT and the TF_IT are statistically classified as a different group with respect to the IN and TF_IN schemas. The VQQL is placed in a different group from the TF_IT schema in three of the five experiments.

3. The means in the VQQL are significantly higher than the others when scaling up the number of agents. Thus, it behaves the worst in all the experiments.

Reading by rows, the TF_IT schema has the best performance (lowest mean) in all the configurations. Note the positive effect of the transfer of knowledge in the TF_IT schema when the number of agents is scaled up, as opposed to the lack of such an effect in the TF_IN schema when scaling. However, if we compare the rows for 18 agents for both schemas with and without transfer, we can see that both improve the performance using transfer. This agrees with the performance results for 18 agents in the learning processes (see Figure 6.4 and Figure 6.5).

From previous evaluations one can draw several conclusions:

1. The analysis of the graphs of speed and distance to the nearest neighbors, which are related to the microscopic behavior of the agents, shows that all the schemas provide behaviors with a correlation between the curves. The observation of individual episodes also reveals that the controller tends to

remain unchanged when the distance to the nearest neighbor is large, which is considered to be a rational behavior.

2. The fundamental diagrams and density maps reveal that the main characteristics of the pedestrian dynamics and collective behavior appear in all the schemas. These tests, which are mainly related with the macroscopic (collective) behavior, do not exhibit any significant differences between the two schemas.

3. The scalability tests displayed in the tables show properties of real pedestrian behavior and demonstrate empirically that the learned behaviors are generalizable to other configurations with more agents.

4. The results shown in Table 6.9 indicate that the VQQL baseline algorithm performs worse (highest mean) than the rest of the schemas when scaling up the number of agents. Also it shows that the TF_IT has the best performance (lowest mean) in all the experiments with a different number of agents.

6.5.5 Macroscopic analysis for the second scenario

In the crossing scenario I am interested in studying the emergence of collective behaviors. Specifically the formation of lanes as the emergent behavior that solves the crossing. Our scenario is $2\,m$ wide, therefore, only three agents are necessary to obstruct the corridor. Without an organization of the groups, clogging is a difficult problem to solve.

The solution found by the agents in the learning schemas (TF_IT, TF_IN) and in VQQL, has the same structure. The agents that belong to a group form one line in anticipation of the moment of crossing as can be seen in the sequence of images in Figure 6.13. The reader can see some videos of the results at URL `http://www.uv.es/agentes/RL/index.htm`.

In this scenario, the performance is measured considering whether all members of the group have reached to the goal. Thus, only the episodes in which the crossing has been solved successfully are taken into account. This performance measure is more restrictive than that used in the first scenario, where how many

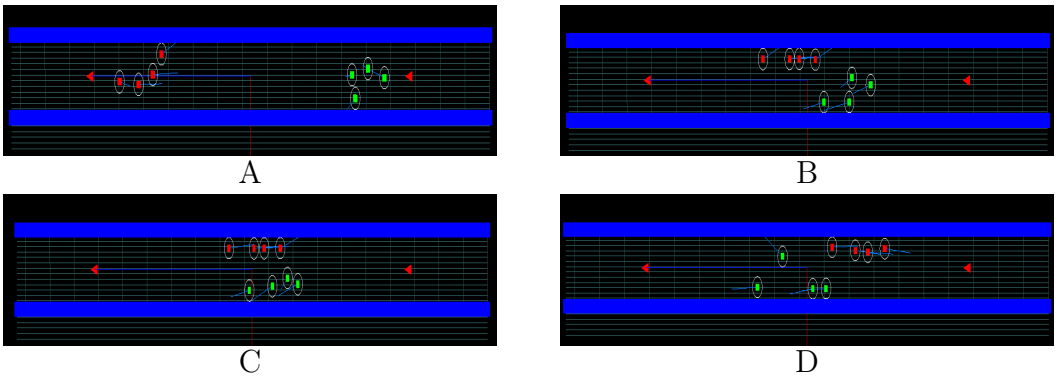


Figure 6.13: Four moments of a simulation from crossing scenario with TF_IT schema. The temporal sequence of stills is sorted alphabetically.

agents reached to the goal is measured. The results in Table 6.10 indicate that not all the episodes solve the crossing. Although the lanes appear in all the episodes, the lane formation needs to anticipate the instant of the crossing to avoid clogging that the agents cannot solve with the limited time given to each episode (80 decisions, that is, 40 seconds in simulation time). The TF_IT schema has the highest performance with significant statistical difference with respect to the TF_IN schema and the VQQL baseline.

In Figure 6.14, the density maps are displayed. As in the first scenario, the maps are built accumulating the positions of the agents in 100 episodes. The flat zones at the borders are the space occupied by the walls. Note the two high density zones near the walls that correspond to the lane formations. The high values of the histogram in those zones reveals that lane formation occurs often (actually in all the episodes) and with a similar structure in all the schemas (including the VQQL algorithm). This fact together with the results shown in Table 6.10 reveal that lane formation is not sufficient by itself to solve the clogging, it is also necessary for the lanes to appear before the crossing. In order to emphasize the differences between the schemas, a side view of the map is included in the right column of Figure 6.14. In the side view, the shape of the middle region of the corridor in terms of densities can be seen. It can be observed that the VQQL experiment has the highest density in the middle zone of the corridor, meaning the existence of cloggings in this zone. The TF_IT schema has the lowest density in the middle of the corridor, suggesting that the anticipation in lane formation

TF_IT	TF_IN	VQQL	P-value
81 a	52 b	63 c	0.0000

Table 6.10: Mean of the number of episodes that end successfully from a series of 100 episodes. A successful episode means that all the agents reach the corresponding goal. Ten series have been carried out. The data was analyzed with an ANOVA test. It proved that the means are significantly different ($P \leq 0.05$). The letters classify the different groups according to Duncan's multiple range test.

has been learned better.

From the results gathered we extract the following conclusions for the second scenario:

1. The emergence of the lanes occurs in all the learning approaches evaluated. In all the cases, the lanes have a similar structure.

2. The TF_IT performs better than the TF_IN schema and the VQQL baseline experiment. The ANOVA test shows significant differences in the results.

6.6 Conclusions of the experiments

Two different algorithmic schemas have been designed to increase the performance of the VQQL algorithm. Each one proposes a different strategy to address the problem of achieving an adequate description of the state space for the task as well as how the problem of multi-agent learning is dealt with. In addition, different techniques of knowledge transfer have been used to accelerate the learning process. In both scenarios, ITVQQL with transfer (TF_IT) performs better than the others for the studied configurations. Specifically, it gives the best performance in the first scenario when scaling up the number of agents as well as in the second scenario in the number of solved crossings. Although the studied scenarios in this chapter are different enough to indicate the broad usability of the ITVQQL with transfer algorithm, it is especially useful in domains in which the random exploration of the state space does not lead to quantizers with a good enough representation of the state space to find the optimal policy.

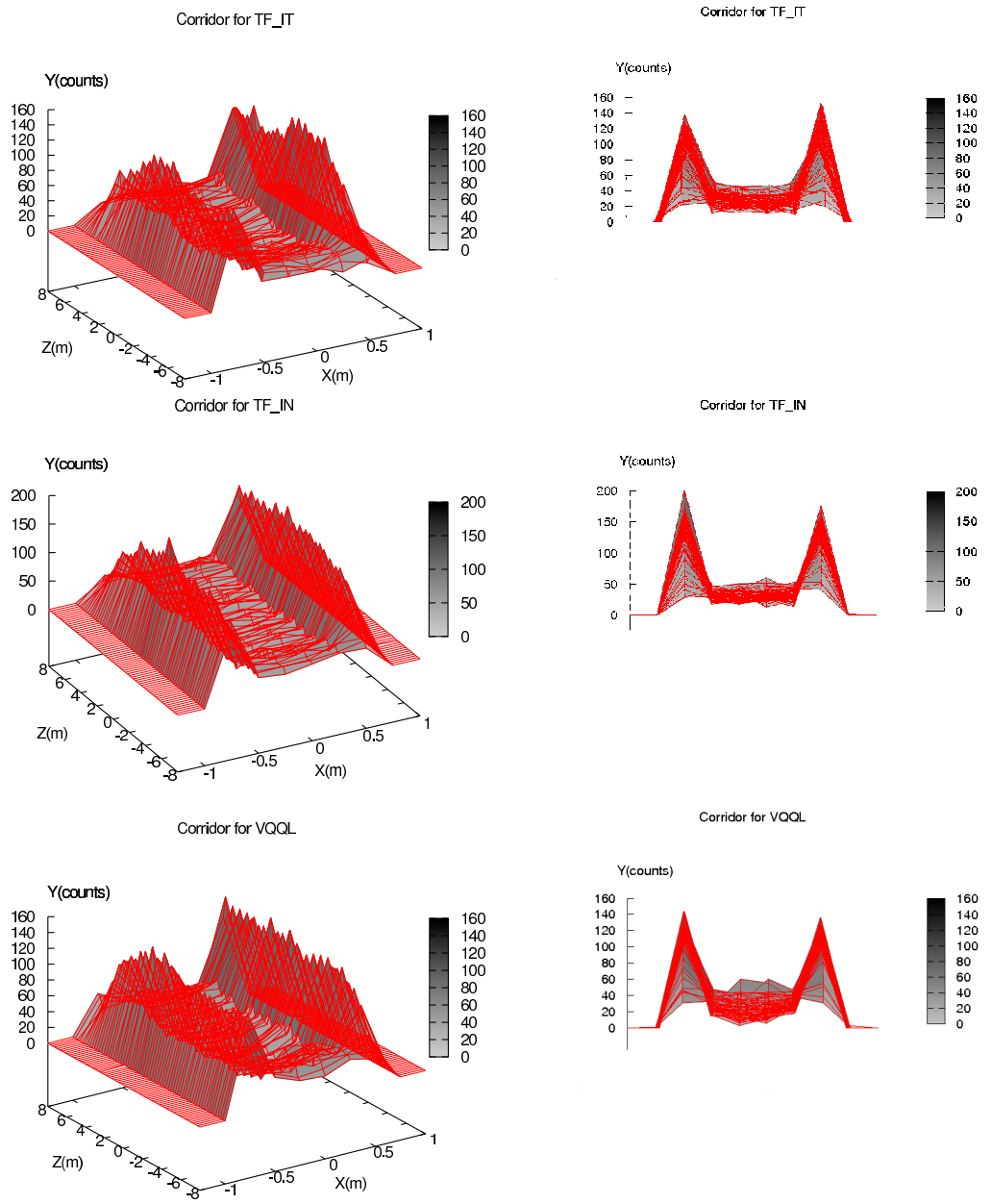


Figure 6.14: Density maps of the schemas for the crossing scenario. The data come from 100 simulations. The high density areas at the sides are created by the emergent lanes. The right column shows a side view of the same map to compare the density of the middle region of the corridor. Note that the graphics have not the same scale.

From a macro-dynamics point of view, the speed-density relation in the fundamental diagram and the emergence of a basic collective behaviors (lane formation in the corridor and shell-shaped organization in the closed room scenario) result as a consequence of the policies learned. Specifically, the collective behaviors indicate a conformance with real pedestrian studies and other pedestrian models.

For the first scenario and from a micro-dynamics perspective, the speed controller seems to be stable in high density situations (where many local interactions occur). The learned behaviors in the first scenario are robust to scalability in the number of agents. Furthermore, the comparison at the macroscopic level with the Helbing pedestrian model of social forces shows similarities with our results in terms of the fundamental diagrams and the density maps in the first scenario. These similarities with the Helbing model support the idea that our agents have developed plausible behaviors of pedestrians.

6.7 Chapter highlights

- Two schemas based in the VQQL algorithm are presented: ITVQQL and INVQQL, which adopt different strategies of learning. ITVQQL learns the same problem with a fixed number of agents in each of its iterations tranferring the value function and improving the VQ in each iteration. INVQQL learns the problem by gradually incrementing the number of agents in each iteration. Therefore, the learning problem increases the difficulty in each iteration.

- The schemas are tested in two scenarios (a closed room with an exit and a crossing inside a corridor). The dynamics analyzed in the behaviors obtained for the room scenario indicate pedestrian-like responses at both the macro and micro levels. The results for the corridor scenario indicate the emergence of collective behaviors.

Chapter 7

Learning approaches based on Sarsa(λ) with tile coding.

Following the objective of this thesis to demonstrate the utility of RL for pedestrian simulation, in this chapter another different configuration including different scenarios will be tested. Specifically, Sarsa(λ) as the learning algorithm and tile coding as the state generalization system will be used. The reader can find a discussion of the algorithms and methods in Section 3.3.2, and a description of the basic algorithm in Algorithm 7. Two important differences with respect to the algorithms used in the previous chapter appear: i) Sarsa(λ) is an on-policy algorithm while Q-learning is off-policy. This property is related with the fact of using, or not, a separate exploratory policy from the learning policy and was discussed in Section 3.1.4. ii) Tile coding is a linear function approximator that assigns the specific state to a codified region of the state space, contrary to VQ, which is a state-aggregation-based method.

The studied scenarios are also different with the exception of the crossing of two groups of pedestrians inside a narrow corridor, also considered in the previous chapter. The new scenarios are; i) choice of the shortest path *vs.* quickest path; ii) two agents who move in opposite directions inside a maze. These scenarios are included to demonstrate that the learned behaviors are not only capable of controlling the local interactions but they generate higher levels of behavior, specifically, tactical behaviors. Thus, the learned behaviors adapt to the level

required by the pedestrian scenario. The shortest path *vs.* quickest path, presents a route choice problem where the agents can select the quickest path, making a detour, and avoiding a bottleneck that represents the shortest path option. The maze represents a classical planning problem where avoiding local minima situations is necessary to reach the goal.

The narrow corridor scenario, studied in the previous chapter, will be revisited. The reason for repeating this experiment is double: first, I will demonstrate that the emergence of the expected collective behavior also appears with this configuration (or, in other words, it is configuration-independent). Second, I will use it as a comparison testbed. Specifically, I will compare the performance of the two approaches (ITVQQL *vs.* Sarsa with tile coding) and study the influence of knowledge transfer techniques on it.

I will also compare the shortest path *vs.* quickest path and the narrow corridor experiments with the Helbing model in similar microscopic conditions. As in the previous chapter, this comparison reveals the level of similarity of the dynamics developed by the learned behaviors with those generated with Helbing's model.

7.1 Modeling the problems

In this section the proposed problems are modeled as MDPs.

7.1.1 The scenarios

Firstly, the scenarios of the experiments are introduced. These scenarios model common situations for real pedestrians in urban environments.

The shortest path *vs.* the quickest path scenario

In the problem of the shortest path *versus* the quickest path, an agent has to choose between two exits to reach the goal. One exit is near the goal and the other is situated farther from it. If the number of agents is large, a jam is generated in front of the exit that is next to the goal. However, another alternative path is available which detours the group using the other exit. Assuming that the extra effort to perform the detour is small, some of the agents at the borders of the

jam can decide to follow the detour instead of remaining inefficiently waiting. This problem happens in different situations in real life (for example pedestrians hustling through a station hall as they are late for a train) and it differentiates pedestrian dynamics from other vehicle dynamics (Johansson & Kretz, 2012).

The layout of the environment in this experiment is a wall with two exits that divides a space in two areas. The agents are placed in the first area with the following dimensions, width $18\,m$ and depth $6\,m$, and the goal to reach is placed in front of one exit in the second area. The exits are $1\,m$ wide, allowing only one agent at a time to traverse them. The two exits are separated by distance of $6\,m$. The distances from the nearest exit and from the farthest exit to the goal are $1.5\,m$ and $6.2\,m$, respectively (see Figure 7.1).

The crossing inside a corridor scenario

The crossing inside a narrow corridor scenario has already been explained in Section 6.1.1. In this case, the same configuration is adopted. An image of the scenario's layout, as well as a rendered image of a simulation can be seen in Figure 7.1.

The maze scenario

Real urban pedestrians find mazes in several common situations. In a congested avenue, vehicles create mazes that pedestrians have to solve to cross the road. In certain railway crossings, pedestrians find fences which build mazes that require them to approach the crossing by turning left and right in order to force the visual detection of the presence of a train. To leave the maze, the virtual embodied agent needs to plan the trajectory. A greedy strategy, like that in which the agent always chooses the straight path, can leave him/her stuck in a local minimum. Path-planning is a necessary task in crowd simulation commonly placed in the high level agent's behavior model (Pelechano *et al.*, 2007) or in the environment (Treuille *et al.*, 2006). In our approach, path planning[1] is a consequence of the RL dynamics and is intrinsic to the learned controller. The exploration policy intrinsically

[1]In this work, path planning should be understood as a path finding capability. This approach is far from traditional planners which provide optimality and completeness guarantees.

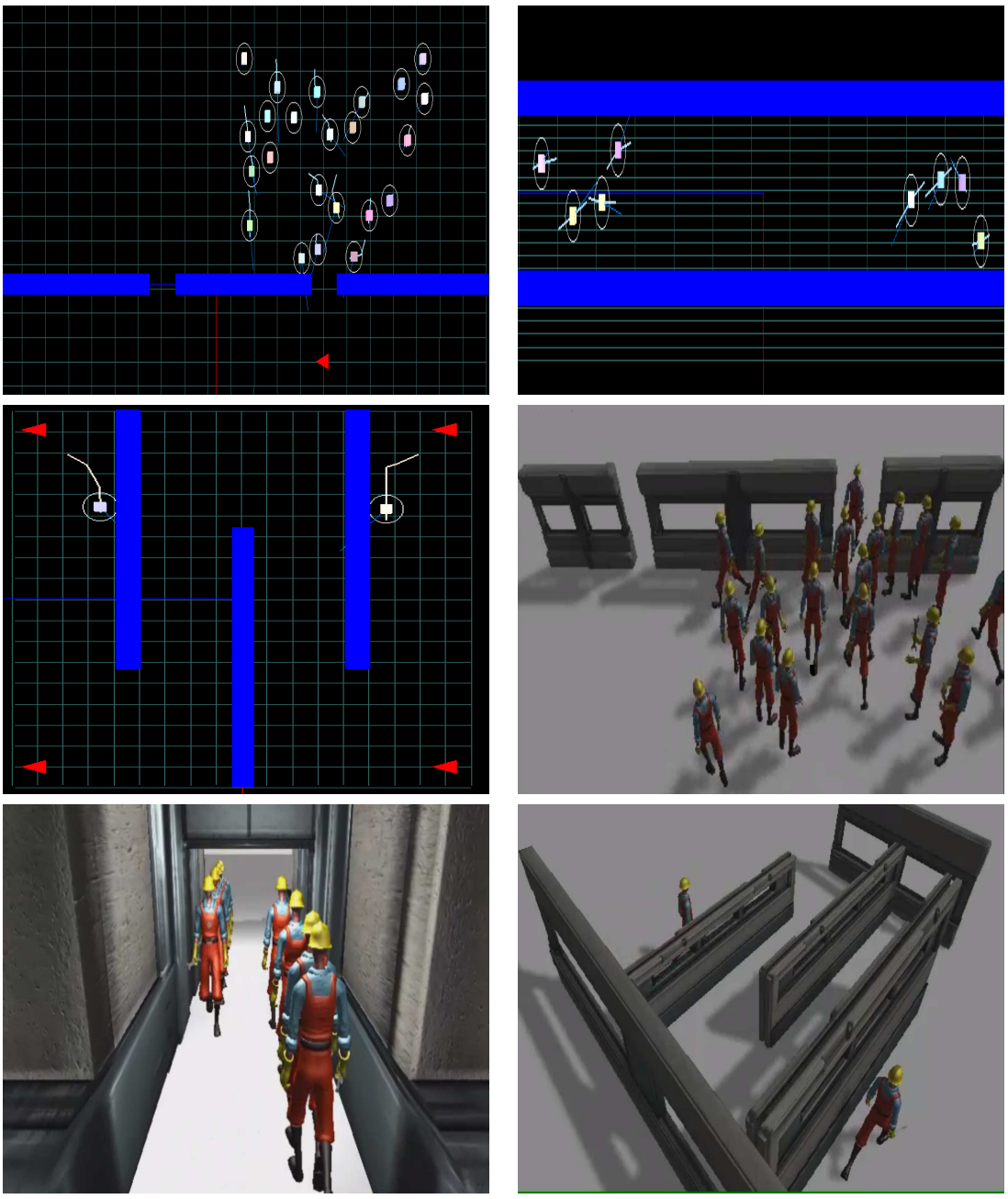


Figure 7.1: Images of the scenarios for the experiments. The first three images correspond to the shortest path *vs* quickest path, the corridor and the maze experimental scenarios. The last three images are stills of simulations correspondent to these experiments. These simulations are renderized with Unity.

assumes the task of planning when forcing the agent to explore different actions in a given state. In this experiment, the virtual environment is a square $4 \times 4\,m$ in which three walls have been appropriately situated to create a maze. Two agents are situated at the beginning of the episode in two adjacent corners of the square, and their goals are placed in the diagonally opposite corners. The agents move across the maze in opposite directions (see Figure 7.1).

7.1.2 State space definition and generalization

The number of features that describe the state space for each experiment is different. In the shortest path *vs.* quickest path, each agent senses 7 neighbors and two objects (the two nearest walls). In the corridor scenario, each agent senses 4 neighbors and two objects. In the maze scenario each agent only senses the other agent and the two nearest walls. When an agent cannot detect the necessary neighbors to complete the perception, (as discussed in Section 6.3), the problem is solved by setting unobserved features to random values (random imputation).

The tile coding generalization system also needs to be adjusted in the number of tilings and the number of partitions of each dimension. Each scenario needs its own study. The selected values are indicated in the parameter tables for each experiment in the corresponding section. As an example of this value-selection work, a study of the number of tilings for the shortest path *vs.* quickest path with 28 agents is displayed in Figure 7.2. As in the case of the selection of the number of prototypes in the VQ method, a trade-off is necessary between the number of tilings and the computational efficiency. In this case, the comparatively high performance achieved by 128 tilings justifies its use in the experiments.

7.2 Learning results

In this section, the configuration, as well as the performance achieved by each learning process, is described. There is not a fixed pattern to define the configuration of parameters and the strategies to be used in the learning process because each scenario has its own challenges that have to be specifically addressed.

The curves that describe the learning results in the following experiments,

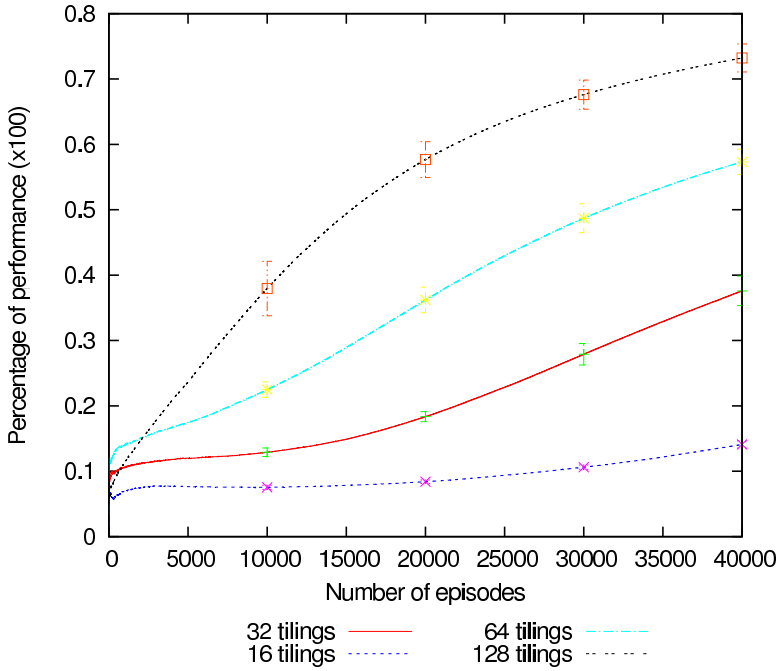


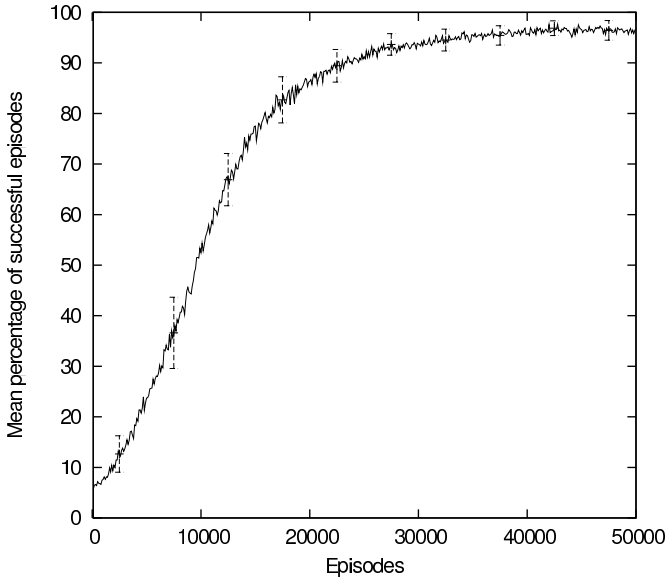
Figure 7.2: Influence of the number of tilings on performance in 'shortest *vs.* quickest path' scenario. The curves are means of 28 agents.

represent the mean curves of the performance values of the last 100 episodes.

7.2.1 Shortest *vs.* Quickest path

On the right of Figure 7.3, the main parameters of the learning process for this scenario are specified. The learning phase has 23 agents, that is a sufficient number to produce the bottleneck in front of the nearest exit to the goal. The learning process has been empirically fixed to a duration of 50000 episodes.

The left of Figure 7.3 shows the learning curve of the mean percentage of success of 23 agents. An agent successfully solves an episode when he/she reaches the goal, independently of the chosen path. As indicated previously, each point in this curve represents the mean percentage of success for the last 100 episodes. The final asymptotic region of the curve, with a percentage of success of more than 90%, indicates that the agent has learned to solve the problem.



Key	Values
Number of learning agents	23
Number of episodes	50000
Parameter λ of Sarsa Algorithm	0.9
Learning rate	0.001
Influence of future rewards (γ)	0.9
Initial value of Epsilon (ϵ_0)	0.9
Number of tilings in Tilecoding	128
Reward when leaving the valid area	-10
Reward when the goal is reached	+100

Figure 7.3: Learning process for 'shortest *vs.* quickest path' experiment. Top: Mean percentage of successful individual episodes. Curve shows the mean and standard deviation for 23 agents. Bottom: Configuration values.

7.2.2 Crossing in a corridor

The crossing scenario is a problem of spatial organization in which anticipatory maneuvers of the pedestrians in each group can play an important role in solving the problem. A way to facilitate the search for a solution is to give useful information during the learning process. We have used Policy-Reuse (PPR) to introduce information in the learning process (see Section 3.4 for a description of the method, and specifically Equation 3.25). In this case, the existing policy

π_{past} consists of using actions that moves the agent towards the right side of the corridor. This policy was also used in the twin experiment in Chapter 6. The idea is to set similar learning conditions to make the results comparable. Hence, the decay pattern for PPR is the same as that used in the ITVQQL experiment with the corridor in the previous chapter. The specific values of the learning parameters for the crossing experiment as well as the probabilistic curves used in PPR are displayed in Figure 7.4.

The averaged percentage of successful episodes for the 8 agents with and without using Policy-Reuse is shown in Figure 7.5. The use of PPR has a positive effect on the learning processes as the agents learn faster. For example, at episode 10000, the percentage of success with PPR is about 75%, whereas in the same episode without PPR it is about 40%.

Key	Values
Number of learning agents	8
Number of episodes	50000
Parameter λ of Sarsa Algorithm	0.9
Learning rate	0.004
Influence of future rewards (γ)	0.9
Initial value of Epsilon (ϵ_0)	1.0
Number of tilings in Tile coding	64
Reward when the goal is reached	+100

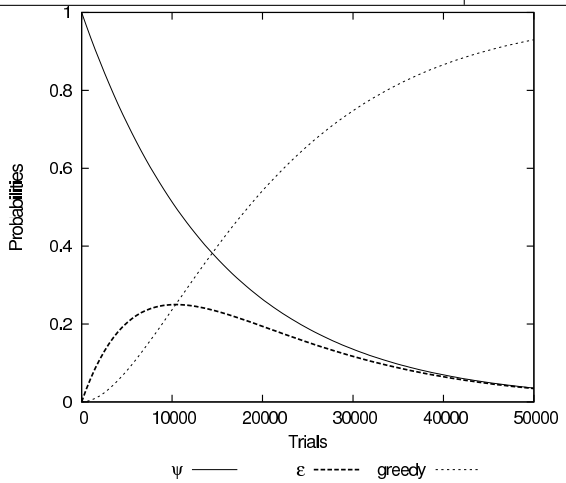


Figure 7.4: Learning configuration in the 'crossing inside a corridor' experiment. Top: Values for the main parameters of learning processes. Bottom: Probability functions for Policy-Reuse transfer method.

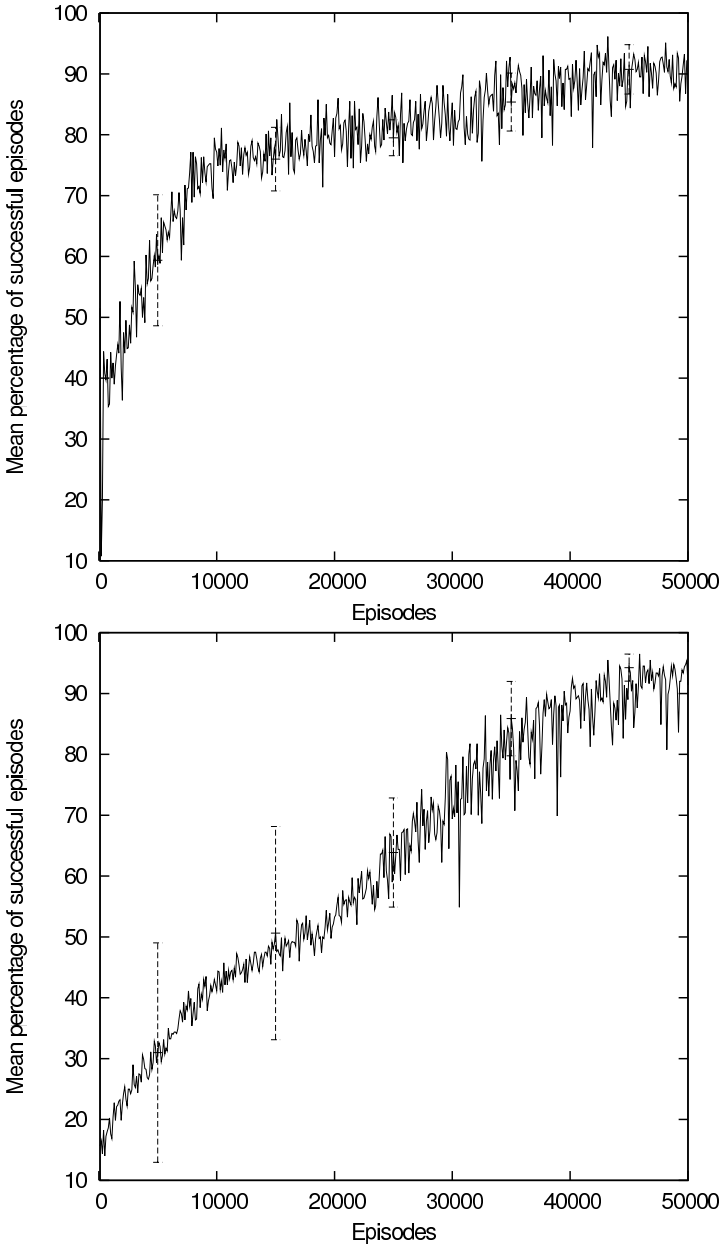
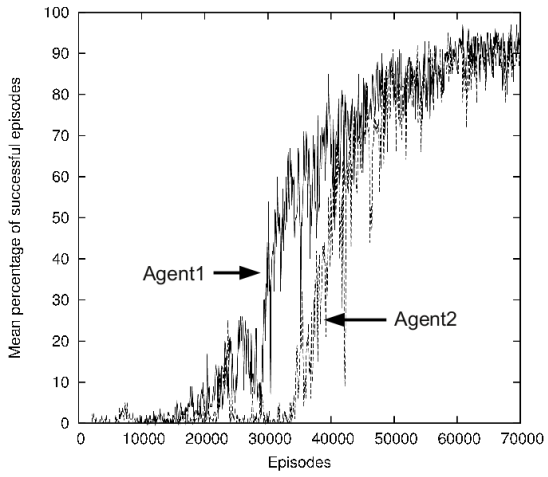


Figure 7.5: Learning performance in the 'crossing inside a corridor' experiment. Top: Mean percentage and standard deviation of successful individual episodes using Policy-Reuse. Down: Mean percentage and standard deviation of successful individual episodes without Policy-Reuse. The means are calculated with the data of 8 agents.

7.2.3 Pedestrians in a maze

The right of Figure 7.6, shows the table with the values of the parameters that make up the experiment. To the left of Figure 7.6 the percentage of successful episodes for each agent is shown. The curves show each agent's independent learning process. The agents do not learn at a similar pace throughout the process. This asynchrony derives from the fact that the goal is discovered at different times by each agent. As each agent perceives the other agent as part of the environment, a fast improvement in the policy of an agent creates a non-stationary environment for the other agent, who has more difficulties learning. A careful adjustment of the learning rate (the α parameter) alleviates the problem. Despite this fact, the percentage of success for both agents at 50000 episodes is about 90%, which indicates that the problem has been solved.



Number of learning agents	2
Number of episodes	70000
Parameter λ of Sarsa Algorithm	0.9
Learning rate	0.007
Influence of future rewards (γ)	0.9
Initial value of Epsilon (ϵ_0)	0.3
Number of tilings in Tilecoding	32
Reward when leaving the valid area	-5.0
Reward when the goal is reached	+100

Figure 7.6: Learning results for the maze. Top: mean percentage curves of successful episodes for each agent. Bottom: values of the parameters.

7.3 Simulation results

In this section, we analyze the performance of the framework with the specific configuration on the described scenarios. The results are illustrated with videos that can be seen at URL `http://www.uv.es/agentes/RL`.

7.3.1 Shortest path *vs.* quickest path

In Figure 7.7, the density map created by the learned behaviors is displayed. The two flows of pedestrians who go through the exits towards the goal are visible. Note in the perspective view, the clogging created around the exit of the shortest path (similar to a mountain with two peak) and the flow that deviates towards the quickest path represented by the shaded area between the values Z=-4 and Z=2. The flat areas at X=0 correspond to the walls (where occupancy of the space is not possible). The flow of agents that use the quickest exit surrounds the longest flat area. In the side view, the ridge that connects the two heaps that represent the exits, also shows the trace of the agents that use the quickest path.

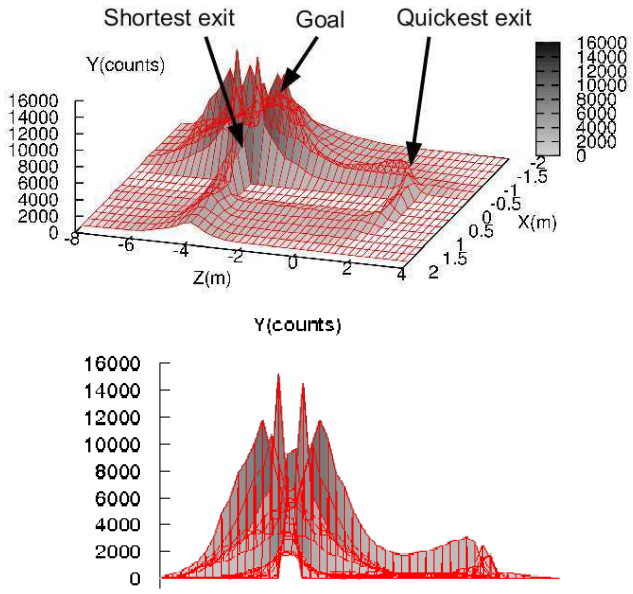


Figure 7.7: Density map for the 'shortest *vs.* quickest path' experiment with 23 agents. Perspective and front views. Data are from 100 simulations.

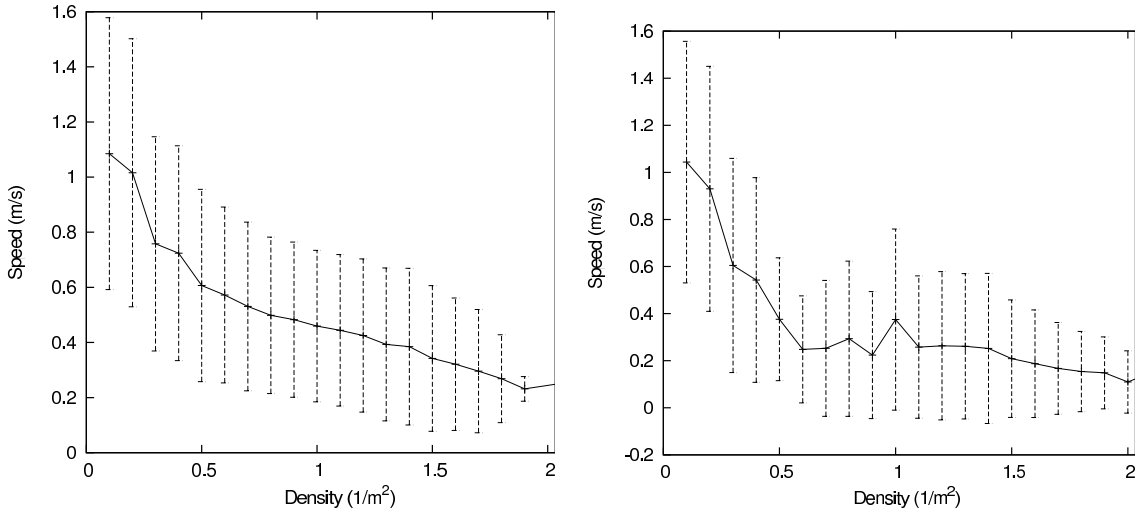


Figure 7.8: Fundamental diagrams for a clogging in front of an exit with 23 agents. Left: 'Shortest *vs.* quickest path' experiment measured in front of the quickest path exit. Right: Helbing's model for a similar clogging in front of an exit.

In Figure 7.8, the fundamental diagrams for Helbing's model and the framework corresponding to a clogging in front of an exit, are displayed. The data in Helbing's curve comes from the experiment of a closed room with an exit described in the work by Helbing *et al.* (2000) with the code available at URL://http.-pedsim.elte.hu. Although the set up of the experiment is not the same, the measured points for both diagrams are placed in front of the exit in which clogging is created (in Helbing's experiment there is only one exit, in our experiment this corresponds with the shortest path exit). The diagrams compared show common characteristics: first, a decreasing shape of the curves when the density increases, which is an important characteristic of pedestrian dynamics. The decrease in speed at high densities is coherent with a state of congestion. Second, the standard deviations of both curves decreases with increasing density which indicates the restriction of the choice of possible velocities in a state of high density. The curves shapes are similar, indicating comparable dynamics.

Table 7.1 summarizes the different performance obtained with 1000 episodes. In this table, individual average performance percentage (percentage of times that an agent reaches the goal) and collective average performance percentage (average

Individual averaged performance percentage	98.6 ± 0.6
Collective averaged performance percentage	72.3 ± 4.5
Averaged outputs through the shortest path (23 agents)	17.0 ± 2.0 (73.9%)
Average outputs through the quickest path (23 agents)	4.0 ± 1.6 (17.4%)

Table 7.1: Performance analysis in simulation for the 'shortest *vs* quickest path' experiment. Mean and standard deviation. Individual averaged performance percentage counts the times (of 100 episodes) that an agent has reached to the goal (independently of the path). Collective averaged performance measures the times (of 100 episodes) in which all agents arrived at the goal (independently of the path choice). Values are averages of 10 experiments. For averaged outputs, the whole set of 1000 episodes with 23 agents each have been used.

percentage of episodes in which all agents reach the goal) are shown. While the individual percentage gives a measure of the quality of the individual learned behaviors, the collective percentage indicates the percentage of valid simulations (in which all agents reach the goal). The figures in the table show that more than 98% of the agents solve the problem and the 72.3% of the simulations end with all the agents finding the goal. In addition, 17.4% of the agents were able to find the alternative route to the shortest path.

In Figure 7.9, a temporal sequence of a simulation with 23 pedestrians is displayed. The high density in front of the exit corresponding to the shortest path, generates a detour of some peripheral pedestrians towards the quickest path. Only the agents situated on the left side of the clogging select the detour, as expected. Note in images B and C of the sequence how two agents situated at the left border of the clogging use their quickest path selecting the farthest exit from the goal. When clogging disappears, the agents again choose the shortest path. This situation can be seen with the pair of agents to the right of the shortest path exit in image D of the sequence. Note how they disappear in image E indicating that they used this exit to reach the goal.

In Figure 7.10, a sequence of stills from a simulation renderized with Unity is displayed.

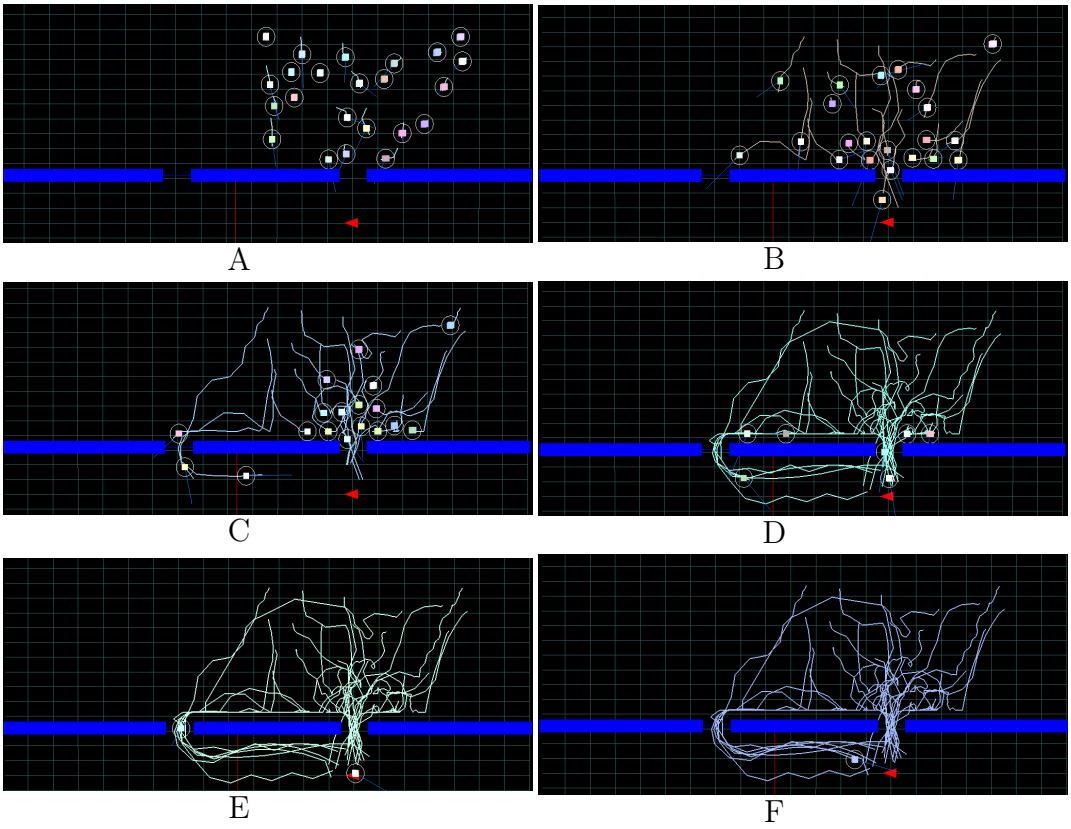


Figure 7.9: Sequence of stills from a simulation in the 'shortest *vs.* quickest path' experiment. The lines show the trajectories of agents. The temporal sequence is sorted alphabetically.

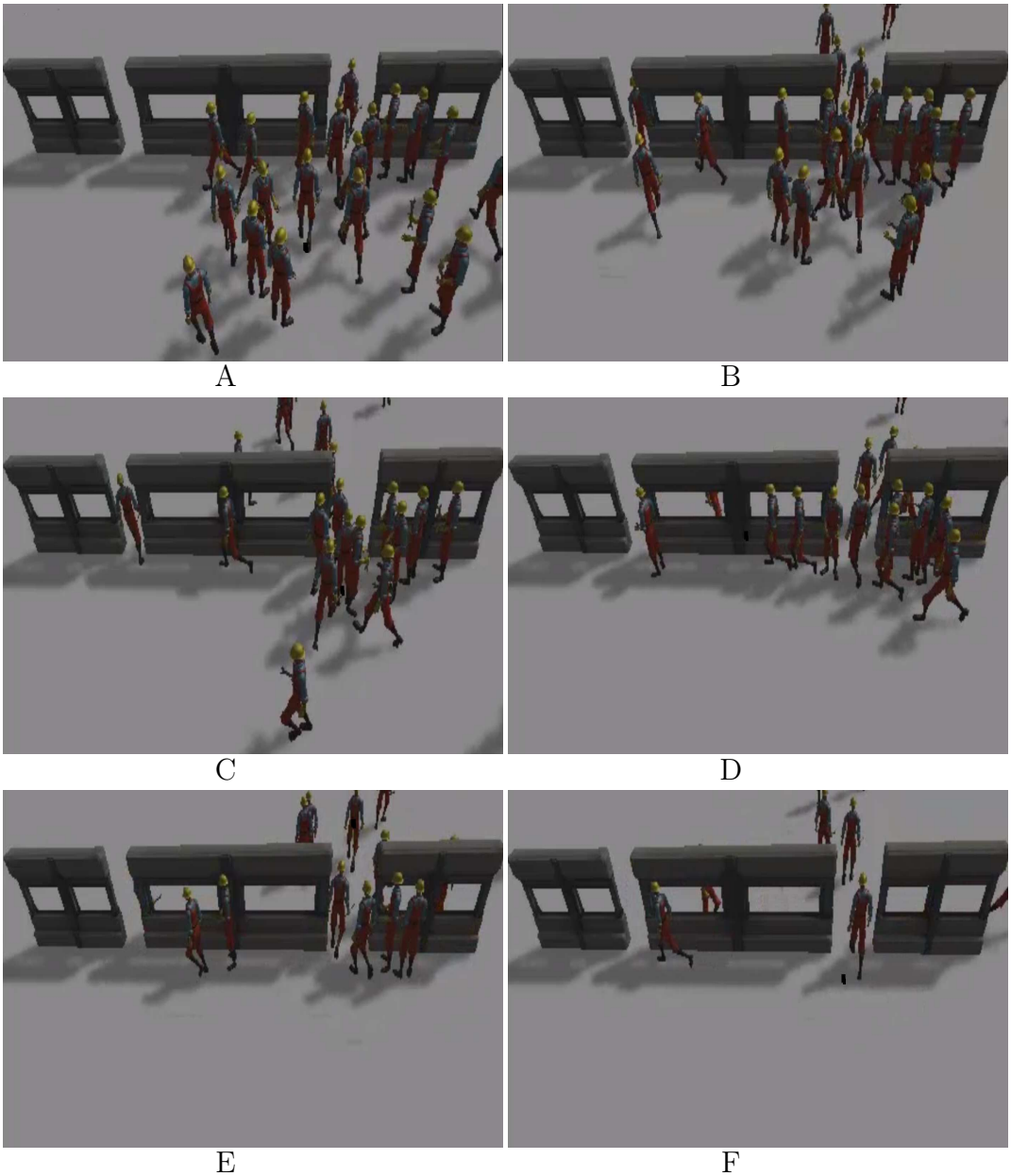


Figure 7.10: A sequence of rendered images of the 'shortest *vs.* quickest path' experiment. Each image presents two different views of the same instant. The temporal sequence of stills is sorted alphabetically.

7.3.2 Crossing in a corridor

In this scenario, a comparison of the results obtained using the developed framework (with and without policy-reuse) and using Helbing's model is performed. The density maps are displayed in Figure 7.11. The density map generated by the framework with PPR shows two high density areas that correspond to the lanes. As can be seen in the first row of Figure 7.11, the lanes appear near the walls of the corridor represented in the density map by two flat zones at both sides. On the contrary, the center of the corridor has a low occupancy. In the second row, the map for the experiment without PPR is displayed. The lanes also appear at both sides of the corridor, near the flat zones that indicate the presence of walls. The occupancy of the center is higher than that obtained in the previous experiment. Moreover, the occupancy of the sides is not as clear as before. These facts indicate that the agents do not create the lanes with the same anticipation and decision than in the experiment with PPR (the agents use the central area of the corridor more often). This is also supported by the results showed in Table 7.2. The third row corresponds to the Helbing experiment. The density map reveals that the space occupancy is different. The two groups of agents try to get to the center of the corridor instead of deviating towards the walls as occurs in my framework. This is an expected behavior because the presence of the walls generate repulsive forces that deviate the agents towards the center. When the individuals from both groups meet half way to the goals, the repulsive forces between the agents allow them to approximate to the walls solving the crossing. Therefore, large occupancy is visible on the map in the center of the corridor where the two groups of agents meet. The presence of lines in Helbing's model is shaded by the central interaction zone.

Figure 7.12 shows two fundamental diagrams corresponding to the framework of the crossing experiment using PPR (left) and Helbing's model (right). The measurement point has been chosen in the center of the corridor with a radius that covers the whole width of the corridor in both experiments. The range of densities in Helbing's diagram cuts with a density of $0.9\,1/m^2$. This indicates that the crossings do not create congested situations. In our framework, the densities are higher than in Helbing's experiment indicating more congested situations. On

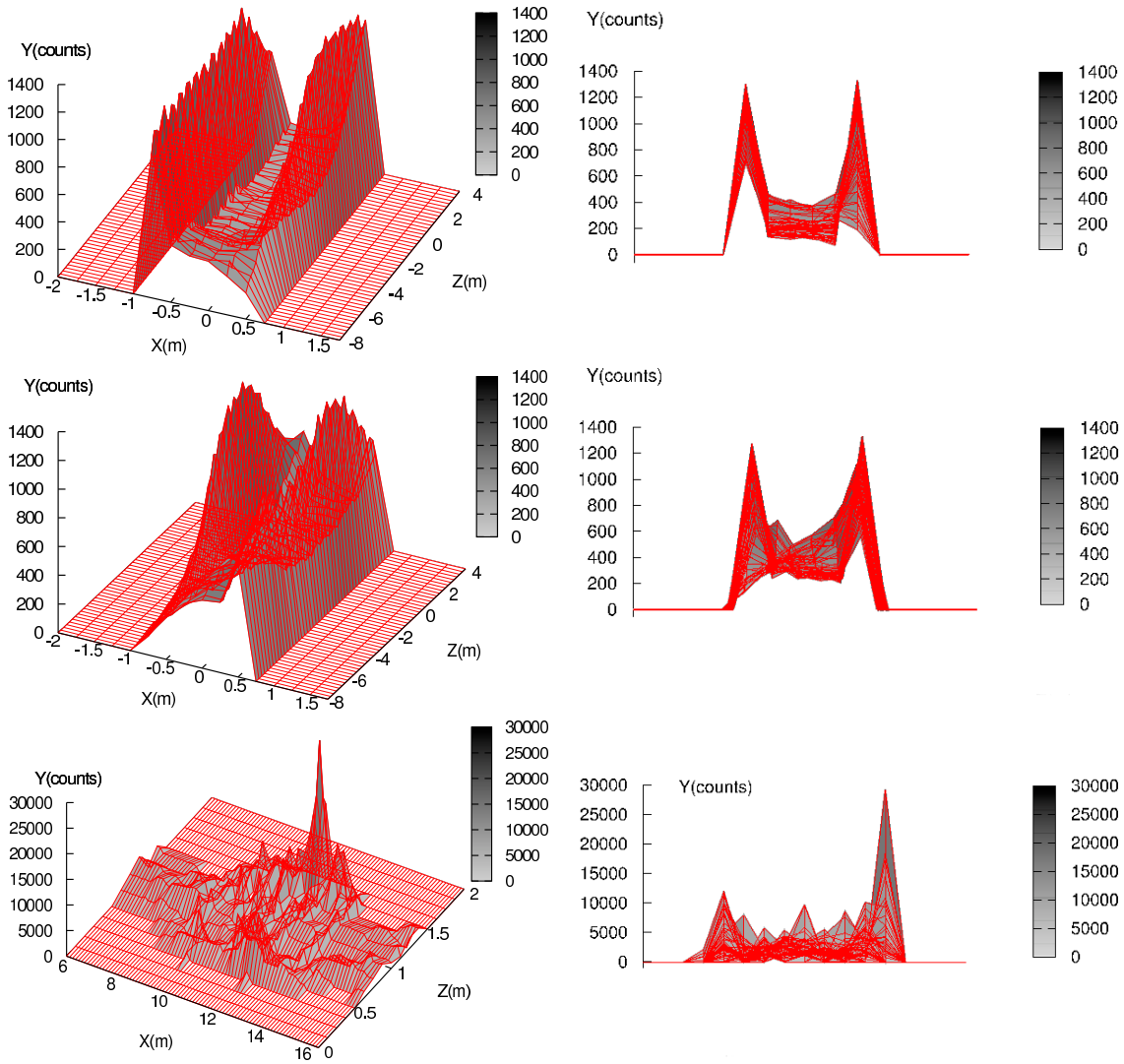


Figure 7.11: Perspective and front views of density maps for the 'crossing inside a corridor' experiment. With PPR (first row). Without PPR (middle row) and experiment with Helbing's model (bottom row). The data of Helbing model have been collected from the implementation described in Helbing *et al.* (2000). The configuration is the same that the other two experiments.

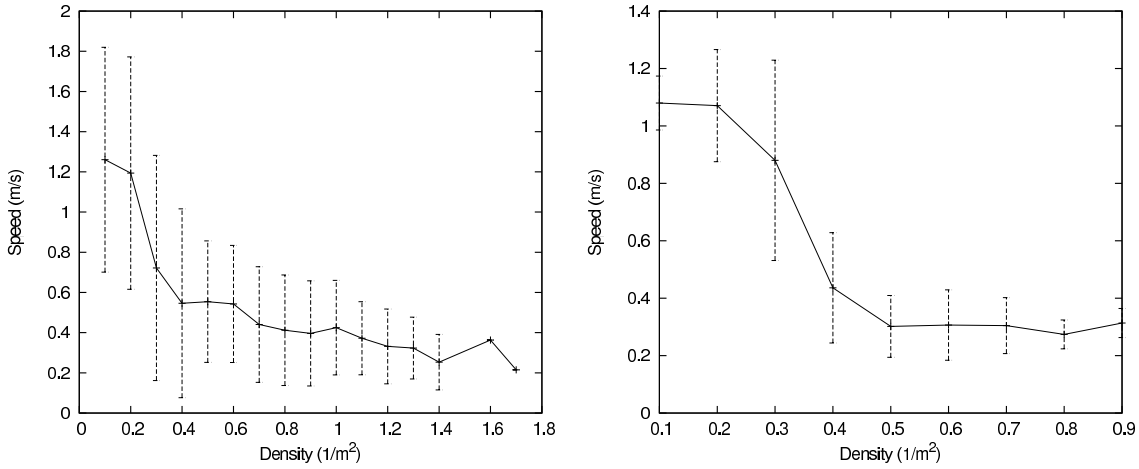


Figure 7.12: Fundamental diagrams of the 'crossing inside a corridor' experiment for a measure point in the center of the corridor with 8 agents. Left: Framework experiment. Right: Helbing's experiment.

	Individual average performance percentage	Collective average performance percentage
With PPR	96.0 ± 0.8	80.0 ± 3.1
Without PPR	92.9 ± 1.0	67.9 ± 4.2
Helbing	100	100

Table 7.2: Performance values for simulation in the crossing experiments. Mean and standard deviation. The individual performance percentage counts the times (of 100 episodes) that an agent has reached to the goal. The collective performance measures the number of times (of 100 episodes) in which all the agents arrived at the goal. The values are the average of 10 experiments.

the other hand, the main characteristics of the diagrams described in the previous experiment also appear in this one, revealing similarities in both models.

In Table 7.2, the performance results of the crossing experiment are displayed. Data show the important influence of PPR in the collective performance, that is, in the rate of right episodes (those in which all the agents arrive at the corresponding goal). On the contrary, the influence is not important at the individual level, stressing the coordination role of the PPR transfer technique in the learning process. In the case of the Helbing experiment all the episodes are successful.

In Figure 7.13, a sequence of images of a simulation is displayed. Note the

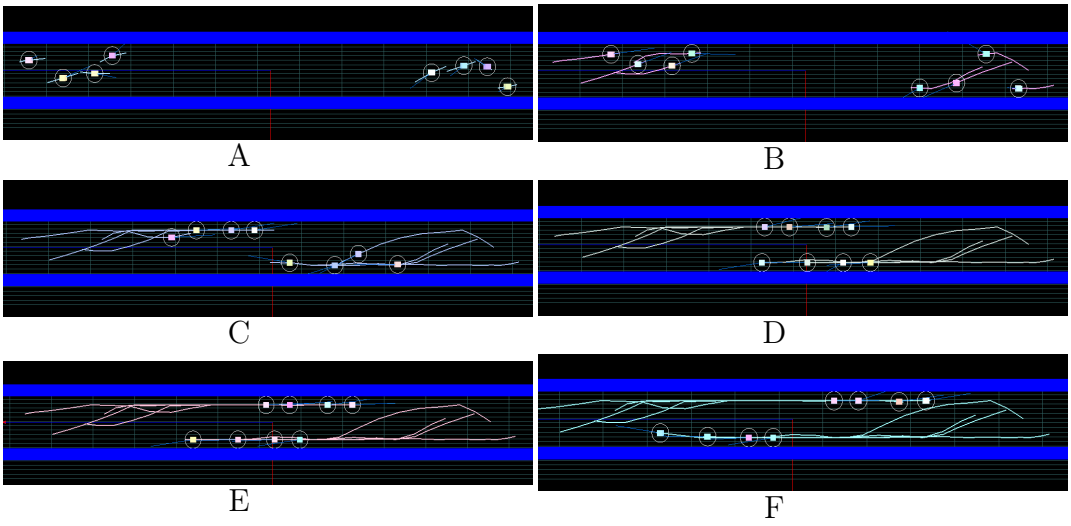


Figure 7.13: A simulation of the 'crossing in a corridor' experiment using learned behaviors. The temporal sequence of stills is sorted alphabetically.

anticipatory organization in lanes at B and C (that is, the agents do not wait for the imminence of a crash to organize, as occurs with models based on forces or potential fields). The videos of several simulations can be seen at `http://www.uv.es/agentes/RL`.

The reader can find a video of special interest at `http://www.uv.es/agentes/RL/crossing_sarsa.htm` that reproduces two simulations of the crossing problem with the Helbing model and the Sarsa(λ) approach. In this video, one can appreciate the different solutions that both approaches provide for the problem. Figure 7.14 contains a sequence from this video. Two different views are shown in each image. The top view shows two corridor scenarios. The left scenario displays the simulation of the learning framework. The right scenario displays the simulation of Helbing's model. In the bottom view, the corridor on the top shows the simulation of the learning framework. The corridor at the bottom shows the simulation of the Helbing's model. In Helbing's model, the agents tend to occupy the center of the corridor where the collisions occur. Whereas, in the Sarsa(λ) approach, the agents have learned to anticipate the crossing and two lanes are created before the crossing occurs.

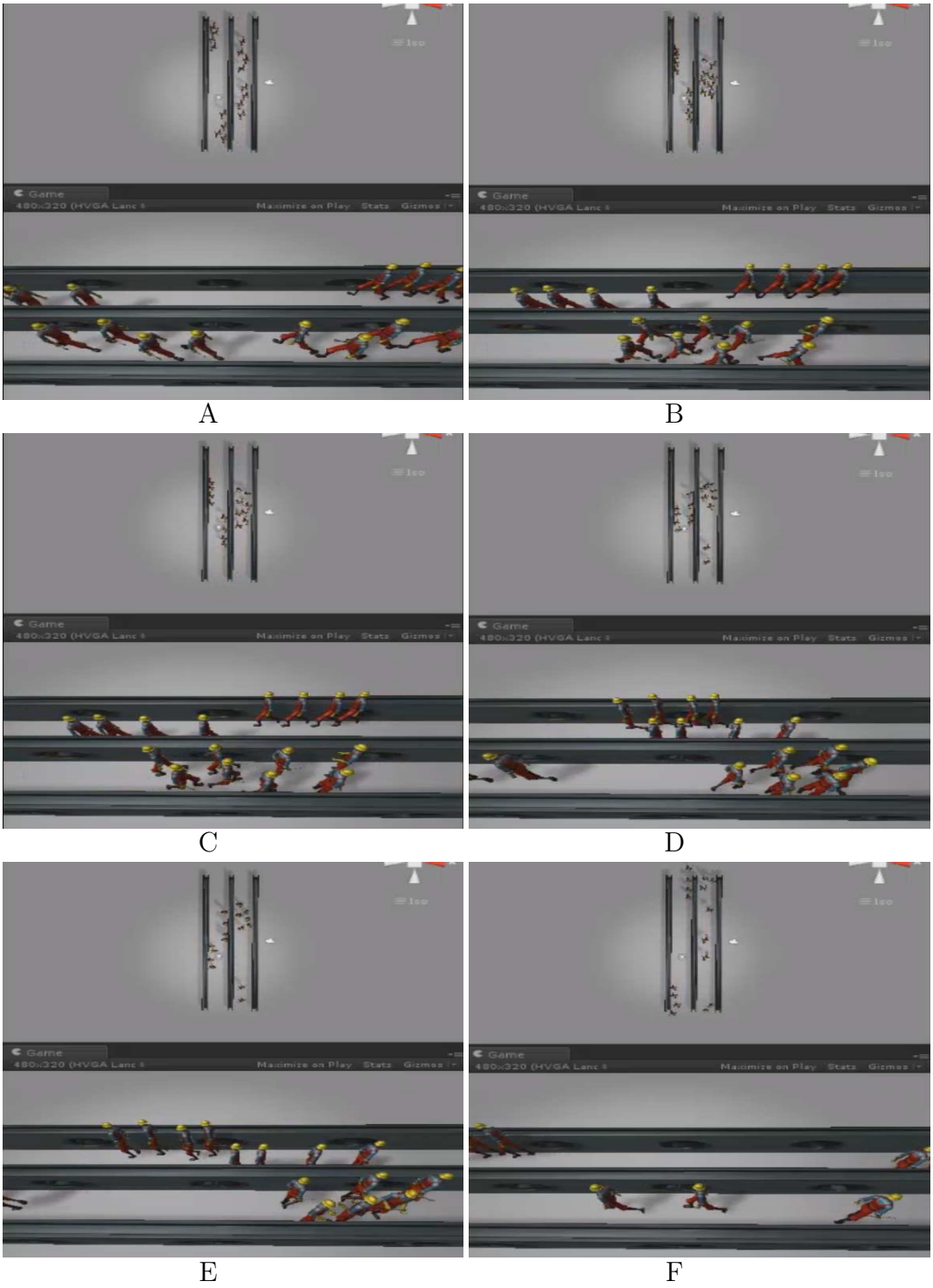


Figure 7.14: A simulation in the 'crossing in a corridor' experiment using learned behaviors with Sarsa algorithm and Helbing's model. Each image presents two views of the same instant. In the top view, simulation with Sarsa corresponds to the left corridor. In the bottom view, simulation with Sarsa corresponds to the above corridor. The temporal sequence of stills is sorted alphabetically.

7.3.3 Pedestrians in a maze

In this experiment, the use of fundamental diagrams and density maps are not appropiate because only two agents are present in the scenario, and therefore, the ranges of density are very short. The view of the learned behaviors shows that the agents have learned to exit from the labyrinth, which implies the use of path planning capabilities (in terms of the definition of a path that solves the problem).

Table 7.3 shows the individual and collective percentages of successful episodes. Note the high percentages of success ($> 95\%$) in both performance measures which indicate that the agents have learned the task.

Individual average performance percentage	Collective average performance percentage
98.1 ± 1.0	96.8 ± 1.5

Table 7.3: Performance results in simulation for labyrinth. Mean and standard deviation. Individual performance percentage counts the times (of 100 episodes) that an agent has reached the goal. Collective performance measures the number of times (of 100 episodes) in which both agents arrived at the goal. The values are the average of 10 experiments.

Figure 7.15 shows a sequence of an episode in simulation. Note in image D of the sequence that the agent coming from the left side executes a maneuver to avoid a collision with the other agent, situated near the wall. This control maneuver corresponds to the operational level. The trajectories of the agents are not symmetrical because each agent learns independently and, therefore, the control of the movement is different for each agent.

In Figure 7.16, a simulation with Unity is displayed. Note how the agents avoid the collision.

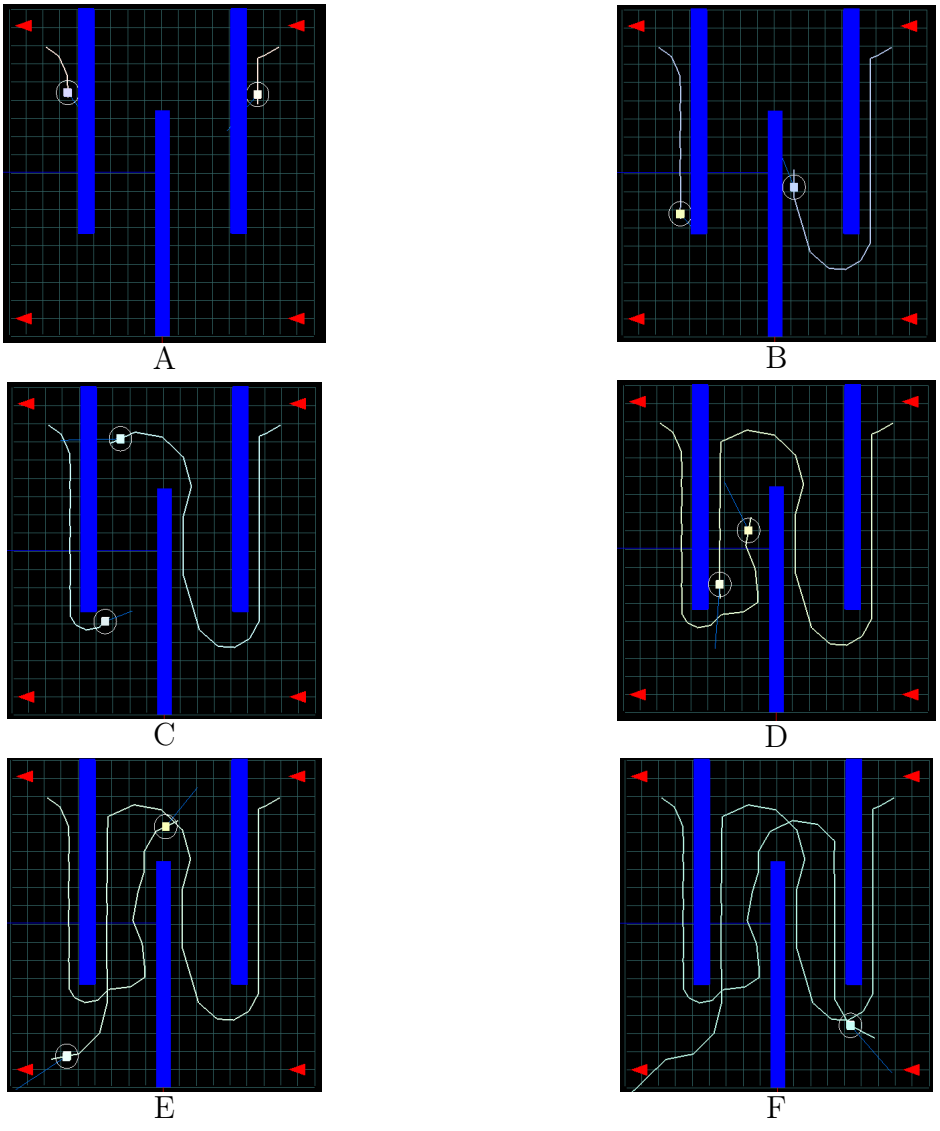


Figure 7.15: Maze experiment. Temporal sequence of stills is sorted alphabetically. Goals are situated at the bottom of the maze

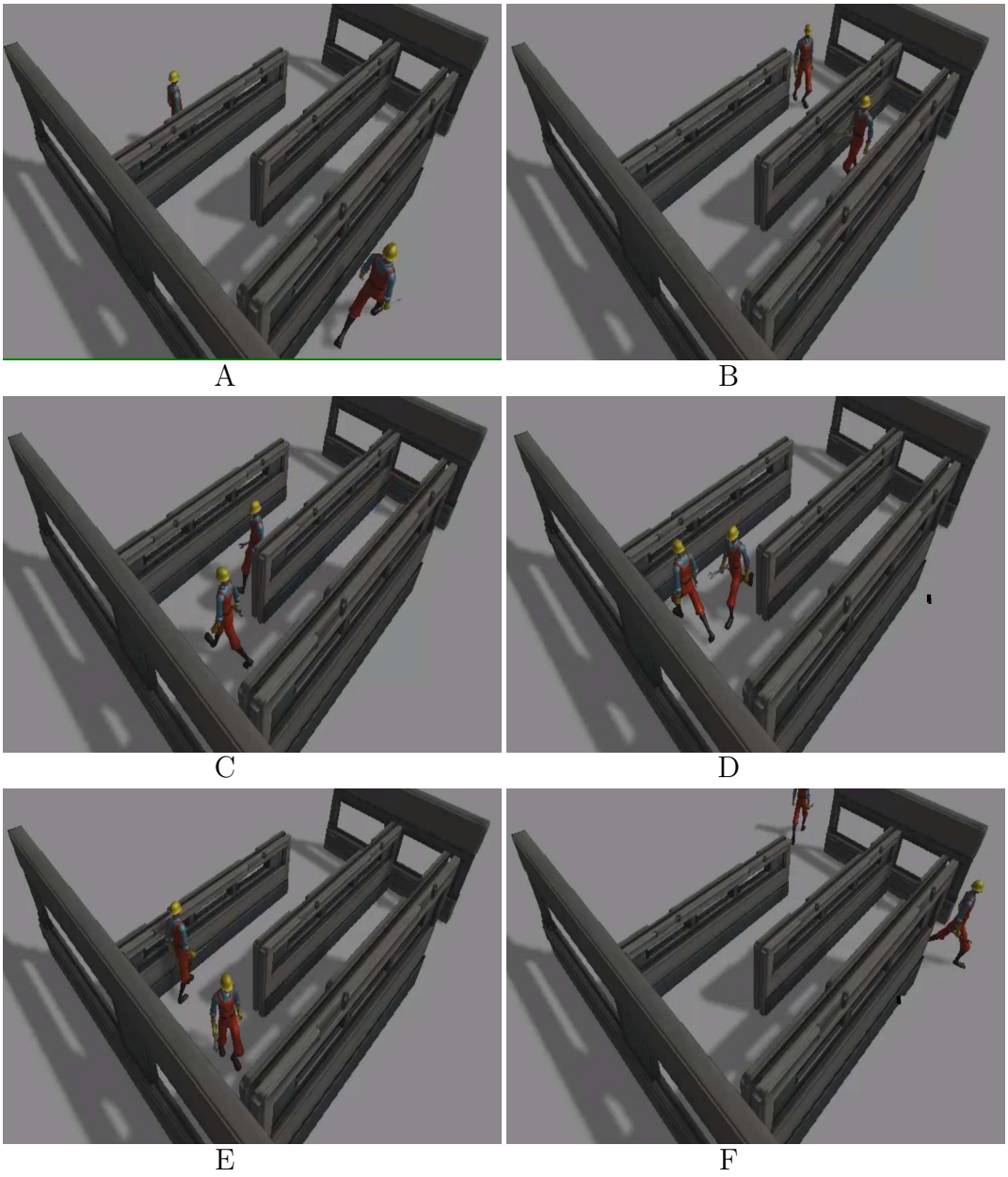


Figure 7.16: Simulation with Unity of the maze experiment. Temporal sequence of stills is sorted alphabetically.

7.3.4 Conclusions of the experiments

These experiments empirically demonstrate that the framework solves the navigation problems at different pedestrian behavior levels (tactical and functional).

Thus, the 'shortest *vs* quickest path' scenario demonstrates the capability to solve navigation problems at the tactical level, because the learned behaviors reproduce the situation of the choice between the shortest and the quickest path. The results of the labyrinth experiment demonstrate capabilities of combining local and global navigation to find a path that solves the problem.

In the crossing scenario, the emergent behaviors (lanes formation) have been created. This implies that the emergence of this collective behavior in this problem is independent of the learning approach (ITVQQL or Sarsa).

The comparison of the fundamental diagrams with Helbing's social forces model reveals similarities in the generated pedestrian dynamics.

7.4 Performance comparison of ITVQQL and Sarsa(λ) with tilecoding (TS) in the 'crossing inside a corridor' scenario

It is difficult to answer the question of what is the best configuration of those studied in this work. First, because it would be necessary to perform studies in many more different scenarios. Secondly, because several parameters of the set up of the different configurations represent trade offs between efficiency and computational cost and, therefore, the configurations are not prepared to obtain the best possible performance but to give good results in a limited time of computation. However, if we restrict the study to a case, it is possible to analyze different setups and give an idea about the strengths and weaknesses of the approaches.

In this section, the influence of the different transfer learning techniques used in the performance is studied. As transfer of knowledge is a way of introducing information in the system, its influence on the results sheds light on the learning efficiency of each configuration.

7.4.1 Experimental setup and results

Six case studies have been designed combining two basic configurations, ITVQQL, described in Section 6.3.1 and Sarsa(λ) with tile coding (this will be abbreviated

as TS in this section), described in Section 3.3.2 and the basic algorithm described in Algorithm 7. Table 7.4 shows the characteristics of each one with its corresponding label.

Config.	Cases	Algorithm	Gener. method	TVF	PPR
ITVQQL	IT-ALL	Q-Learning	VQ	Yes	Yes
	IT-NOPR	Q-Learning	VQ	Yes	No
	IT-NOVF	Q-Learning	VQ	No	Yes
	IT-NOTHING	Q-Learning	VQ	No	No
TS	TS-ALL	Sarsa(λ)	Tile coding	No	Yes
	TS-NOTHING	Sarsa(λ)	Tile coding	No	No

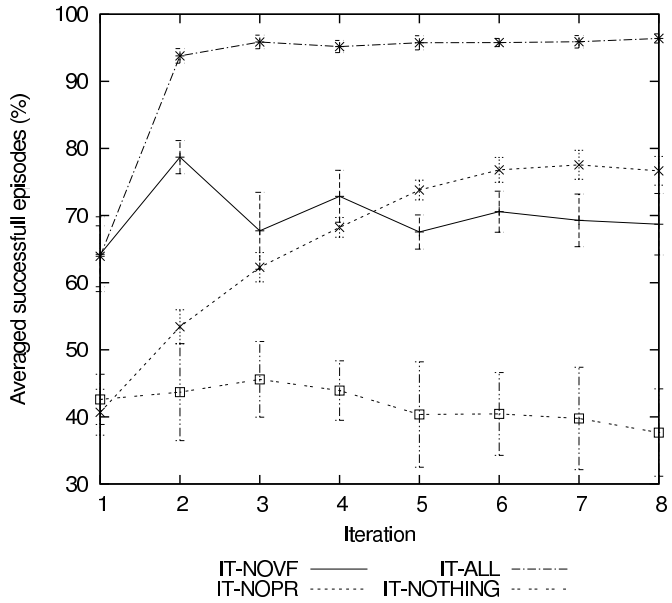
Table 7.4: Case studies considered in this section with their characteristics. The acronym TVF means 'Transfer of Value Function' and PPR means 'Probabilistic Policy Reuse'.

Figure 7.17 shows the performance (mean percentage of times that an agent reach its goal independently of the rest of agents) obtained at the end of each learning process. At the first iteration, the IT-NOVF curve has a similar value to the IT-ALL curve because in that iteration both experiments have the same configuration (there is no value transfer). A similar situation occurs with the IT-NOPR and IT-NOTHING curves. The gap between the IT-NOPR/IT-NOTHING curves and the curves that use PPR, indicates that the bias provided by the policy π_{past} through PPR is very useful for the learning process. Observing all the iterations, the IT-ALL curve (which represent the process that uses both transfer learning techniques) attains the highest values. This iterative schema reaches the asymptote in the third iteration (therefore, it would be sufficient to stop at this iteration).

For the IT-NOPR curve, the iterative schema is useful from iteration 1 to 7. This fact shows that the transfer of the value function needs more iterations when used alone than when used in combination with PPR.

The performance of the IT-NOVR curve remains with soft oscillations along the iterative process. It is partially justified by the fact that PPR is providing the same information in all the iterations. However, an increment would be expected by the consecutive refinement of the VQ as the iterative process progresses (but this is not observed). The same occurs in the IT-NOTHING experiment, possibly

because the learning processes are not long enough to generate better VQs between consecutive iterations. The low values obtained with IT-NOTHING with respect to those observed in the rest of cases reveals the benefits of using both transfer techniques.

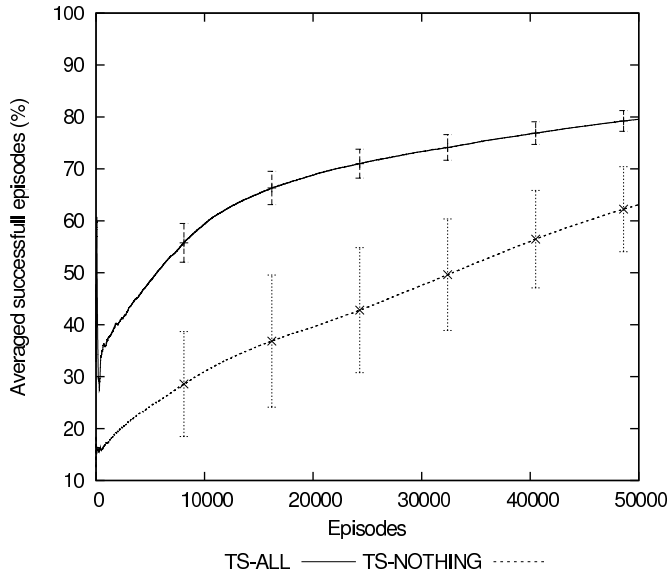


Key	Values
Number of agents	8
α	From 0.3 to 0.15
γ	0.9
Initial value of ϵ	1.0
Iterations	8
Episodes per iteration	50000
Reward Goal reached	100
Number of prototypes	8192

Figure 7.17: Top: Averaged performance for the ITVQQL schema (number of times that an agent reach its goal independently of the rest of agents). Each point is the average performance of all agents at the end of each iteration of learning process. Means are over 8 agents. Down: Common configuration for the cases of study derived of the ITVQQL schema.

In Figure 7.18, the results of the TS approach are reported. Two curves that represent data collected from a single learning process, using PPR (TS-ALL) or not (TS-NOTHING) are represented. The initial gap between both curves is

typical in a knowledge transfer process. In the TS-ALL curve, the beneficial effect of PPR is clear throughout the process with higher performance with respect to the TS-NOTHING case.



Key	Values
Number of agents	8
α	0.004
γ	0.9
Initial value of ϵ	1.0
λ	0.9
Episodes	50000
Reward Goal reached	100
Number of tilings	64

Figure 7.18: Top: Averaged performance for the TS schema (number of times that an agent reach its goal independently of the rest of agents). Each curve is the average performance of all agents throughout a single learning process. Means are over 8 agents. Down: common configuration for the case studies derived of the TS schema.

The analysis of the simulation performance is displayed in Table 7.5. Now the performance measures the percentage of correct simulations, that is, the simulations in which all the agents reach the corresponding goal. The results for the IT-ALL and TS-ALL cases show that the performance is similar for both

Schema	Label	Performance in simulation
ITVQQL	IT-ALL	81 ± 4
	IT-NOPR	30 ± 4
	IT-NOVF	8 ± 2
	IT-NOTHING	0
TS	TS-ALL	80 ± 3
	TS-NOTHING	68 ± 4

Table 7.5: Analysis of the performance in simulation. Mean and standard deviatiation of the percentage of episodes that end successfully from a series of 100 episodes. In a successful episode, all the agents reach to the correspondent goal. The mean of ten series is displayed.

schemas when using transfer of knowledge. On the contrary, when no transfer techniques are used, the performance of the TS-NOTHING is significantly higher than IT-NOTHING. These results indicate that the TS schema is more efficient, in terms of learning capabilities, than ITVQQL in this problem.

The IT-NOPR and TS-NOTHING cases, that do not use PPR, have lower performance with respect to IT-ALL and TS-ALL cases (30 and 68 *vs.* 81 and 80). The percentages are relevant enough to indicate that the emergent collective phenomena depends on whether the additional information is provided by the PPR transfer method. From this data, we can also conclude that the influence of the use of PPR is higher in the ITVQQL schema than in the TS approach.

The low value for IT-NOVF with respect to the other cases that include transfer of value function (IT-ALL and IT-NOPR) indicates that the use of this kind of knowledge transfer has a strong influence in the performance of the ITVQQL schema. For the IT-NOTHING case, a value performance of 0 is obtained. This reveals that the crossing problem is not solved correctly, and, in simulation, lane formation can not be clearly observed.

In Table 7.6 a comparison of the CPU time for the learning processes of the two schemas is shown. The experiments were executed on a Altix UltraViolet 1000 Silicon Graphics server with 64 CPU Xeon Serie 7500 hexacore, 2,67 GHz and 18 MB of L3 on-die memory with a total of 384 cores and 960 GB of RAM memory (also known as 'Lluis Vives'). Each agent of the framework runs in a different core unit. The time consumed by the process that gives the best results for

Schema	Label	CPU time (secs)	CPU time (hh:mm:ss)
ITVQQL	IT-ALL	**7315**	**02 : 01 : 55**
	IT-NOPR	9578	02 : 39 : 38
	IT-NOVF	9128	02 : 32 : 08
	IT-NOTHING	9094	02 : 31 : 34
TS	TS-ALL	**4201**	**01 : 10 : 01**
	TS-NOTHING	4250	01 : 10 : 50

Table 7.6: CPU time for one learning process in different cases. The ITVQQL schema CPU times are estimates for one iteration (50000 episodes) in order to make measurements comparable with the TS schema CPU time. The best result of each schema is bolded.

each approach is bolded. The time for the ITVQQL family is an estimate. Only the execution time of the whole iterative process (8 learning iterations of 50000 episodes each) was available in the hardware system. In order to make the time comparable with that of the TS configuration (a single learning process of 50000 episodes), the total CPU time consumed by the eight iterations is divided to get a per-iteration estimation. From this table we can extract two main conclusions: i) the iterative processes are computationally more expensive than the TS processes for this problem. This is likely due to the cost of the VQ calculation. ii) the use of transfer of knowledge techniques accelerate the learning processes in all the cases, as expected.

7.4.2 Conclusions of the experiment

From the results of the experiments described in the previous section we can derive the following conclusions:

- Lane formation is an emergent collective behavior that appears independently of the learning/generalization approach used. However, it requires a learning bias in the exploration process, which is provided through the PPR method.
- The ITVQQL and TS schemas show similar performance results in simulation when the transfer learning techniques are active. ITVQQL uses more CPU time than TS in this problem.

- The use of knowledge transfer techniques improves the performance of both schemas. Specifically, the transfer of value function technique is important in the TRVQQL schema. However, TS is more efficient than ITVQQL in this problem when knowledge transfer techniques are not used.

The emergence of collective behaviors, independently of the learning algorithm and generalization method, suggests that our RL multi-agent framework is robust, in term of its configuration, to address pedestrian simulation problems.

Although the TS configuration seems more efficient that the ITVQQL configuration, it would be unwise to prefer one over the other. First, because more experiments would be needed to make a reliable statement and, secondly, because there are not guarantees that the algorithms were optimally tuned, which is a common problem when comparing complex parametrized systems.

7.5 Chapter highlights

- In this chapter, the Sarsa(λ) with tile coding configuration of the framework is studied. The new experiments carried out analyze the capability of generating learned behaviors that function at higher levels than the operational level (the level where the local interactions are considered). The 'crossing inside a narrow corridor' scenario is also revisited. The results indicate that the learned behaviors are complex enough to solve situations (route choice and path planning) that require higher levels of abstraction in pedestrian simulation. Moreover, the comparison of the fundamental diagrams with those of the Helbing model in two experiments indicates similarities in the dynamics.

- A comparison between the ITVQQL schema and Sarsa(λ) with tile coding is performed in the corridor scenario. Six case studies are carried out using different configurations of transfer of knowledge techniques. The experiments reveal that transfer of knowledge techniques are important in order to achieve good performance marks in both schemas.

Chapter 8

Conclusions and future works

In this chapter I will set out the contributions of this work to the state of the art in pedestrian simulation, RL and Multi-agent RL. Additionally, several proposals for future work will be indicated.

8.1 Conclusions

The main contribution of this thesis is the proposal of a new approach to the problem of pedestrian simulation. Through several experiments, this work has demonstrated that Multi-agent RL operating in calibrated-like-pedestrian embodied agents can generate plausible pedestrian behaviors in different scenarios. This new approach has different benefits (variability of produced behaviors, fast decision making in simulation time, behaviors capable of operating at different levels and producing emergent collective behaviors). However, the most important benefit is transferring the responsibility of designing the pedestrian behaviors from the user to the learning system.

The first challenge was the design, implementation, calibration and validation of a multi-agent learning framework in a continuous state space. The implemented framework uses the MPI communication protocol to build parallel programming-based software which is flexible enough to allow for experimentation with different learning algorithms and different generalization techniques. The calibration of the physics module was achieved by designing a model of collisions based on the human characteristics of the skin, as well as a model of pedestrian dynamics.

The learned dynamics obtained in simple scenarios showed similarities with real pedestrian dynamics which validates the system.

Two different learning strategies have been implemented: one based on the Q-learning algorithm and another based on Sarsa(λ). In the first strategy, two VQQL-based schemas (ITVQQL and INVQQL) were compared in two different scenarios. First, the learning experiments demonstrate that both are convergent and capable of finding policies that carry out the proposed tasks. In the first scenario (closed room with an exit), a deeper study (at the macro and micro levels) of the learned dynamics indicates robustness for scalability in the number of agents. Moreover, the comparison at the macroscopic level with Helbing's social forces model shows similarities with our results in terms of the fundamental diagrams and density maps. This supports the idea that our agents have developed plausible pedestrian behaviors. With the second scenario (crossing inside a narrow corridor) it is shown that the iterative schemas are capable of generating emergent collective behaviors, which is an indicator of quality.

The Sarsa(λ) with tile coding schema has been tested in scenarios where higher level of behavior (tactical) were needed to solve the task. The learned behaviors are capable of solving the problems (route choice and path finding), because these high level skills were acquired. Additionally, comparison with Helbing's social forces model indicates similarities among the fundamental diagrams in both approaches. The crossing scenario has been revisited using this learning schema and a similar emergent collective behavior appeared. This suggests that the emergence of collective behaviors could be independent of the learning approach. Additionally, the comparison between ITVQQL and Sarsa(λ) schemas in the narrow corridor scenario reveals the influence of transfer of learning techniques on the performance of the algorithms. The results indicate that the Sarsa schema is more efficient than the ITVQQL schema in this scenario.

Finally, the observation of the simulations (different simulations can be found in (`http://www.uv.es/agentes/RL`) reveals that agents have developed plausible pedestrian behaviors in the different scenarios.

8.2 Future work

As with any work that has covered its early stages, further research in a number of directions is needed. I propose several directions that I consider to be important.

Concerning calibration.

It would be interesting to use real pedestrian dynamics data to calibrate or guide the learning process. The use of real data in the learning process could give stronger support to the idea that the introduced approach could actually become a model of pedestrian navigation. In this framework, it is difficult to use statistical methods to adjust parameters because there is no direct influence of the learning parameters on the dynamics. However, RL techniques exist that can be used to exploit the information provided from real examples. Batch reinforcement learning (Kalyanakrishnan & Stone, 2007; Lange *et al.*, 2012) learns the best possible policy from a fixed set of a priori-known transition samples. After a, likely non trivial, transformation task of real pedestrian data in experience tuples, Bath RL would learn a policy based on real data. This policy could be then transferred to the RL framework using PPR or other knowledge transfer techniques. Databases exist with real data collected from motion capture and video tracking that represent real interactions among pedestrians(Metoyer & Hodgins, 2004) which can be used to test the framework with other real situations.

A similar idea is introduced in learning from demonstration (LFD) techniques. A demonstration in this context consists of sequences of state-actions pairs that are recorded during the teacher's demonstration of the desired behavior (Argall *et al.*, 2009). LFD algorithms utilize this data set of examples to derive a policy that reproduces the demonstrated behavior. In the work by Taylor *et al.* (2011) human demonstrations are transferred into a baseline policy for an agent and refined using reinforcement learning.

Concerning authoring capabilities.

A drawback in this approach is the lack of editability or authoring of the learned behaviors. The learning process is carried out autonomously and without super-

vision by each agent. The result is a learned value function which is not easy to modify, which constitutes an important issue in behavioral animation for virtual environments. The problem of authoring (the capability of the user/author to control the final animation) has to be situated during the learning process and not after it has finished. There are several ways of integrating human feedback with RL. One way to address the problem is to treat human feedback as a shaping reward (Knox & Stone, 2008; Tenorio-Gonzalez *et al.*, 2010). The idea of reward shaping is to provide an additional reward representative of prior knowledge beyond that supplied by the underlying MDP (Ng *et al.*, 1999). This reward has the form of a potential function that is defined over a source s and a destination state s'. Another approach, policy shaping (Griffith *et al.*, 2013) treat human feedback as direct information about policies that can be used as policy advise. Another type of shaping consists of giving the learning agent a series of relatively easy problems building up to the harder problem of ultimate interest. This idea is used in (Randlov & Alstrom, 1998), to learn how to ride a simulated bicycle.

Another approach to the problem consists of using LFD techniques, commented on above, to provide a correction of the executed behavior. In this case, the expert would carry out demonstrations about the behavior that should be modified. In Nicolescu & Matarić (2003), the correct discrete action provided by a human teacher updates the structure of a hierarchical neural network of robot behaviors. The same occurs in Chernova & Veloso (2008), where the correction updates an action classifier. A work that has already implemented this idea in an RL framework is Dinerstein *et al.* (2007). In this work, the user demonstrates the desired behavior by dictating the agent's actions during an interactive animation. Later, when the agent is to behave autonomously, the recorded data is generalized to form a continuous state-to-action mapping.

Concerning new experimental tests.

Once the usefulness of the RL techniques in the pedestrian simulation domain has been proved, other experiments will be reproduced and tested in order to validate the power of the approach. They can be classified as:

- Reactive experiments. Focused mainly on local interactions. The work

by Shao & Terzopoulos (2005), proposes different routines based on basic reactive behaviors: static obstacle avoidance, avoidance in a complex turn, separation maintenance inside an organization, close neighbors avoidance. The work by Kim *et al.* (2013b) also uses these basic behaviors to build more complex scenarios such as two bottlenecks scenario or the cluttered office scenario.

- Anticipated collision avoidance experiments. Results in the crossing and the maze scenario provided evidence that these anticipatory maneuvers appear. A candidate is pair-interaction in a crossing, proposed and analyzed by Paris *et al.* (2007); Pettré *et al.* (2009) to better understand the anticipation process. In this scenario, two agents cross in a small squared area using different spatial configurations.

- Emergent behaviors experiments. In the paper by Pelechano *et al.* (2007) several emergent behaviors are proposed such as line organization or pushing. Other collective behaviors are proposed in the paper by Helbing *et al.* (2005) such as roundabout traffic patterns at intersections.

- Imitating social behaviors. To develop group patterns similar to a meeting with friends, queuing at ticketing areas or selecting an unoccupied seat, described in the work by Shao & Terzopoulos (2005).

- High level behaviors. Other route-choice scenarios will be proposed. For instance, the work by Asano *et al.* (2010) describes several studies with real pedestrians in this kind of problem using a reproducible railway station scenario.

In addition, a scalability study from hundred to thousands of agents could be addressed to analyze the robustness of the learned behaviors. Scalability is not a problem at the architectural level. The framework is prepared to use a blackboard architecture to share the learned value functions in simulation time. Given N agents and V learned value functions, N/V agents will share the same value function and its associated generalization system. Several challenges arise in this study such as maintaining the consistence of the scaled scenario in terms

of density, or managing with situations where an agent remains in a bad learned area of the state space (including strategies to detect these areas in learning time).

Concerning design improvement.

Another interesting line to open up consists of the modeling of internal and psychological motivations in the agents. Scenarios like panic evacuations or impatience in a queue are challenging to model with this approach. Related to this, one direction of research is to study the incorporation and management of new perceptions in the definition of the state space that make the agent aware of internal or psychological characteristics of neighbors and him/herself. Furthermore, the definition of new actions oriented to the modification of the features corresponding to these internal states would be advisable. How these new internal capabilities could affect high-level and low-level agent behavior is an interesting consideration.

Concerning the learning configurations, the libraries of learning algorithms and generalization methods should be extended. A wide collection of algorithms available is a tool to discover affinities between learning/generalization algorithms and problems. This could be a method to discover patterns in problems that are satisfactorily solved with the same configuration.

Concerning tool availability.

A work of dissemination of the approach introduced in this work is also necessary. Although visibility is already present on the web site designed for this purpose (and referenced many times in the text), it is necessary to provide the research community with an operative version of this work in order for it to be checked with or against other pedestrian simulation approaches. Therefore, a work of documentation of the tool, arrangement of software packages and some kind of open source software license is also required.

8.3 Publications derived from this work

Below, there is a list of the publications directly derived from this work. The cites are ordered by date.

- Francisco Martinez-Gil, Fernando Barber, Miguel Lozano, Francisco Grimaldo, Fernando Fernández. A Reinforcement Learning Approach for Multiagent Navigation. *Proceedings of the International Conference on Agents and Artificial Intelligence.* ICAART(**1**) pp. 607–610. Valencia, Spain, January 22-24, 2010. INSTICC Press 2010, ISBN 978-989-674-021-4

- Francisco Martinez-Gil, Miguel Lozano, Fernando Fernández. Multi-Agent Reinforcement Learning for Simulating Pedestrian Navigation. Adaptive and Learning Agents Workshop at AAMAS (ALA'2011). LNAI 7113 (P. Vrancx, M. Knudson, M. Grzes Eds.). Pags.54-69. Springer. 2012

- Francisco Martinez-Gil, Miguel Lozano, Fernando Fernández. Calibrating a motion model based on reinforcement learning for pedestrian simulation. ACM SIGGRAPH Conference on Motion in Games (MIG 2012) Rennes (France). LNCS 7660 (M. Kallmann, K. Bekris eds.) pp. 302-313. Springer. 2012.

- Francisco Martinez-Gil, Miguel Lozano, Fernando Fernández. Emergent collective behaviors in a multi-agent reinforcement learning based pedestrian simulation. AAMAS 2014 Workshop: Multi-Agent-Based Simulation (MABS 2014) Paris, France. 2014. To be included in the LNAI collection of Springer. (Included also as an extended abstract in the booklet of the First Multidisciplinary Conference on Reinforcement Learning and Decision Making (RLDM2013). Princeton University, New Jersey, USA. 25-27 October 2013.)

- Francisco Martinez-Gil, Miguel Lozano, Fernando Fernández. Strategies for simulating pedestrian navigation with multiple reinforcement learning agents. *Autonomous Agents and Multi-Agent Systems.* 2014. Springer. DOI: 10.1007/s10458-014-9252-6 (In Press).

- Francisco Martinez-Gil, Miguel Lozano, Fernando Fernández. MARL-Ped: a Multi-Agent Reinforcement Learning Based Framework to Simulate Pedestrian Groups. *Simulation and modelling. Practice and theory* **47**:259-275. 2014. Elsevier. DOI: 10.1016/j.simpat.2014.06.005

Appendix: Resumen de la Tesis Doctoral

Introducción

Esta Tesis Doctoral se centra en el estudio, desarrollo y aplicación de aprendizaje por refuerzo (RL) al dominio de la simulación de grupos de peatones. El aprendizaje por refuerzo es una rama del Aprendizaje Automático dentro de la Inteligencia Artificial. Su campo de aplicación se extiende sobre los problemas de optimización que son modelados como Procesos de Decisión de Markov (MDP) (Busoniu *et al.*, 2008; Kaelbling *et al.*, 1996; Sutton & Barto, 1998). Un MDP es un modelo de toma de decisiones secuencial en el que cada decisión viene acompañada de un valor, llamado recompensa inmediata, que señala su idoneidad dado el estado en que se encuentra el proceso. El objetivo del MDP consiste en optimizar una función de esas recompensas. En el caso más general el proceso es estocástico tanto en las transiciones entre estados como en las recompensas, desconociéndose el modelo de transición.

La simulación de peatones y en general el modelado de las dinámicas de peatones han adquirido interés en los últimos años en la medida en que el campo de la simulación se ha extendido a otras áreas. Así se utilizan simulaciones con modelos de grupos de peatones en entornos arquitectónicos para estudiar y diseñar la accesibilidad de los mismos. En obras como estadios, puentes peatonales, auditorios etc. se realizan estudios de simulación para analizar la respuesta del edificio o de la instalación frente a casos de pánico, evacuaciones o picos eventua-

les de uso. También el uso de peatones se ha extendido a los entornos virtuales y videojuegos ya que son parte integrante de las simulaciones en ambientes urbanos.

Existen actualmente diferentes modelos de peatones (Bierlaire & Robin, 2009) con diferentes aproximaciones (macroscópicas y microscópicas) y orientaciones (hacia peatones individuales o hacia grupos de peatones e incluso multitudes). Entre los más conocidos se encuentran los modelos de fuerzas sociales de Helbing (Helbing & Molnár, 1995), los modelos basados en reglas de Reynolds (Reynolds, 1987), el modelo de colas de Lovas (Lovas, 1994), los modelos de multitudes de Still (Still, 2000) y los modelos basados en mediciones sobre peatones reales de Daamen (Daamen & Hoogendoorn, 2003), Teknomo (Teknomo, 2002) o Bierlaire (Robin *et al.*, 2009).

En el presente trabajo se plantea una nueva aproximación en el campo de la simulación microscópica de peatones, basada en sistemas multiagente que usan aprendizaje por refuerzo. En esta propuesta, la tarea es el aprendizaje individual y autónomo por parte de cada agente con entidad física (embodied agent) de un control de velocidad que le dirija en la navegación por un entorno virtual. La aproximación propuesta tiene características que la hacen interesante para la simulación de peatones. Concretamente:

1. No es necesario disponer a priori de un modelo de interacciones ni de comportamiento, ya que esto es el objetivo del aprendizaje.

2. El uso de aprendizaje por refuerzo representa una opción prometedora para la generación de comportamientos colectivos emergentes, al estar éstos relacionados con procesos de optimización.

3. Las políticas de navegación obtenidas por aprendizaje por refuerzo son capaces de operar a diferentes niveles, como lo prueban experimentos hechos en robótica.

4. Una vez el sistema multiagente ha aprendido, la simulación del comportamiento individual es poco costosa computacionalmente.

5. Los comportamientos son aprendidos individualmente por cada agente, con lo que se dispone de un conjunto de comportamientos diferentes que dan más realismo a la simulación que si fuera una repetición del mismo esquema.

Objetivos

El objetivo de esta tesis es la creación de agentes autónomos capaces de aprender comportamientos que produzcan simulaciones plausibles de navegación de peatones, utilizando una aproximación multi-agente basada en aprendizaje por refuerzo.

Para realizar este objetivo se consideran los siguientes hitos en el trabajo de esta tesis:

1. Diseño, desarrollo, calibración y validación de un marco de trabajo multi-agente con capacidades de aprendizaje.

2. Evaluación de dos estrategias de aprendizaje diferentes: una basada en el algoritmo Q-Learning y otra basada en el algoritmo Sarsa(λ).

3. Estudio de la adecuación de los comportamientos dinámicos generados a la dinámica de peatones real. Para ello se usarán herramientas específicas del campo de la simulación de peatones y comparaciones con otro modelo de peatones.

4. Estudio de la capacidad del sistema de generar comportamientos colectivos emergentes así como que dichos comportamientos tengan capacidades de operar en los niveles tácticos y estratégicos. Para ello, se propondrán escenarios específicos en los que el control a esos niveles es necesario para la resolución del problema de navegación.

En resumen, la principal contribución de este trabajo consiste en proponer una nueva aproximación al problema de la simulación de peatones basada en un enfoque multi-agente con técnicas de aprendizaje por refuerzo y demostrar empíricamente que esta aproximación proporciona resultados positivos en un conjunto de problemas paradigmáticos en la navegación de peatones.

Metodología

La implementación del marco de trabajo y experimentación exige el diseño y codificación de un software flexible que permita la incorporación de los diferen-

tes algoritmos y técnicas que pretendemos experimentar. De tal manera que los módulos sean fácilmente intercambiables para favorecer la experimentación con configuraciones diferentes. Por ejemplo, es necesario que tanto los algoritmos de aprendizaje de los agentes como los algoritmos de generalización del espacio de estados se configuren en módulos de tal manera que sean intercambiables y extensibles. Igualmente, es necesario que el simulador del entorno físico sea intercambiable para hacer más flexible y potente la simulación de diferentes entornos virtuales. Desde este punto de vista de la modularización, podemos distinguir tres unidades funcionales en la arquitectura que va a ser propuesta:

1. La entidad "Agente" en el que se deberán desarrollar interfaces y submódulos para incorporar técnicas de generalización, algoritmos de aprendizaje y conocimiento aprendido

2. La entidad "Entorno" encargado de la simulación del entorno virtual y de la sensorización de los agentes virtuales. Igualmente posee un módulo de evaluación y recompensa de las acciones que emprenden los agentes y que es la base del aprendizaje por refuerzo.

3. El módulo de paso de mensajes entre el entorno y los agentes

El sistema se ha desarrollado bajo el modelo de programación paralela MPI. Bajo este modelo, cada 'Agente' es una entidad autónoma que realiza de manera individual y original el proceso de aprendizaje, observando al resto de agentes como parte del entorno. Igualmente, el 'Entorno' es un agente único y separado que está a cargo de la simulación física del mundo virtual así como de la ejecución de las acciones sugeridas por los agentes sobre dicho mundo virtual y de la crítica de estas mismas acciones. El uso del protocolo MPI permite el flujo de información entre los agentes y el entorno. La propia paralelización aumenta la eficiencia de la ejecución al distribuir la carga en diferentes núcleos de computación (idealmente un agente por núcleo). Además de la eficiencia, esta arquitectura favorece la escalabilidad en el número de agentes.

La calibración del sistema se ha realizado básicamente en dos aspectos. La primera se ha centrado en calibrar el simulador del entorno físico que forma parte

del entorno virtual antes mencionado. Para ello se extraerá de fuentes bibliográficas los datos correspondientes a peatones humanos necesarios para modelar y calibrar los parámetros del simulador físico. La segunda área corresponde a la calibración de las acciones que ejecutan los agentes virtuales (básicamente la modificación de su impulso) para que correspondan a reacciones humanas. Para la validación de los experimentos se escojerán entornos ya estudiados por trabajos relevantes dentro del campo del modelado de peatones y la simulación. Existe una amplia bibliografía en este campo. Como candidatos podemos nombrar el trabajo sobre movimientos de peatones en fila de Seyfried et al. (Seyfried *et al.*, 2005) y el trabajo en áreas abiertas de Weidmann (Weidmann, 1993).

Para la consecución de los objetivos, se han desarrollado dos grupos importantes de experimentos con dos configuraciones de aprendizaje diferentes que incluyen además técnicas de transferencia de conocimiento (Taylor & Stone, 2009). La primera configuración estudia dos variantes del algoritmo VQQL (Fernández *et al.*, 2005) denominadas ITVQQL y INVQQL sobre dos escenarios distintos (salida de un grupo a través de una puerta y cruce de dos grupos en un pasillo estrecho). A través de estos experimentos se estudian las características dinámicas de los comportamientos aprendidos y se comparan con el modelo de fuerzas sociales de Helbing (Helbing & Molnár, 1995). La segunda configuración consiste en el algoritmo de aprendizaje Sarsa(λ) con el método de generalización tile coding (Sutton & Barto, 1998). Sobre esta configuración se realizan experimentos en escenarios donde la planificación y el comportamiento a nivel táctico son relevantes. Estos nuevos escenarios son, concretamente, un escenario con dos salidas donde cada peatón puede elegir entre el camino más corto y el camino más rápido y un laberinto donde los agentes deben poseer capacidades de planificación para encontrar la salida. Además, se revisita el escenario del cruce de grupos en el pasillo para compararlo con los resultados obtenidos con la primera configuración. Se analiza en dicho escenario la influencia de las téccnicas de transferencia de conocimiento incluídas en los algoritmos de aprendizajes.

Todos los experimentos son analizados a partir de los datos generados por un número determinado de simulaciones. Se utilizan herramientas de análisis específicas de dominio del modelado y simulación de peatones, como es el diagrama fundamental (Weidmann, 1993), que establece la relación entre densidad en un

área y las velocidades que alcanzan los peatones en ella. Otras herrramientas utilizadas para el análisis de los resultados son los mapas de densidad (histogramas que miden el uso de la superficie) y los análisis de rendimiento (en términos del número de agentes que logran el objetivo por episodio).

Conclusiones

La aportación fundamental de este trabajo es la presentación de una nueva aproximación para la simulación de peatones. A través de los diferentes experimentos realizados, se demuestra empíricamente que esta nueva aproximación multi-agente basada en RL es capaz de generar comportamientos plausibles de peatones en términos visuales y de adecuación a las dinámicas observadas en los peatones reales y de otros modelos. Los comportamientos aprendidos tienen las ventajas de ser naturalmente heterogéneos, de poder manejar diferentes niveles de coportamiento (como el operacional y el táctico) y de ser computacionalmente eficientes. Sin embargo, el mayor beneficio de esta aproximación consiste en el desplazamiento de la responsabilidad del diseño del comportamiento de los peatones desde el usuario hacia el sistema.

En el desarrollo de este trabajo se han logrado diferentes hitos:

- Se ha diseñado y construído un sistema multi-agente que utiliza aprendizaje por refuerzo para obtener agentes virtuales con comportamientos dirigidos a la simulación de peatones.

- Se ha calibrado y validado dicho sistema comparándolo con otros experimentos similares sobre peatones reales.

- Se han desarrollado nuevas estrategias algorítmicas basadas en el algoritmo VQQL con el fin de adaptarlo al problema multi-agente.

- Se han diseñado y realizado experimentos en escenarios que requerían para su correcta solución de características observadas en peatones reales tales como comportamientos emergentes y comportamientos a diferentes niveles.

- Se han comparado los resultados en los diferentes escenarios con los obtenidos por el modelo de Helbing en similares escenarios.

Como trabajo futuro se presentan otros desafíos:

1. Aplicar técnicas de RL para incluir en el proceso de experiencia tuplas de experiencias de peatones reales. Esto puede hacerse a través de técnicas de RL como el aprendizaje basado en ejemplos o el RL por lotes (Batch RL) (Kalyanakrishnan & Stone, 2007; Lange *et al.*, 2012).

2. Dotar al sistema de capacidades de edición de los comportamientos. Esto, por la naturaleza de RL, debe hacerse dentro del mismo proceso de aprendizaje y no cuando éste ya ha terminado. Existen técnicas de RL como el aprendizaje por demostración Nicolescu & Matarić (2003) o el modelado de las recompensas (reward shaping) (Ng *et al.*, 1999) que pueden ayudar a esta tarea.

3. Es necesario hacer pruebas en otros nuevos escenarios para corroborar la capacidad de afrontar diferentes clases de problemas de esta aproximación. Podemos detectar varios tipos de escenarios donde es interesante hacer estos experimentos:

    - Experimentos reactivos. Centrados en el análisis de las interacciones locales. Los trabajos de Shao & Terzopoulos (2005) analizan varios escenarios que pueden ser reproducidos.
    - Experimentos de anticipación de colisiones. El trabajo de Pettré *et al.* (2009) propone determinados escenarios pare comprender mejor estos procesos de anticipación.
    - Experimentos de comportamientos emergentes. Los trabajos de Helbing *et al.* (2005); Pelechano *et al.* (2007) describen nuevas situaciones que implican la aparición de este tipo de comportamientos.
    - Comportamientos a alto nivel. El trabajo de Asano *et al.* (2010) propone nuevos casos donde la capacidad de elección de rutas tiene importancia para la resolución del problema.

4. Queda, por último, indicar un trabajo de diseminación de esta nueva aproximación al problema. Aunque la visibilidad del trabajo ya está presente en el sitio web referenciado a lo largo de este trabajo, es necesario proporcionar a la comunidad de investigadores una versión operativa de la herramienta. Por lo tanto, es necesario retomar el trabajo de documentación y organización del software bajo algún tipo de licencia Open Source.

References

Abbeel, P., Coates, A. & Ng, A.Y. (2010). Autonomous helicopter aerobatics through apprenticeship learning. *Robotics Research*, **29**, 1608–1639. 91

Agre, P. & Chapman, D. (1987). Pengi: An implementation of a theory of activity. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, 268–272, Morgan Kaufmann. 110

Albus, J.S. (1975). A new approach to manipulator control: the cerebellar model articulation controller (CMAC). *Journal of Dynamic Systems, Measurement, and Control*, **97**, 220–227. 81

Ali, S. & Shah, M. (2008). Floor fields for tracking in high density crowd scenes. In *Computer Vision - ECCV 2008, 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part II*, 1–14. 39

Almeida, J., Kokkinogenis, Z. & Rossetti, R. (2013). Towards a framework for pedestrian simulation for intermodal interfaces. In *7th European Modelling Symposium (EMS'13). Manchester. UK*. 40

Aloimonos, Y., Bandyopadhyay, A. & Weiss, I. (1988). Active vision. *International Journal of Computer Vision*, **1**, 333–356. 37

Andre, D. & Russell, S.J. (2002). State abstraction for programmable reinforcement learning agents. In *Eighteenth National Conference on Artificial Intelligence*, 119–125. 86

Argall, B.D., Chernova, S., Veloso, M. & Browning, B. (2009). A survey of robot learning from demonstration. *Robot. Auton. Syst.*, **57**, 469–483. 214

Asada, M., Noda, S., Tawaratsumida, S. & Hosoda, K. (1994). Vision-based behavior adquisition for a shooting robot by using reinforcement learning. In *IAPR/IEEE Worshop on Visual Behaviors*, 112–118. 86

Asano, M., Iryo, T. & Kuwahara, M. (2010). Microscopic pedestrian simulation model combined with a tactical model for route choice behaviour. *Transportation Research Part C: Emerging Technologies*, **18**, 842–855. 216, 226

Bandini, S., Manzoni, S. & Vizzari, G. (2009). Agent based modeling and simulation: An informatics perspective. *Journal of Artificial Societies and Social Simulation*, **12**, 4. 29

Banerjee, B. & Stone, P. (2007). General game learning using knowledge transfer. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-07)*, 672–677, Hyderabad, India. 87

Banerjee, S., Grosan, C. & Abraham, A. (2005). Emotional ant based modeling of crowd dynamics. In *Seventh International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC 2005), 25-29 September 2005, Timisoara, Romania*, 279–286, IEEE Computer Society. 47

Batty, M. (2003). *Advanced Spatial Analysis: The CASA book of GIS*, chap. Agent-based Pedestrian Models, 81–106. Redlands: ESRI Press. 30

Batty, M. (2005). *Cities and Complexity: Understanding cities with Cellular Automata, Agent-based Models, and Fractals*. The MIT Press. Cambridge, MA. 30

Baxter, J., Tridgell, A. & Weaver, L. (1998). Knightcap: A chess program that learns by combining td(lambda) with game-tree search. In *Proceedings of the 15th International conference on Machine Learning (ICML'98)*, 28–36. 92

Bellman, R. (1957). *Dynamic Programming*. Princeton University Press. 57

Ben-Akiva, M. & Lerman, S.R. (1985). *Discrete Choice Analysis: Theory and Application to Travel Demand*. MIT Press. 17

Berrou, J.L., Beecham, J., Quaglia, P., Kagarlis, M.A. & Gerodimos, A. (2007). *Pedestrian and evacuation dynamics 2005*, chap. Calibration and validation of the Legion simulation model using empirical data, 167–180. Springer. 101

Bierlaire, M. (2003). BIOGEME: A free package for the estimation of discrete choice models. In *Proceedings of the 3rd Swiss Transportation Research Conference. Ascona, Switzerland*. 19

Bierlaire, M. & Robin, T. (2009). *Pedestrians Choices*, chap. Pedestrian Behavior, 1–25. Emerald. 17, 110, 114, 125, 221

Blue, V.J. & Adler, J.L. (2001). Cellular automata microsimulation for modeling bi-directional pedestrian walkways. *Transportation Research Part B: Methodological*, **35**, 293–312. 20

Blumberg, B., Downie, M., Ivanov, Y., Berlin, M., Johnson, M. & Tomlinson, B. (2002). Integrated learning for interactive synthetic characters. In *Proceedings of SIGGRAPH 2002*, 417–426. 4, 95

Bonabeau, E. (2007). Agent-based modelling: methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciencies (PNAS)*, **99**, 7280–7287. 28

Bongard, J.C. (2013). Evolutionary robotics. *Communications of the ACM*, **56**, 74–83. 4

Bonneaud, S. & Warren, W.H. (2012). A behavioral dynamics approach to modelling realistic pedestrian behavior. In *International Conference on Pedestrian and Evacuation dynamics*. 103

Borgers, A. & Timmermans, H. (1986). A model for pedestrian route choice and demand for retail facilities within inner-city shopping areas. *Geographical Analysis*, **18**, 115–128. 18

Bourgois, L., Saunier, J. & Auberlet, J.M. (2012). Towards contextual goal-oriented perception for pedestrian simulation. In *Proceedings of the 4th International Conference on Agents and Artificial Intelligence. ICAART 2012.*, vol. 2, 197–202. 119, 145

Boyan, J.A. & Moore, A.W. (1995). Generalization in reinforcement learning: Safely approximating the value function. In *Advances in Neural Information Processing Systems 7*, 369–376, MIT Press. 73

Brafman, R.I. & Tennenholtz, M. (2002). R-max–a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, **3**, 213–231. 70

Braun, A., Musse, S., de Oliveira, L. & Bodmann, B. (2003). Modeling individual behaviors in crowd simulation. In *Computer Animation and Social Agents (CASA)*, 143–148, IEEE Computer Society. Washington, DC, USA. 40

Burstedde, C., Klauck, K., Schadschneider, A. & Zittartz, J. (2001). Simulation of pedestrian dynamics using a two-dimensional cellular automaton. *Physica A*, **295**, 507–525. 20, 34

Busoniu, L., Babuska, R. & Schutter, B.D. (2008). A comprehensive survey of multi-agent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics Part C: Applications and Reviews*, **38**, 156–172. 72, 220

Busoniu, L., Babuska, R., Schutter, B.D. & Ernst, D. (2010). *Reinforcement Learning and Dynamic Programming Using Function Approximators*. CRC Press. 57, 58, 62, 63

Caruana, R. (1995). Multitask learning. *Machine Learning*, **28**, 41–75. 86

Castelletti, A., Pianosi, F. & Restelli, M. (2012). Tree-based fitted q-iteration for multi-objective markov decision problems. In *Neural Networks (IJCNN), The 2012 International Joint Conference on*, 1–8. 101

Chenney, S. (2004). Flow tiles. In *Proc. Eurographics/ACM SIGGRAPH Symposium on Computer Animation SCA'04. Grenoble. France*, 233–242. 44

Chernova, S. & Veloso, M.M. (2008). Learning equivalent action choices from demonstration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, September, 2008, Nice, France*, 1216–1221. 215

Chevaleyre, Y. & Pamponet, A. (2012). Adaptive probabilistic policy reuse. In T. Huang, Z. Zeng, C. Li & C. Leung, eds., *Neural Information Processing*, vol. 7665 of *Lecture Notes in Computer Science*, 603–611, Springer Berlin Heidelberg. 90

Chiu, C.C. & Soo, V.W. (2008). Cascading decomposition and state abstractions for reinforcement learning. In *Seventh Mexican International Conference on Artificial Intelligence, 2008. MICAI '08.*, 82–87. 152

Chowdhury, D., Nishinari, K. & Schadschneider, A. (2005). Physics of transport and and traffic phenomena in biology: From molecular motors and cells to organisms. *Physics of Life review*, **2**, 318–352. 23

Chraibi, M., Kemloh, U., Schadschneider, A. & Seyfried, A. (2011). Force-based models of pedestrian dynamics. *Networks and Heterogeneous Media*, **6**, 424–442. 27, 28

Claus, C. & Boutilier, C. (1998). The dynamics of reinforcement learning in cooperative multiagent systems. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, 746–752, AAAI Press. 72

Connell, J. & Mahadevan, S. (1997). *Robot Learning*, chap. Introduction to robot learning, 1–18. Kluwer Academic Publishers. 73

Courty, N. & Corpetti, T. (2007). Crowd motion capture. *Comput. Animat. Virtual Worlds*, **18**, 361–370. 44

Croonenborghs, T., Driessens, K. & Bruynooghe, M. (2007). Learning relational options for inductive transfer in relational reinforcement learning. In *Inductive Logic Programming, 17th International Conference, ILP 2007, Corvallis, OR, USA*, 88–97. 87

Daamen, W. (2004). *Modelling Passenger Flows in Public Transport Facilities*. Ph.D. thesis, Delft University of Technology, The Netherlands. 5, 12, 45

DAAMEN, W. & HOOGENDOORN, S. (2003). Experimental research of pedestrian walking behavior. In *Transportation Research Board Annual Meeting 2003*, 1–16, Washington, National Academy Press. 3, 164, 221

DAYAN, P. & SEJNOWSKI, T.J. (1994). Td(λ) converges with probability 1. *Machine Learning*, **14**, 295–301. 74

DE PAVIA, D., VIEIRA, R. & RAUPP MUSSE, S. (2005). Ontology-based crowd simulation for normal life situations. In *Computer Graphics International 2005*, 221–226. 48

DEFFUANT, G., NEAU, D., AMBLARD, F. & WEISBUCH, G. (2000). Mixing beliefs among interacting agents. *Advances in Complex Systems*, **03**, 87–98. 30

DEMPSTER, A., LAIRD, N. & RUBIN, D. (1977). Maximum -likelihood from incomplete data via the em algorithm. *J. Royal Statistical Society Set. B (methodological)*, **39**, 1–38. 87

DEN BERG, J.Q.V. & BOUVY, H.B. (1994). Pedestrian flow (survey of literature). Literature research, Delft University of Technology. Faculty of Civil Engineering. 13

DIGNUM, F. (2012). Agents for games and simulations. *Autonomous Agents and Multi-Agent Systems*, **24**, 217–220. 2

DINERSTEIN, J., EGBERT, P.K. & VENTURA, D. (2007). Learning policies for embodied virtual agents through demonstration. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence, IJCAI'07*, 1257–1262, Morgan Kaufmann Publishers Inc. 215

DRUMWRIGHT, E., HSU, J., KOENIG, N. & SHELL, D. (2010). Extending open dynamics engine for robotics simulation. In *Simulation, Modeling, and Programming for Autonomous Robots*, vol. 6472 of *Lecture Notes in Computer Science*, 38–50, Springer Berlin Heidelberg. 113

DUDA, R.O. & HART, P.E. (1973). *Pattern Classification and Scene Analysis*. John Wiley and Sons. New York. 75

Ekker, R., van der Werf, E. & Schomaker, L. (2004). Dedicated td-learning for stronger gameplay: Applications to go. In *Proceedings of the 13th Belgian-Dutch Conference on Machine Learning*, 46–52. 92

El-Fakdi, A. & Carreras, M. (2008). Policy gradient based reinforcement learning for real autonomous underwater cable tracking. In *International Conference on Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ*, 3635–3640. 91

Ennis, C., Peters, C. & O'Sullivan, C. (2008). Perceptual evaluation of position and orientation context rules for pedestrian formations. In *Proceedings of the 5th Symposium on Applied Perception in Graphics and Visualization*, APGV '08, 75–82, ACM, New York, NY, USA. 103

Ertel, W., Schneider, M., Cubek, R. & Tokic, M. (2009). The teaching-box: A universal robot learning framework. In *International Conference on Advanced Robotics, 2009. ICAR 2009.*, 1–6. 96

Escoffier, C., de Rigal, J., Rochefort, A., Vasselet, R., Lévêque, J.L. & Agache, P.G. (1989). Age-related mechanical properties of human skin: An in vivo study. *Journal of Investigative Dermatology*, 353–357. 123

Esfahani, A. & Analoui, M. (2008). Widest k-shortest paths q-routing: A new qos routing algorithm in telecommunication networks. In *Computer Science and Software Engineering, 2008 International Conference on*, vol. 4, 1032–1035. 92

Even-Dar, E. & Mansour, Y. (2003). Learning rates for q-learning. *Journal of Machine Learning Research*, **5**, 1–25. 70

Eves, F., Webb, O. & Mutrie, N. (2006). A workplace intervention to promote stair climbing: Greater effects in overweight. *Obesity*, **14**, 2210–2216. 18

Ewing, R., Schroeer, W. & Greene, W. (2007). School location and student travel analysis of factors affecting model choice. *TransportationResearch Board of the NAtional Academies*, **1895**, 55–63. 18

Fern, A., Yoon, S. & Givan, R. (2006). Approximate policy iteration with a policy language bias: Solving relational markov decision processes. *AJournal of Artificial Intelligence Research*, **25**, 75–118. 86

Fernández, F. & Borrajo, D. (2008). Two steps reinforcement learning. *International Journal of Intelligent Systems*, **23**, 213–245. 78, 80, 146

Fernández, F. & Parker, L.E. (2001). Learning in large cooperative multi-robot domains. *International Journal of Robotics Automat.*, **16**, 217–226. 142

Fernández, F. & Veloso, M. (2006). Probabilistic policy reuse in a reinforcement learning agent. In *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '06, 720–727, ACM, New York, NY, USA. 88, 89

Fernández, F. & Veloso, M. (2013). Learning domain structure through probabilistic policy reuse in reinforcement learning. *Progress in Artificial Intelligence*, **2**, 13–27. 90

Fernández, F., Borrajo, D. & Parker, L.E. (2005). A reinforcement learning algorithm in cooperative multi-robot domains. *Journal of Intelligent and Robotic Systems*, **43**, 161–174. 72, 142, 224

Fernández, F., García, J. & Veloso, M. (2010). Probabilistic policy reuse for inter-task transfer learning. *Robotics and Autonomous Systems*, **58(7)**, 866–871. 88, 152

Feurtey, F. (2000). *Simulating the collision Avoidance Behavior of Pedestrians*. Ph.D. thesis, University of Tokio. Department of Electronic Engineering. 42

Foster, D. & Dayan, P. (2004). Structure in the space of value functions. *Machine Learning*, **49**, 325–346. 87

Foster, I. (1995). *Designing and Building Parallel Programs*. Addison-Wesley Publishing. 115

Frampton, M. & Lemon, O. (2008). Using dialogue acts to learn better repair strategies for spoken dialogue systems. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, 5045–5048. 92

Fredslund, J. & Matarić, M.J. (2002). A general algorithm for robot formations using local sensing and minimal communication. *IEEE Transactions on Robotics and Automation*, **18**, 837–846. 4

Fridman, N. & Kaminka, G. (2010). Modeling pedestrian crowd behavior based on a cognitive model of social comparison theory. *Computational and Mathematical Organization Theory*, **16**, 348–372. 34

Fruin, J.J. (1971a). Designing for pedestrians: A level-of-service concept. *Highway Research Record*, **355**, 1–15. 9, 11

Fruin, J.J. (1971b). *Pedestrian Planning and Design*. Metropolitan Association of Urban Designers and Environment Planners, Inc. New York. 11, 118

García, J. & Fernández, F. (2012). Safe exploration of state and action spaces in reinforcement learning. *Journal of Artificial Intelligence Research*, **45**, 515–564. 75, 149

García, J., López-Bueno, I., Fernández, F. & Borrajo, D. (2010). *A Comparative Study of Discretization Approaches for State Space Generalization in the Keepaway Soccer Task. In Reinforcement Learning: Algorithms, Implementations and Aplications*. Nova Science Publishers. 73, 78, 80, 100, 146, 152

García, J., Borrajo, F. & Fernández, F. (2012). Reinforcement learning for decision-making in a business simulator. *International Journal of Information Technology and Decision Making*, **11**, 935–960. 92, 146

García-Polo, F. & Fernández, F. (2011). Safe reinforcement learning in high-risk tasks through policy improvement. In *IEEE Symposium on Adaptive Dynamic Programming And Reinforcement Learning (ADPRL), 2011*, 76–83. 149

Garey, M.R. & Johnson, D.S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freemen. New York. 75

Gayle, R., Moss, W., Lin, M.C. & Manocha, D. (2009). Multi-robot coordination using generalized social potential fields. In *IEEE International Conference on Robotics and Automation (ICRA'09)*. 40

Gibson, J. (1979). *The ecological Approach to Visual Perception*. Houghton Mifflin. Boston MA. 37

Gilbert, N. (2007). *Agent-Based Models*. Sage Publications: London. 30, 41

Gilman, M., Moldovan, H. & Tencer, M. (2005). *Pedestrian and Evacuation Dynamics 2005*, chap. Dynamic Navigation Field- A Local and O-Demand Family of Algorithms for Wayfinding. Springer. 23

Gipps, P.G. & Marsjo, B. (1985). A microsimulation model for pedestrian flows. *Math Comp Sim*, **27**, 95–105. 3, 20

Gratch, J., Marsella, S., Wang, N. & Stankovic, B. (2009). Assessing the validity of appraisal-based models of emotions. In *Proc. IEEE International Conference on Affective Computing and Intelligent Interaction (ACII'09)*, 1–8, IEEE. 34

Gray, R.M. (1984). Vector quantization. *IEEE ASSP Magazine*, **1**, 4–29. 74

Greenlaw, R., Hoover, H.J. & Ruzzo, W.L. (1995). *Limits to Parallel Computation: P-Completeness Theory*. Oxford University Press. 71

Griffith, S., Subramanian, K., Scholz, J., Isbell, C. & Thomaz, A.L. (2013). Policy shaping: Integrating human feedback with reinforcement learning. In C. Burges, L. Bottou, M. Welling, Z. Ghahramani & K. Weinberger, eds., *Advances in Neural Information Processing Systems 26*, 2625–2633, Curran Associates, Inc. 215

Gu, Q. & Deng, Z. (2011). Context-aware motion diversification for crowd simulation. *IEEE Computer Graphics and Applications*, **31**, 54–65. 49

Gutierrez, M., Vexo, F. & Thalmann, D. (2005). Semantics-based representation of virtual environments. *J. Computer Applications in Technology*, **23**, 229–238. 48

Guy, S., Curtis, S., Lin, M. & Manocha, D. (2012). Least-effort trajectories lead to emergent crowd behaviors. *Physics Review E*, **85**, 0161100–0161107. 32, 33, 100

Guy, S.J., Chhugani, J., Kim, C., Satish, N., Lin, M.C., Manocha, D. & Dubey, P. (2009). Clearpath: highly parallel collision avoidance for multi-agent simulation. In D.W. Fellner & S.N. Spencer, eds., *Symposium on Computer Animation*, 177–187, ACM. 42

Hartmann, D. (2010). Adaptive pedestrian dynamics based on geodesics. *New Journal of Physics*, **12**, 043032. 23, 34

Heidemann, D. (1997). Queueing at unsignalized intersections. *Transportation Research Part B*, **31**, 239–263. 21

Heïgeas, L., Luciani, A., Thollot, J. & Castagné, N. (2003). A physicallybased particle model of emergent crowd behaviors. In *Proc. Graphikon 2003*. 39, 121

Helbing, D. (1992a). A fluid-dynamic model for the movement of pedestrians. *Complex Systems*, **28**, 391–415. 24, 25

Helbing, D. (1992b). *Stochastische Methoden, nichtlineare Dynamik ind quantitative Modelle sozialer Prozesse*. Ph.D. thesis, University of Stuttgart. Germany. 26

Helbing, D. (2004). Collective phenomena and states in traffic and self-driven many-particle systems. *Computational materials science*, **30**, 180–187. 98, 119, 172

Helbing, D. & Johansson, A. (2009). *Encyclopedia of Complexity and Systems Science*, vol. 16, chap. Pedestrian, Crowd and Evacuation Dynamics, 6476–6495. Springer. 26, 28, 34, 100, 143

HELBING, D. & MOLNÁR, P. (1995). Social force model for pedestrian dynamics. *Phys. Rev. E*, **51**, 4282–4286. 3, 13, 26, 221, 224

HELBING, D., MOLNAR, P. & SCHWEITZER, F. (1997). Computer simulation of pedestrian dynamics and trail formation. *Evolution of Natural Structures*, 229–234. 14, 39

HELBING, D., FARKAS, I. & VICSEK, T. (2000). Simulating dynamical features of escape panic. *Nature*, **407**, 487–490. 27, 170, 193, 198

HELBING, D., MOLNÁR, P., FARKAS, I.J. & BOLAY, K. (2001). Self-organizing pedestrian movement. *Environment and Planning B: Planning and Design*, 361–383. 120, 174

HELBING, D., BUZNA, L., JOHANSON, A. & WERNER, T. (2005). Self-organized pedestrian crowd dynamics: Experiments, simulations, and design solutions. *Transportation Science*, **39**, 1–24. 27, 34, 144, 216, 226

HELBING, D., JOHANSSON, A. & AL-ABIDEEN, H.Z. (2007). The dynamics of crowd disasters: An empirical study. *Physical Review E*, **75**, 046109. 132

HENDERSON, L. (1971). The statistics of crowd fluids. *Nature*, **229**, 381–383. 31

HENDERSON, L. (1974). On the fluid mechanics of human crowd motion. *Transportation Research*, **8**, 509–515. 24, 34

HERMAN, I.P. (2007). *Physics of the Human Body*. Springer. 121, 123

HERNANDEZ-LERMA, O. (1989). *Adaptive Markov Control Processes*. Springer-Verlag. 61

HESTER, T. & STONE, P. (2012). Intrinsically motivated model learning for a developing curious agent. In *AAMAS Adaptive Learning Agents (ALA) Workshop*. 94

HOOGENDOORN, S. & BOVY, P. (2004). Pedestrian route-choice and activity scheduling theory and models. *Transportation Research Part B: Methodology*, **38**, 169–190. 18, 31

HOOGENDOORN, S.P., BOVY, P.H.L. & DAAMAEN, W. (2001). *Pedestrian and Evacuation Dynamics*, chap. Microscopic Pedestrian Wayfinding and Dynamics Modelling. Springer. 13

HOOGERNDOORN, S.P. & DAAMEN, W. (2009). A novel calibration approach of microscopic pedestrian models. In H. Timmermans, ed., *Pedestrian Behavior*, 195–214, Emerald. 127

HOOGERNDOORN, S.P., DAAMEN, W. & BOVY, P.H.L. (2003). Extracting microscopic pedestrian characteristics from video data. In *Transportation Research Board. Annual Meeting*, CD–Rom. 15, 16

HOWARD, R.A. (1960). *Dynamic Programming and Markov processes*. The MIT Press. Cambridge. Mass. 57

HUGHES, R.L. (2003). The flow of human crowds. *Annu. Rev. Fluid Mech.*, **35**, 169–182. 25

IKEMOTO, L., ARIKAN, O. & FORSYTH, D. (2005). Learning to move autonomously in a hostile world. In *ACM SIGGRAPH 2005 Sketches*, SIGGRAPH '05, ACM, New York, NY, USA. 93

IZQUIERDO, S. (2007). The impact of quality uncertainty without assymetric information on market efficiency. *Journal of Business Research*, **60.2007**, 858–867. 30

JAAKKOLA, T., JORDAN, M.I. & SINGH, S. (1994). On the convergence of stochastic iterative dynamic programming algorithms. *Neural Computation*, **6**, 1185–1201. 65

JOHANSSON, A. (2009). Constant-net-time headway as a key mechanism behind pedestrian flow dynamics. *Physical Review E*, **80**, 026120. 13

JOHANSSON, A. & KRETZ, T. (2012). *Agent-Based Models of Geographical Systems*, chap. Applied Pedestrian Modeling. Springer. 3, 36, 41, 184

Johansson, A., Helbing, D. & Shukla, P.K. (2007). Specification of the social force pedestrian model by evolutionary adjustment to video tracking data. *Advances in Complex Systems*, **10**, 271–288. 13, 119

Kaelbling, L.P., Littman, M.L. & Moore, A.W. (1996). Reinforcement learning: A survey. *Int. Journal of Artificial Intelligence Research*, **4**, 237–285. 57, 220

Kalyanakrishnan, S. & Stone, P. (2007). Batch reinforcement learning in a complex domain. In *The Sixth International Joint Conference on Autonomous Agents and Multiagent Systems*, 650–657, ACM, New York, NY, USA. 214, 226

Kanungo, T., Mount, D.M., Netanyahu, N., Piatko, C., Silverman, R. & Wu, A.Y. (2002). An efficient k-means clustering algorithm: Analysis and implementation. *IEEE transactions on Pattern Analysis and Machine Intelligence*, **24**, 881–892. 75

Karamouzas, I. & Overmars, M. (2012). Simulating and evaluating the local behavior of small pedestrian groups. *IEEE Transactions on Visualization and Computer Graphics*, **18**, 394–406. 49

Karamouzas, I., Heil, P., Beek, P. & Overmars, M.H. (2009). A predictive collision avoidance model for pedestrian simulation. In *Proceedings of the 2Nd International Workshop on Motion in Games*, MIG '09, 41–52, Springer-Verlag, Berlin, Heidelberg. 42

Kearns, M. & Singh, S. (2002). Near-optimal reinforcement learning in polynomial time. *Machine Learning*, **49**, 209–232. 70

Kessel, A., Klüpfel, H., Wahle, J. & Schreckenberg, M. (2002). *Microscopic Simulation of Pedestrian Crowd Motion*, chap. Pedestrian and Evacuation Dynamics, 193–200. Springer. 15

Kim, I., Galiza, R. & Ferreira, L. (2013a). Modeling pedestrian queueing using micro-simulation. *Transportation Research Part A: Policy and Practice*, **49**, 232–240. 22

Kim, S., Guy, S.J. & Manocha, D. (2013b). Velocity-based modeling of physical interactions in multi-agent simulations. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '13, 125–133, ACM, New York, NY, USA. 216

Kirchner, A. & Schadscheneider, A. (2002). Simulation of evacuation processes using a bionics-inspired cellular automaton model for pedestrian dynamics. *Physica A*, **312**, 260–276. 20

Kitazawa, K. & Batty, M. (2004). Pedestrian behaviour modelling an application to retail movements using a genetic algorithm. In *Proceedings of the 7th International Conference on design and Decission Support Systems in architectural and Urban Planning*. 4

Knoblauch, R., Pietrucha, M. & Nitzburg, M. (2007). Field studies of pedestrian walking speed ann start-up time. *Transportation Research Record: Journal of the Transportation Research Board*, **1538**, 27–38. 18

Knox, W.B. & Stone, P. (2008). Tamer: Training an agent manually via evaluative reinforcement. In *Proceedings of the 7th IEEE ICDL*, 292–297. 215

Kochenderfer, M.J. (2006). *Adaptive Modelling and Planning for Learning Intelligent Behaviour*. Ph.D. thesis, University of Edinburgh. College of Science and Engineering. School of Informatics. 95

Koh, W.L. & Zhou, S. (2011). Modeling and simulation of pedestrian behaviors in crowded places. *ACM Trans. Model. Comput. Simul.*, **21**, 20:1–20:23. 98

Kohl, N. & Stone, P. (2004). Policy gradient reinforcement learning for fast quadrupedal locomotion. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2004* , 2619–2624, IEEE. 91

Korein, J.U. & Badler, N.I. (1982). Techniques for generating the goal-directed motion of articulated structures. *IEEE Comput. Graph. Appl.*, **2**, 71–81. 35

Kovar, L., Gleicher, M. & Pighin, F. (2002). Motion graphs. *ACM Trans. Graph.*, **21**, 437–482. 92

KRETZ, T. (2007). *Pedestrian Traffic. Simulations and Experiments*. Ph.D. thesis, Department of Physics of the University of Duisburg-Essen. Germany. 16, 19

KRETZ, T. (2009). Pedestrian traffic: on the quickest path. *Journal of Statistical Mechanics: Theory and Experiment*, **3**, p03012. 22

KROON, M. & WHITESON, S. (2009). Automatic feature selection for model-based reinforcement learning in factored mdps. In *International Conference on Machine Learning and Applications. ICMLA'09*, 324–330. 110

KUFFNER, J. (1999). Fast synthetic vision, memory, and learning models for virtual humans. In *Computer Animation*, 118–127. 38

KUHLMANN, G. & STONE, P. (2007). Graph-based domain mapping for transfer learning in general games. In *Proceedings of the 18th European Conference on Machine Learning*. 88

KWIATKOWSKA, M., FRANKLIN, S., HENDRIKS, C. & KWIATKOWSKI, K. (2009). Friction and deformation behaviour in human skin. *Wear*, 1264–1273. 124

LAIRD, J.E., ROSENBLOOD, P.S. & NEWELL, A. (1986). Chunking in soar: the anatomy of a general learning mechanism. *Machine Learning*, **1**, 11–46. 86

LAM, W.H.K. & CHEUNG, C.Y. (2000). Pedestrian speed-flow relationships for walking facilities in honng-kong. *Journal of Transportation Engineering*, **126**, 343–349. 18

LÄMMEL, G., KLÜPFEL, H. & NAGEL, K. (2009). *The MATSim Network Flow Model for Traffic Simulation Adapted to Large-Scale Emergency Egress and Application to the Evacuation of the Indonesian City of Padang in Case of a Tsunami Warning*, chap. Pedestrian Behavior, 244–265. Emerald. 12

LANE, T., RIDENS, M. & STEVENS, S. (2007). Reinforcement learning in non-stationary environment navigation tasks. In *Advances in Artificial Intelligence (LNCS 4509)*, 429–440, Springer. 110

Lange, S., Gabel, T. & Riedmiller, M. (2012). Batch reinforcement learning. In *Reinforcement Learning*, 45–73, Springer, Berlin, Heidelberg. 214, 226

Lauer, M. & Riedmiller, M. (2000). An algorithm for distributed reinforcement learning in cooperative multi-agent systems. In *Proceedings of the Seventeenth International Conference On Machine Learning*, 535–542, Morgan-Kaufmann. 72

LaValle, S.M. (2006). *Planning Algorithms*. Cambridge. 24

Lazaric, A. (2012). *Reinforcement Learning: State-of-the-art*, chap. Transfer in Reinforcement Learning: a framework and a survey, 143–173. Springer. 86

Lee, J. & Lee, K.H. (2004). Precomputing avatar behavior from human motion data. In *Eurographics/ACM SIGGRAPH Symposium on Computer Animation. R. Boulic, D.K. Pai (Eds.)*, 79–87. 94

Lee, K.H., Choi, M.G., Hong, Q. & Lee, J. (2007). Group behavior from video: A data-driven approach to crowd simulation. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '07, 109–118, Eurographics Association. 44

Lee, S.J. & Popović, Z. (2010). Learning behavior styles with inverse reinforcement learning. *ACM Trans. Graph.*, **29**, 122:1–122:7. 94

Lee, Y., Lee, S.J. & Popović, Z. (2009). Compact character controllers. *ACM Trans. Graph.*, **28**, 169:1–169:8. 93

Lemercier, S., Jelic, A., Kulpa, R., Hua, J., Fehrenbach, J., Degond, P., Appert-Rolland, C., Donikian, S. & Pettré, J. (2012). Realistic following behaviors for crowd simulation. *Comput. Graph. Forum*, **31**, 489–498. 41, 103

Leng, B., Wang, J., Zhao, W. & Xiong, Z. (2014). An extended floor field model based on regular hexagonal cells for pedestrian simulation. *Physica A*, **In Press**. 22

Levine, S. & Popović, J. (2012). Physically plausible simulation for character animation. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '12, 221–230, Eurographics Association. 103

Levine, S., Lee, Y., Koltun, V. & Popović, Z. (2011). Space-time planning with parameterized locomotion controllers. *ACM Trans. Graph.*, **30**, 23:1–23:11. 93

Li, D.W. & Han, B.M. (2011). Modeling queue service system in pedestrian simulation. *Advanced Materials Research*, **187**, 1–6. 21

Li, W., Xu, G., Wang, Z. & Xu, Y. (2008). Dynamic energy management for hybrid electric vehicle based on approximate dynamic programming. In *Intelligent Control and Automation, 2008. WCICA 2008. 7th World Congress on*, 7864–7869. 92

Linde, Y., Buzo, A. & Gray, R. (1980). An algorithm for vector quantizer design. *IEEE Trans. on Commun.*, **28**, 84–95. 75

Lister, W.D. & Day, A. (2012). Technical section: Stream-based animation of real-time crowd scenes. *Comput. Graph.*, **36**, 651–657. 47

Littman, M.L. (1994). Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the 11th International Conference On Machine Learning*, 157–163. 72

Littman, M.L. (2001). Value-function reinforcement learning in markov games. *Cognitive Systems Research*, **2**, 55–66. 72

Liu, Y. & Stone, P. (2006). Value-function-based transfer for reinforcement learning using structure mapping. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, 415–20. 88

Lloyd, S.P. (1982). Least squares quantization in pcm. *IEEE transactions on Information Theory*, **28**, 129–137. 75

Lo, W.Y., Knaus, C. & Zwicker, M. (2012). Learning motion controllers with adaptive depth perception. In *Proc. Eurographics/ACM SIGGRAPH Symposium on Computer Animation SCA'12. Lausanne, Switzerland*, 145–154. 94

Lovas, G.G. (1994). Modeling and simulation of pedestrian traffic flow. *Transportation Research Part B: Methodological*, **28**, 429–443. 21, 221

Lozano, M., Morillo, P., Orduña, J.M., Cavero, V. & Vigueras, G. (2009). A new system architecture for crowd simulation. *J. Network and Computer Applications*, **32**, 474–482. 48

Magnenat-Thalmann, N., Kalra, P., Leveque, J., Bazin, R., Batisse, D. & Querleux, B. (2002). A computational skin model: Fold and wrinkle formation. *IEEE Transactions on Information Technology in Biomedicine*, **6**, 317–323. 121, 123

Mahadevan, S. & J.Connell (1991). Scaling reinforcement learning to robotics by exploiting the subsumtion architecture. In *Proceedings of the 8th International workshop on Machine Learning*, volume 1 773–780, Morgan Kaufmann. 80

Manenti, L. & Manzoni, S. (2011). Crystals of crowd: Modelling pedestrian groups unsing mas-based approach. In *12th Workshop on Objects and Agents, Rende (italy)*, 51–57, CEUR. 34

Martinez-Gil, F., Barber, F., Lozano, M., Grimaldo, F. & Fernández, F. (2010). A reinforcement learning approach for multiagent navigation. In *ICAART 2010 - Proceedings of the International Conference on Agents and Artificial Intelligence, Volume 1 - Artificial Intelligence, Valencia, Spain*, 607–610, INSTICC Press. 107, 142

Martinez-Plumed, F., Ferri, C., Hernández-Orallo, J. & Ramírez-Quintana, M.J. (2013). Policy reuse in a general learning framework. In *Fiftennth Spanish Conference for Artificial Intelligence (CAEPIA 2013). Madrid*. 90

Masen, M. (2011). A systems based experimental approach to tactile friction. *Journal of the Mechanical Behavior of Biomedical Materials*, 1620–1626. 125

Matarić, M.J. (1994). Learning to behave socially. In *From Animals to Animats: International Conference on Simulation of Adaptive Behavior*, 453–462, MIT Press. 72

Matarić, M.J. (1997). Reinforcement learning in the multi-robot domain. *Auton. Robots*, **4**, 73–83. 91

May, A.D. (1990). *Traffic Flow Fundamental*. Prentice Hall. 10

McCann, J. & Pollard, N.S. (2007). Responsive characters from motion fragments. *ACM Transactions on Graphics (SIGGRAPH 2007)*, **26**. 4, 93

McFadden, D. (1981). *Econometric Models of Probabilistic Choice*, chap. Structural Analysis of Discrete Data with Econometric Applications, 198–269. MIT Press. 4, 17

Mehran, R., Oyama, A. & Shah, M. (2009). Abnormal crowd behavior detection using social force model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR'09. Miami. USA*, 935–942. 27

Melo, F. & Veloso, M. (2009). Learning of coordination: Exploiting sparse interactions in multiagent systems. In *Proceedings of the 8th International Conference On Autonomous Agents and Multiagent Systems (AAMAS09)*, 773–780. 72

Merrick, K. & Maher, M.L. (2006). Motivated reinforcement learning for non-player characters in persistent computer game worlds. In *Proceedings of the 2006 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology*, ACE '06, ACM, New York, NY, USA. 95

Merrick, K.E. & Maher, M.L. (2007). Motivated reinforcement learning for adaptive characters in open-ended simulation games. In *Proceedings of the International Conference on Advances in Computer Entertainment Technology*, ACE '07, 127–134, ACM, New York, NY, USA. 95

Merrick, K.E. & Maher, M.L. (2009). *Motivated Reinforcement Learning: Curious Characters for Multiuser Games*. Springer. London. 95

Metoyer, R.A. & Hodgins, J.K. (2004). Reactive pedestrian path following from examples. *The Visual Computer*, **20**, 635–649. 214

Meyer-König, T., Küpfel, H. & Schreckenberg, M. (2001). A microscopic model for simulating mustering evacuation processes onboard passenger ships. In *Proceedings of the International Emergency Management Society Conference*. 20

Monzani, J.S., Guye-Vuilleme, A. & de Sevin, E. (2004). *Handbook of Virtual Humans*, chap. Behavioral Animation, 260–286. Springer. 35

Moody, J.E. & Saffell, M. (2001). Learning to trade via direct reinforcement. *IEEE Transactions on Neural Networks*, **12**, 875–889. 91

Moore, A.W. & Atkeson, C.G. (1993). Prioritized sweeping: reinforcement learning with less data and less time. *Machine Learning*, **13**, 103–130. 62

Mori, M. & Tsukaguchi, H. (1987). A new method for evaluation of level of service in pedestrian facilities. *Transportation Research part A*, **21**, 223–234. 134, 137

Moussaïd, M., Helbing, D. & Theraulaz, G. (2011). How simple rules determine pedestrian behavior and crowd disasters. In *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, National Academy of Sciences. 35, 49

Musse, S.R. & Thalmann, D. (1997). A model of human crowd behavior: Group interrelationship and collision detection analysis. In *Proc. Workshop of Computer Animation and Simulation of Eurographics'97*, 39–51. 37

Musse, S.R. & Thalmann, D. (2001). Hierarchical model for real time simulation of virtual human crowds. *IEEE Transactions on Visualization and Computer Graphics*, **7**, 152–164. 47

Musse, S.R., Babski, C., Capin, T. & Thalmann, D. (1998). Crowd modelling in collaborative virtual environments. In *ACM symposium on Virtual Reality Software and Technology*, 115–123, ACM. New York. 3, 28

Musse, S.R., Jung, C.R., Silveira-Jacques, J.C. & Braun, A. (2007). Using computer vision to simulate the motion of virtual agents. *Journal of Visualization and Computer Animation*, **18**, 83–93. 44

Nagel, K., Barret, C. & Rickert, M. (1996a). Parallel traffic microsimulation by cellular automata and application for large scale transportation modeling. Tech. Rep. 96:0050, Los Alamos National Laboratory. New Mexico. 20

Nagel, K., Schreckenberg, M. & Latour, A. (1996b). Two-lane traffic simulation using cellular automata. *Physica A*, **231**, 534. 19

Narain, R., Golas, A., Curtis, S. & Lin, M.C. (2009). Aggregate dynamics for dense crowd simulation. *ACM Trans. Graph.*, **28**, 122:1–122:8. 44

Navin, P.D. & Wheeler, R.J. (1969). Pedestrian flow characteristics. *Traffic Engineering*, **39**, 31–36. 13

Nelson, H.E. & Mowrer, F.W. (2002). *SFPE handbook of fire protection engineering, 3rd. edition*, chap. Emergency Movement, 367. National Fire Protection Association. 13

Ng, A.Y., Harada, D. & Russell, S. (1999). Policy invariance under reward transformations: Theory and application to reward shaping. In *In Proceedings of the Sixteenth International Conference on Machine Learning*, 278–287, Morgan Kaufmann. 215, 226

Nguyen, T., Li, Z., Silander, T. & Leong, T.Y. (2013). Online feature selection for model-based reinforcement learning. *Journal of Machine Learning Research*, **28**, 498–506. 110

Nicolescu, M.N. & Matarić, M.J. (2003). Natural methods for robot task learning: instructive demonstrations, generalization and practice. In *Proceed-*

ings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS 2003, Melbourne, Australia, 241–248. 215, 226

Nishinari, K., Kirchner, A., Namazi, A. & Schadschneider, A. (2004). Extended floor field ca model for evacuation dynamics. *IEICE Transactions*, **87-D**, 726–732. 17

Noser, H., Renault, O., Thalmann, D. & Thalmann, N.M. (1995). Navigation for digital actors based on synthetic vision, memory and learning. *Computers and Graphics*, **19**, 7–19. 38

Nouri, A. & Littman, M.L. (2008). Multi-resolution exploration in continuous spaces. In *Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems.*, 1209–1216. 64

Okada, E. & Asami, Y. (2007). A pedestrian route choice model to evaluate alternative plans for regeneration of Galata region. *ARI Bulletin of the Istambul Technical University*, **55**, 11–32. 18

Okaya, M. & Takahashi, T. (2011). BDI agent model based evacuation simulation (demonstration). In *Proceedings of the 10th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2011)*, 1297–1298, IFAAMAS. 31

Okazaki, S. & Matsushita, S. (1993). A study of simulation model for pedestrian movement with evacuation and queueing. In *International Conference on Engineering for Crowd Safety*, 271–280. 22

Olivier, A.H., Marin, A., Crétual, A. & Pettré, J. (2012). Minimal predicted distance: A common metric for collision avoidance during pairwaise interactions between walkers. *Gait and Posture*, **36**, 399–404. 42

Ondrej, J. (2011). *Modeling and Planning a Realistic Navigation for Autonomous Virtual Humans*. Ph.D. thesis, INSA Rennes. Université Européenne de Bretagne. 145

Ondrej, J., Pettré, J., Olivier, A.H. & Donikian, S. (2010). A synthetic-vision based steering approach for crowd simulation. *ACM Trans. Graph.*, **29**. 38, 110

O'Sullivan, D. & Haklay, M. (2000). Agent based models and individualism: Is the world agent-based? *Environment and Planning A*, 1409–1425. 3, 28

Pan, S.J. & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, **22**, 1345–1359. 85

Pan, X., C.S.Han & Law, K. (2005). A multi-agent based simulation framework for the study of human and social behavior in egress analysis. In *Proceedings of the International Conference on Computing in Civil Engineering*, 12–15. 33

Papadimitriou, C.H. & Tsitsiklis, J.N. (1987). The complexity ofmarkov decision processes. *Mathematics of Operations Research*, **12**, 441–450. 71

Paris, S., Pettré, J. & Donikian, S. (2007). Pedestrian reactive navigation for crowd simulation: a predictive approach. *Comput. Graph. Forum*, **26**, 665–674. 41, 216

Patil, S., den Berg, J.V., Curtis, S., Lin, M.C. & Manocha, D. (2011). Directing crowd simulations using navigation fields. *IEEE Transactions on Visualization and Computer Graphics*, **17**, 244–254. 5, 43

Pauls, J.L. (1977). *Human Response to Tall Buildings*, chap. Movement of People in Building Evacuations, 281–292. Hutchinson and Ross, Inc. 9

Pelechano, N. & Badler, N. (2006). Modeling crowd and trained leader behavior during building evacuation. *IEEE Computer Graphics and Applications*, **26**, 80–86. 27, 40

Pelechano, N., Allbeck, J. & Badler, N. (2007). Controlling individual agents in high-density crowd simulation. In *Proc. ACM/SIGGRAPH/Eurographics Symp. Computer Animation*, 99–108. 3, 5, 34, 40, 184, 216, 226

Pelechano, N., Allbeck, J. & Badler, N. (2008). *Virtual Crowds. Methods, Simulation and Control*. Morgan Claypool Publishers. 10, 40, 46, 49

Perkins, T.J. & Precup, D. (1999). Using options for knowledge transfer in reinforcement learnng. Tech. Rep. UM-CS-1999-034, The University of Massachusetts at Amhrest. 86

Peters, C. & Ennis, C. (2009). Modeling groups of plausible virtual pedestrians. *IEEE Comput. Graph. Appl.*, **29**, 54–63. 103

Peters, C. & O' Sullivan, C. (2003). Bottom-up visual attention for virtual human animation. In *Proceedings of the 16th International Conference on Computer Animation and Social Agents (CASA 2003)*, 111–, IEEE Computer Society, Washington, DC, USA. 38

Pettré, J., Ciechomski, P.d.H., Maïm, J., Yersin, B., Laumond, J.P. & Thalmann, D. (2006). Real-time navigating crowds: Scalable simulation and rendering: Research articles. *Comput. Animat. Virtual Worlds*, **17**, 445–455. 47

Pettré, J., Grillon, H. & Thalmann, D. (2007). Crowds of moving objects: Navigation planning and simulation. In *2007 IEEE International Conference on Robotics and Automation, ICRA 2007, 10-14 April 2007, Roma, Italy*, 3062–3067. 43

Pettré, J., Ondrej, J., Olivier, A., Creutal, A. & Donikian, S. (2009). Experiment-based modeling, simulation and validation of interactions between virtual walkers. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 189–198, ACM, New York. 42, 216, 226

Predtechenskii, V.M. & Milinskii, A.I. (1978). *Planning for foot traffic flow in buildings*. New Dehli: Amerind Publishing. 9, 12, 13

Prigogine, I. & Herman, R. (1971). *Kinetic Theory for Vehicular Traffic*. Elsevier. 24

Proper, S. & Tadepalli, P. (2006). Scaling model-based average-reward reinforcement learning for product delivery. In *Proceedings of the European Conference on Machine Learning (ECML'06). Lecture Notes on Artificial Intelligence 4212*, 735–742. 91

Qiu, F. & Hu, X. (2013). Spatial activity-based modeling for pedestrian crowd simulation. *Simulation*, **89**, 451–465. 48

Qiu, F., Hu, X., Wang, X. & Karmakar, S. (2010). Modeling social group structures in pedestrian crowds. *Simulation Modelling Practice and Theory*, **18**, 109–205. 34

Rahman, K., Ghani, N.A., Kamil, A.A., Mustafa, A. & Chowdhury, M.K. (2013). Modelling pedestrian travel time and the design of facilities: A queueing approach. *PLoS ONE*, **8**, e63503. 21

Ramalho, A., Silva, C., Pais, A. & Sousa, J. (2007). In vivo friction study of human skin: Influence of moisturizers on differen anatomical sites. *Wear*, 1044–1049. 125

Ramming, M. (2002). *Network Knowledge and Route Choice*. Ph.D. thesis, Massachussetts Institute of Technology, Mass., USA. 32

Randlov, J. & Alstrom, P. (1998). Learning to drive a bicycle using reinforcement learning and shaping. In J.W. Shavlik, ed., *Proceedings of the Fifteenth International Conference on Machine Learning (ICML 1998)*, 463–471, Morgan Kauffman, San Francisco, CA, USA. 215

Ranjan, R. & Phophalia, A. (2008). Reinforcement learning for dynamic channel allocation in mobile cellular systems. In *Recent Advances in Microwave Theory and Applications, 2008. MICROWAVE 2008. International Conference on*, 924–927. 92

Ravindran, B. & Barto, A.G. (2003). An algebraic approach to abstraction in reinforcement learning. In *Twelfth Yale Workshop on Adaptive and Learning Systems*, 109–114. 86

Renault, O., Magnenat-Thalmann, N. & Thalmann, D. (1990). A vision-based approach to behavioral animation. *Visualization and Computer Animation*, **1**, 18–21. 6, 38

Reynolds, C.W. (1987). Flocks, herds and schools: A distributed behavioral model. *Computer Graphics*, **4**, 25–34. 2, 3, 36, 221

Reynolds, C.W. (1999). Steering behaviors for autonomous characters. In *Game Developers Conference*, 763–782, Miller Freeman Game Group, San Francisco, California. 37, 45

Rindsfüser, G. & Klügl, F. (2007). Agent-based pedestrian simulation: A case study of the Bern railway station. *disP*, **3**, 9–18. 3

Robin, T., Antonioni, G., Bierlaire, M. & Cruz, J. (2009). Specification, estimation and validation of a pedestrian walking behavior model. *Transportation Research*, **43**, 36–56. 3, 5, 110, 221

Rogsch, C. & Klingsch, W. (2012). Basics of software-tools for pedestrian movement, identification and results. *Fire Technology*, **48**, 105–125. 54

Ronald, N. & Sterling, L. (2005). A BDI approach to agent-based modelling of pedestrians. In *Proceedings of the 19th European Conference on Modelling and Simulation*, 169–174, European Council for Modelling and Simulation. 30

Rosenfeld, A. & Kraus, S. (2009). Modeling agents through bounded rationality theories. In *IJCAI 2009, Proceedings of the 21st International Joint Conference on Artificial Intelligence*, 264–271. 29

Rouphail, N., Humme, J., Milazzo, J. & Allen, P. (1998). Literature review for chapter 13, pedestrians, of the highway capacity manual. Literature research, Federal Highway Administration. USA. 118

Rummery, G.A. & Niranjan, M. (1994). On-line q-learning using connectionist systems. Tech. Rep. CUED/F-INFENG/TR 166, Engineering Departement, Cambridge University. 65

Rusmevichientong, P., Salisbury, J.A., Truss, L.T., van Roy, B. & Glynn, P.W. (2006). Opportunities and challenges in using online preference data for vehicle pricing: A case study at general motors. *Journal of Revenue and Pricing Management*, **5**, 45–61. 91

Sahaleh, A., Bierlaire, M., Farooq, B., Danalet, A. & Hänseler, F. (2012). Scenario analysis of pedestrian flow in public spaces. In *Proceedings of the 12th Swiss Transport Research Conference (STRC)*. 25

Sakuma, T., Mukai, T. & Kuriyama, S. (2005). Psychological model for animating crowded pedestrians: Virtual humans and social agents. *Comput. Animat. Virtual Worlds*, **16**, 343–351. 3, 45, 103

Sandholm, T. (2007). Perspectives on multi-agent learning. *Artificial Intelligence*, **171**, 382–391. 4

Santamaria, J.C., Sutton, R.S. & Ram, A. (1997). Experiments with reinforcement learning in problems with continuous state and action spaces. *Adaptive Behavior*, **6**, 163–217. 56, 79

Sarmady, S., Haron, F., Salahudin, M.M.M. & Talib, A.Z.H. (2007). Evaluation of existing software for simulating of crowd at masjid al-haram. *Jabatan Wakaf, Zakat dan Haji Malaysia*, **1**, 83–95. 52

Schadscheneider, A. & Seyfried, A. (2009). Validation of ca models of pedestrian dynamics with fundamental diagrams. *Cybernetics and Systems*, **40**, 367–389. 13

Schadschneider, A. & Seyfried, A. (2009a). *Pedestrian Behavior*, chap. Empirical results for Pedestrian Dynamics and Their Implications for Cellular Automata Models. Emerald. 23

Schadschneider, A. & Seyfried, A. (2009b). Validation of CA models of pedestrian dynamics with fundamental diagrams. *Cybernetics and Systems*, **5**, 367–389. 13, 131, 137

Schadschneider, A. & Syfried, A. (2011). Empirical results for pedestrian dynamics and their implications for modeling. *Networks and Heterogeneous Media*, **6**, 545–560. 3

Schadschneider, A., Klingsch, W., Kluepfel, H., Kretz, T., Rogsch, C. & Seyfried, A. (2008). *Evacuation dynamics: empirical results, modelling and applications. Encyclopedia of Complexity and Systems Science*, 3142–3176. Edited by R.A. Meyers. Springer. 12, 101, 172

Schelling, T.C. (1971). Dynamic models of segregation. *Journal of Mathematical Sociology*, **1**, 143–186. 30

Schreckenberg, M. & Wolf, D.E. (1998). *Traffic and granular flow'97*. Springer. 119

Seer, S., Brändle, N. & Bauer, D. (2010). *Pedestrian and Evacuation Dynamics 2008*, chap. Design of Decision Rules for Crowd Controlling Using Macroscopic Pedestrian Flow Simulation, 577–583. Springer. 36

Seitz, M. & Köster, G. (2012). Natural discretization of pedestrian movement in continuous space. *Physical Review E*, **86**, 046108. 23

Selfridge, O., Sutton, R. & Barto, A. (1985). Trainning and tracking in robotics. In *Ninth International Joint Conference on Artificial Intelligence*, 670–672. 86

Sen, S., Sen, I. & Sekaran, M. (1996). Multiagent coordination with learning classifier systems. 218–233, Springer Verlag. 72

Seneviratne, P. & Morrall, J. (1985). Analysis of factors affecting the choice route of pedestrians. *Transportation Planning and Technology*, **10**, 147–159. 18

Seyfried, A., Steffen, B., Klingsch, W. & Boltes, M. (2005). The fundamental diagram of pedestrian movement revisited. *Journal of Statistical Mechanics: Theory and Experiment*, P10002. 15, 224

Seyfried, A., Steffen, B., Klingsch, W., Lippert, T. & Boltes, M. (2007). Steps toward the fundamental diagram. empirical results and modelling. In *Pedestrian and Evacuation Dynamics 2005*, Part 3 , 377–390, Springer. 134

Shao, W. & Terzopoulos, D. (2005). Autonomous pedestrians. In *Proceedings of the 2005 ACM SIGGRAPH symposium on Computer animation*, 19–28, ACM Press, New York, NY, USA. 3, 37, 45, 216, 226

Shendarkar, A., Vasudevan, K., Lee, S. & Son, Y.J. (2008). Crowd simulation for emergency response using BDI agents based on immersive virtual reality. *Simulation and Modelling Practice and Theory*, **16**, 1415–1429. 31

Shiwakoti, N., Sarvi, M., Rose, G. & Burd, M. (2011). Animal dynamics based approach for modeling pedestrian crowd egress under panic conditions. *Transportation Research Part B*, **45**, 1433–1449. 35

Singh, S., Jaakkola, T., Littman, M. & Szepesvari, C. (2000). Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine Learning*, **39**, 287–308. 67

Singh, S., Litman, D., Kearns, M. & Walker, M. (2002). Optimizing dialogue management with reinforcement learning: Experiments with the njfun system. *Journal of Artificial Intelligence Research*, **16**, 105–133. 92

Singh, S., Barto, A. & Chentanez, N. (2004). Intrinsically motivated reinforcement learning. In *Advanced in Neural Information Processing Systems 17 (NIPS04)*, 13–18. 94

Singh, S., Lewis, R., Barto, A. & Sorg, J. (2010). Intrinsically motivated reinforcement learning: An evolutionary perspective. *Transactions on Autonomous Mental Development, IEEE*, **2**, 70–82. 94

Smith, R. (2001). ODE: Open Dynamics Engine. 113

Sobel, R.S. & Lillith, N. (1975). Determinant of nonstationary personal space invasion. *Journal of Social Psychology*, **97**, 39–45. 10

Steffen, B. & Seyfried, A. (2010). Methods for measuring pedestrian density, flow, speed and direction with minimal scatter. *Physica A*, **389**, 1902–1910. 132

Steiner, A., Philipp, M. & Schmid, A. (2007). Parameter stimation for pedestrian simulation model. In *7th Swiss Transport Research Conf.*, 1–29. 13, 126, 138, 164

Still, K. (2000). *Crowd Dynamics*. Ph.D. thesis, Department of Mathematics. Warwick University, UK. 11, 41, 164, 221

Stone, P. (1998). *Layered Learning in Multiagent Systems*. Ph.D. thesis, Computer Science Department. School of Computer Science, Carnegie Mellon University. 5

Stone, P. & Veloso, M. (2008). Multiagent systems: A survey from the machine learning perspective. *Autonomous Robots*, **8**, 345–383. 71

Stone, P., Sutton, R.S. & Kuhlmann, G. (2005). Reinforcement learning for robocup soccer keepaway. *Adaptive Behavior*, **13**, 165–188. 4, 73, 85, 92

Strehl, A.L. & Littman, M. (2005). A theoretical analysis of model-based interval estimation. In *22th International Conference on Machine Learning.*, 857–864. 70

Strehl, A.L., Li, L., Wiewiora, E., Langford, J. & Littman, M. (2006). Pac model-free reinforcement learning. In *23th International Conference on Machine Learning. Pittsburg*, 881–888. 70

Sung, M., Gleicher, M. & Chenney, S. (2004). Scalable behaviors for crowd simulations. In *SIGGRAPH'04 symposium on Computer animation*, 519–528, ACM Press. 5, 45, 48

Sutton, R.S. (1990). Integrated architecture for learning, planning and reacting based on approximating dynamic programming. In *Seventh International Conference on Machine Learning. Morgan Kaufmann*, 216–224. 62

Sutton, R.S. (1996). Generalization in reinforcement learning: Successful examples using sparse coarse coding. *Advances in Neural Information Processing Systems*, **8**, 1038–1044. 85

Sutton, R.S. & Barto, A.G. (1998). *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA. 57, 60, 66, 67, 68, 80, 81, 83, 85, 151, 220, 224

Szepesvári, C. (2010). *Algorithms for reinforcement learning*. Morgan Claypool. 57, 81

Tanner, B. & White, A. (2009). RL-Glue : Language-independent software for reinforcement-learning experiments. *Journal of Machine Learning Research*, **10**, 2133–2136. 96

Taylor, M.E. & Stone, P. (2007). Representation transfer for reinforcement learning. In *AAAI Fall Symposium on Computational Approaches to Representation Change during Learning and Development*. 88

Taylor, M.E. & Stone, P. (2009). Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, **10**, 1633–1685. 86, 90, 154, 224

Taylor, M.E., Stone, P. & Liu, Y. (2007). Transfer learning via inter-task mappings for temporal difference learning. *Journal of Machine Learning Research*, **8**, 1225–2167. 87, 153

Taylor, M.E., Suay, H.B. & Chernova, S. (2011). Using human demonstrations to improve reinforcement learning. In *The AAAI 2011 Spring Symposium — Help Me Help You: Bridging the Gaps in Human-Agent Collaboration*. 214

Teknomo, K. (2002). *Microscopic Pedestrian Flow Characteristics: Development of an Image Processing Data Collection and Simulation Model*. Ph.D. thesis, Department of Human Social Information Sciencies. Tohoku University, Japan. 3, 15, 126, 221

Teknomo, K., Takeyama, Y. & Inamura, H. (2003). Measuring microscopic flow performance for pedestrians. In *Selected papers of the 9th World Conference on Transport Research (WCTR). Elsevier*. 15

Templer, J.A. (1974). *Stair Shape and Human Movement*. Ph.D. thesis, Columbia University. 9

Tenorio-Gonzalez, A., Morales, E. & Villaseor-Pineda, L. (2010). Dynamic reward shaping: Training a robot by voice. In *Advances in Artificial Intelligence. IBERAMIA*, 483–492. 215

Terzopoulos, D. & Rabie, T.F. (1997). Animat vision: Active vision in artificial animals. *Videre: Journal of Computer Vision Research*, **1**, 1–19. 38

Tesauro, G. (1992). Practical issues in temporal difference learning. *Machine Learning*, **8**, 257–277. 73

Tesauro, G. (1995). Temporal difference learning and td-gammon. *Communications of the ACM*, **38**, 58–68. 92

Thalmann, D. & Musse, S.R. (2007). *Crowd Simulation*. Springer. London. 49

The MPI Forum (1993). MPI: A message passing interface. 114

Thorndike, E. & Woodworth, R. (1901). The influence of improvement in one mental function upon the efficiency of other functions. *Psychological Review*, **8**, 247–261. 85

Thrun, S. & Pratt, L. (1998). *Learning to learn*. Kluwer Academics Publishers. 85

Torrey, L. (2010). Crowd simulation via multi-agent reinforcement learning. In *Proceedings of the Sixth AAAI Conference On Artificial Intelligence and Interactive Digital Entertainment*, AAAI Press, Menlo Park, California. 5, 73, 96

Torrey, L., Walker, T., Shavlik, J.W. & Maclin, R. (2005). Using advice to transfer knowledge acquired in one reinforcement learning task to another. In *ECML 2005, 16th European Conference on Machine Learning, Porto, Portugal*, 412–424. 87

Train, K. (2003). *Discrete Choice Methods with Simulation*. Cambridge University Press. 17

Treuille, A., Cooper, S. & Popovic, Z. (2006). Continuum crowds. *ACM Trans. Graph.*, **25**. 5, 43, 184

Treuille, A., Lee, Y. & Popović, Z. (2007). Near-optimal character animation with continuous control. *ACM Transactions on Graphics (SIGGRAPH'07)*, **26**. 4, 92

Tsitsiklis, J.N. & Roy, B.V. (1997). An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, **42**, 674–690. 74

Van den Berg, J., Patil, S., Sewall, J., Manocha, D. & Lin, M. (2008). Interactive navigation of multiple agents in crowded environments. In *Proceedings of the 2008 Symposium on Interactive 3D Graphics and Games*, I3D '08, 139–147, ACM, New York, NY, USA. 47

Vandaele, N., van Woensel, T. & Verbruggen, A. (2000). A queueing based traffic flow model. *Transportation Research Part D: Transport and Environment*, **5**, 121–135. 21

Varas, A., Corneo, M., Mainemer, D., Toledo, B., Rogan, J., Muñoz, V. & Valdivia, J. (2007). Cellular automaton model for evacuation process with obstacles. *Physica A*, **382**, 631–642. 20

Vincent, R.A., Mitchell, A.I. & Robertson, D.I. (1980). User guide to TRANSYT version 8. Manual lr 888, Transport Research Laboratory. Crown. 14

Wagoum, A.U.K., Chraibi, M., Mehlich, J., Seyfried, A. & Schadschneider, A. (2010). Efficient and validated simulation of crowds for an evacuation assistant. *Computer Animation and Virtual Worlds*, **23**, 3–15. 39

Waldau, N. (2002). *Massenpanik in Gebäuden: Grundlagen und Simulationsmodelle, Planungskriterien zur Orientierung in Gebäuden bei steigender Stressbelastung*. Ph.D. thesis, Technische Universität Wien, Vienna, Austria. 34

Wampler, K., Andersen, E., Herbst, E., Lee, Y. & Popovic, Z. (2010). Character animation in two-player adversarial games. *ACM Trans. Graph.*, **29**, 26:1–26:13. 93

Ward, J. (2007). *Urban Movement: Models of Pedestrian Activity*. Ph.D. thesis, University College. London. UK. 30

Waters, K. & Terzopoulos, D. (1990). A physical model of facial tissue and muscle articulation. In *Proceedings of the First Conference on Visualization in Biomedical Computing. Pp 77–82*. 121

Watkins, C.J.C.H. (1989). *Learning from Delayed Rewards*. Ph.D. thesis, University of Cambridge, U.K. 63

Watkins, C.J.C.H. & Dayan, P. (1992). Q-learning. *Machine Learning*, **8**, 279–292. 63, 65

Wei-Guo, S., Yan-Fei, Y., Bing-Hong, W. & Wei-Cheng, F. (2006). Evacuation behaviors at exit in ca model with force essentials: a comparison with social force model. *Physica A*, **371**, 658–666. 20

Weidmann, U. (1993). Transporttechnik der fussgänger - transporttechnische eigenschaften des fussgängerverkehrs (literaturstudie). Literature Research 90, IVT an der ETH Zürich, ETH-Hönggerberg, CH-8093 Zürich. 12, 134, 137, 224

Weizenbaum, J. (1976). *Computer Power and Human Reason: From Judgement to Computation*. W.H. Freeman. San Francisco. 6

Weyns, D., Steegmans, E. & Holvoet, T. (2004). Towards active perception in situated multi-agent systems. *Applied Artificial Intelligence*, **18**, 867–883. 108

Whitehead, S.D. & Ballard, D.H. (1991). Learning to perceive and act by trial and error. *Machine Learning*, 45–83. 110

Whiteson, S., Taylor, M.E. & Stone, P. (2007). Adaptive tile coding for value function approximation. Tech. Rep. AI-TR-07-339, University of Texas at Austin. 85

Wiering, M. & van Otterlo, M. (2012). *Reinforcement Learning: State-of-the-Art*. Springer. 57

Wijn, P. (1980). *The aliniear viscoelastic properties of human skin in vivo for small deformation*. Thesis, University of Nijmegen, Netherlands. 121

Wooldridge, M. (2000). *Reasoning about Rational Agents*. The MIT Press. 29

Wooldridge, M. (2013). *Multi-Agent Systems 2nd Edition*, chap. Intelligent Agents, 3–50. MIT Press. 3, 29

Yan, Y. (2010). *Improved Social Force Model for Building Evacuation simulation*. Ph.D. thesis, University of Hong Kong. 40

Yang, L., Zhao, D., Li, J. & Fang, T. (2005). Simulation of the kin behavior in building occupant evacuation based on cellular automaton. *Build. Environ.*, **40**, 411–415. 20, 34

Yersin, B., Maïm, J., Ciechomski, P.D.H., Schertenleib, S. & Thalmann, D. (2005). Steering a virtual crowd based on a semantically augmented navigation graph. In *In Proceedings of the First International Workshop on Crowd Simulation (VCROWDS'05)*, 169–178. 43

Young, S. (2007). Evaluation of pedestrian walking speeds in airport terminals. *Transportation Research Record: Journal of the Transportation Research Board*, **1674**, 20–26. 18

Yu, W., Chen, L., Dong, R. & Dai, S. (2005). Centrifugal model for pedestrian dynamics. *Physical Review E*, **72**, 026112. 27

Zainuddin, Z. & Shuaib, M. (2010). Incorporating decision making capability into the social forces model in unidirectional flow. *Research Journal of Applied Science*, **5**, 388–393. 27

Zeltzer, D. (1991). Making them move. chap. Tasklevel Graphical Simulation: Abstraction, Representation, and Control, 3–33, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA. 35

Zhou, S., Chen, D., Cai, W., Luo, L., Low, M.Y.H. & Tian, F. (2010). Crowd modeling and simulation technologies. *ACM Transactions on Modeling and ComputerSimulation*, **20**, A20–1–35. 32, 33, 46, 52, 104

Zhu, W. & Timmermans, H. (2007). *Pedestrian and Evacuation Dynamics 2005*, chap. Exploring Pedestrian Shopping Decision Processes. An application of Gene Expression Programming, 145–153. Springer. 4, 19

Zipf, G.K. (1949). *Human Behavior and the Principle of Least Effort: An Introduction to Human Ecology*. Addison-Wesley. 31

Explicit