Trajectory Analysis using Point Distribution Models: Algorithms, Performance Evaluation, and Experimental Validation using Mobile Robots

THÈSE N^{O} 4262 (2009)

PRÉSENTÉE LE 9 JANVIER 2009

À LA FACULTE SCIENCES ET TECHNIQUES DE L'INGÉNIEUR

LABORATOIRE DE PRODUCTION MICROTECHNIQUE 1

Laboratoire de systèmes et algorithmes intelligents distribués

PROGRAMME DOCTORAL EN SYSTÈMES DE PRODUCTION ET ROBOTIQUE

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Pierre RODUIT

acceptée sur proposition du jury:

Prof. H. Bleuler, président du jury
Prof. J. Jacot, Prof. A. Martinoli, directeurs de thèse
Prof. M. Bierlaire, rapporteur
Dr Y. Lopez de Meneses, rapporteur
Prof. U. Nehmzow, rapporteur

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Suisse
2009

Remerciements

A mes deux professeurs, Jacques Jacot et Alcherio Martinoli, pour m'avoir permis de faire cette thèse. Sans leur expérience, leurs conseils et le temps qu'ils y ont investi, cette thèse n'aurait jamais pu être ce qu'elle est devenue.
Au docteur Yuri Lopez-de-Meneses qui m'a donné le goût de la statistique et de la recherche scientifique. Sans lui, il n'y aurait pas eu de projet et encore moins de thèse.
Aux membres du jury et du comité de parrainage : les professeurs Michel Bierlaire, Hannes Bleuler, Ulrich Nehmzow, Jean-Philippe Thiran et le docteur Yuri Lopez-de-Meneses.
Au *Fond National Suisse de la Recherche Scientifique*, pour avoir subventionné ces quatre années de recherche.
Au docteur Alain Dufaux et à Giuseppe Zamuner pour avoir lu ce monolithe et m'avoir permis d'en améliorer le contenu.
A Christopher Cianci et Friederike Ilschner pour leurs corrections linguistiques, sans qui la lisibilité de ce document serait bien moins agréable.
A Corinne Farquharson, Karine Genoud et Erika Raetz qui ont oeuvré et oeuvrent tant pour le LPM et le DISAL. Faire partie de deux laboratoires en même temps durant quatre ans n'a pas toujours été facile; elles ont su cependant simplifier les démarches administratives et m'ont permis de rester en lien avec les deux laboratoires.
Aux compagnons de mes multiples bureaux: Mario Bellino, Christopher Cianci, Nikolaus Correll, Corinne Farquharson, Sandra Koelemeijer, Amanda Prorok, Surya Shravan Kumar Sajja et Giuseppe Zamuner. J'ai beaucoup apprécié les discussions que nous avons eues, liées ou non à cette thèse.
A tous mes collègues du *Laboratoire de Production Microtechnique* et du *Laboratoire de Systèmes et Algorithmes Intelligents Distribués*.
A Lydiane, pour m'avoir supporté et soutenu durant ces années.
A mes parents, ma famille et mes amis.

Abstract

This thesis focuses on the analysis of the trajectories of a mobile agent. It presents different techniques to acquire a quantitative measure of the difference between two trajectories or two trajectory datasets. A novel approach is presented here, based on the Point Distribution Model (PDM). This model was developed by computer vision scientists to compare deformable shapes. This thesis presents the mathematical reformulation of the PDM to fit spatiotemporal data, such as trajectory information. The behavior of a mobile agent can rarely be represented by a unique trajectory, as its stochastic component will not be taken into account. Thus, the PDM focuses on the comparison of trajectory datasets. If the difference between datasets is greater than the variation within each dataset, it will be observable in the first few dimensions of the PDM. Moreover, this difference can also be quantified using the inter-cluster distance defined in this thesis. The resulting measure is much more efficient than visual comparisons of trajectories, as are often made in existing scientific literature.

This thesis also compares the PDM with standard techniques, such as statistical tests, Hidden Markov Models (HMMs) or Correlated Random Walk (CRW) models. As a PDM is a linear transformation of space, it is much simpler to comprehend. Moreover, spatial representations of the deformation modes can easily be constructed in order to make the model more intuitive. This thesis also presents the limits of the PDM and offers other solutions when it is not adequate. From the different results obtained, it can be pointed out that no universal solution exists for the analysis of trajectories, however, solutions were found and described for all of the problems presented in this thesis.

As the PDM requires that all the trajectories consist of the same number of points, techniques of resampling were studied. The main solution was developed for trajectories generated on a track, such as the trajectory of a car on a road or the trajectory of a pedestrian in a hallway. The different resampling techniques presented in this thesis provide solutions to all the experimental setups studied, and can easily be modified to fit other scenarios. It is however very important to understand how they work and to tune their parameters according to the characteristics of the experimental setup.

The main principle of this thesis is that analysis techniques and data representations must be appropriately selected with respect to the fundamental goal. Even a simple tool such as the t-test can occasionally be sufficient to measure trajectory differences. However, if no dissimilarity can be observed, it does not necessarily mean that the trajectories are equal—it merely indicates that the analyzed feature is similar. Alternatively, other more complex methods could be used to highlight differences. Ultimately, two trajectories are equal if and only if they consist of the exact same sequence of points. Otherwise, a difference can always be found. Thus, it is important to know which trajectory features have to be compared.

Finally, the diverse techniques used in this thesis offer a complete methodology to analyze trajectories.

Keywords: trajectory analysis, mobile robots, behavioral identification, quantitative measure of trajectory differences, motion analysis

Résumé

Cette thèse traite de l'analyse de trajectoires afin de comparer le comportement d'un agent mobile. Elle présente diverses techniques qui permettent d'acquérir une mesure quantitative de la différence entre deux trajectoires ou deux groupes de trajectoires. La méthode principalement décrite est le *Point Distribution Model* (PDM), qui met en avant les déformations de variance maximale des trajectoires. Ainsi, si la différence entre des groupes de trajectoires est plus importante que la variation interne de ces groupes due à la variabilité des comportements, elle pourra être observée dans les premières dimensions du PDM. De plus, cette différence peut être quantifiée en utilisant la distance inter-cluster qui est définie dans cette thèse. Cette solution est bien plus efficace que les comparaisons visuelles faites habituellement dans la littérature scientifique.

Cette thèse inclut aussi une comparaison de la méthode PDM avec d'autres techniques telles que les différents tests statistiques, les modèles de Markov cachés (*Hidden Markov Model* : HMM) ou la marche aléatoire corrélée (*Correlated Random Walk* : CRW). Bien que le PDM soit plus transparent et plus simple à appréhender, il possède aussi des limites décrites dans cette thèse. Il en ressort qu'aucune solution universelle existe pour la comparaison de trajectoires. Cependant, les performances des différentes techniques sont démontrées par les résultats obtenus en simulations et lors d'expériences réelles.

Vu que le PDM est uniquement applicable à des trajectoires possédant le même nombre de points, il est important de posséder des techniques pour ré-échantillonner les données. Une méthode a particulièrement été développée pour des trajectoires générées sur une piste. Cette méthode est très générale et peut être appliquée, par exemple, à l'analyse des trajectoires d'une voiture sur une route, au déplacement d'un piéton dans un corridor ou au mouvement d'un robot mobile sur un circuit. Il reste cependant très important de maîtriser les paramètres de ré-échantillonnage et surtout de les régler en fonction de l'analyse.

Le message principal de cette thèse est qu'il faut choisir la méthode d'analyse et surtout les données sur lesquelles se fait cette analyse en fonction des buts recherchés. Par exemple, un simple *t-test* peut être suffisant pour acquérir la mesure désirée de la différence entre deux trajectoires. Cependant, si ce test ne parvient pas à les différencier, cela ne veut pas dire qu'elles soient égales. Cela signifie juste que la caractéristique comparée est similaire. Ultimement, deux trajectoires sont égales si et seulement si elles possèdent exactement la même séquence de points. De ce fait, choisir la bonne caractéristique et la bonne méthode est primordial. Par sa description très complète des techniques utilisées, cette thèse présente une méthodologie complète pour comparer des trajectoires.

Mots clés: analyse de trajectoires, robots mobiles, comparaison quantitative de comportements

Contents

Chapter 1

Introduction

How can the difference between two trajectories or two trajectory datasets be measured? The difference between two points can be easily evaluated with the Euclidean distance. Likewise, two circles can be compared based on their centers and radii. However, no common method to quantitatively evaluate trajectory differences stands out in the scientific literature. Most of the time, the comparison is qualitatively assessed from cursory visual inspection of the trajectories. However, a quantitative measure could be very helpful to evaluate performances as well as understand subtle mechanisms related to the interplay of a given mobile system with its environment. Potential targeted systems include engineered mobile systems such as mobile robots and natural systems such as athletes and animals. It could also be used to classify or identify behaviors of mobile agents.

This thesis presents the Point Distribution Model as a scientific tool to acquire a quantitative measure of trajectory differences. The demonstration of its performance is assessed with mobile robot trajectories. A thorough study of its advantages and drawbacks is presented, together with an analysis of the influence of model parameters. Its performance will be also be compared with standard methods, such as the Correlated Random Walk (CRW), the Hidden Markov Model (HMM), or statistical tests.

1.1 Problem description

In recent years, decreasing cost and increased performance of electronic tracking systems (video, radar, laser, GPS, etc.) have prompted their wide use in consumer, industrial and scientific applications. A variety of uses have been proposed for these systems, tracking anything from pedestrians and cars to insects and other animal societies. Therefore, a large amount of trajectory data is becoming available.

On the other hand, behavior or performance analysis based on trajectories is not a well explored field. As we will mention in more detail below, trajectory clustering, behavioral analysis, and trajectory modeling have been proposed, but a good methodology to extract behavioral features from trajectories is still lacking. Thus, this thesis presents the Point Distribution Model as a tool for analyzing trajectories, and its performance is compared with alternative methods. As trajectories are influenced by the environment as well as the agent's behavior and morphology, different case studies have been considered to evaluate this influence on the resulting models as well as the methodology. We chose a mobile robotic platform

as a trajectory generator for its flexibility and repeatability. Using mobile robots allows us to collect hundreds of trajectories in a short time and their simulation with software tools makes this time even shorter. Moreover, unlike living beings, it is quite easy to modify their behavior and morphology. Finally, mobile robots can be seen as a fairly faithful emulator of living creatures. They can be engineered to reproduce natural behaviors in the same physical environment and they allow for a clear decoupling of the behavior (software) from the morphology (hardware).

The possible applications of this methodology include performance analysis for robotics and athletics as well as behavioral identification. The performance of an athlete, such as a skier or a tennis player, can be improved by analyzing the way he moves and by comparing his trajectory with trajectories of other athletes. The interest for robotics is not so straightforward. A robot can in principle be fully engineered, and therefore its trajectories can be reasonably well controlled, however, hard volume, mass or cost constraints imposed by a specific application might have a major impact on possible design choices at both software and hardware level and , as a consequence, on the controllability of trajectories produced by such platforms. In such framework, this methodology could have a major impact. Being able to model and analyze trajectories could allow for further system optimization. For instance, trajectory analysis could point out that a given, less expensive combination of hardware and software achieves comparable results to those produced by a more sophisticated solution. Finally, behavior identification has a major interest for biologists in the context of learning and modeling animal behavior. Dangerous behavioral identification has also a great interest for surveillance and for reducing car accidents. While this thesis validates theoretical results and algorithms mainly with the help of mobile robots, its methodological framework impacts all the areas mentioned above requiring some sort of trajectory analysis.

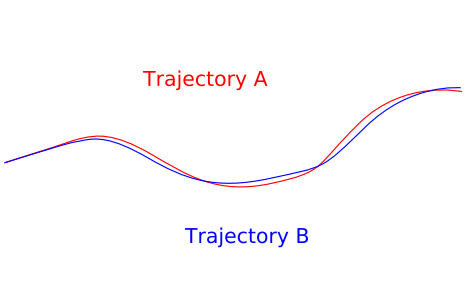
1.2 Trajectory analysis

Trajectory analysis is a major field of research and the state of the art presented in Chapter 2 will give an idea of how many domains are concerned by this topic. Current research is mainly in trajectory classification, motion modeling, or gesture analysis; while this thesis is related to those applications, it is focusing on the quantitative measure of trajectory differences. Even though certain techniques used in other fields can be recycled to address this problem, it remains largely unexplored.

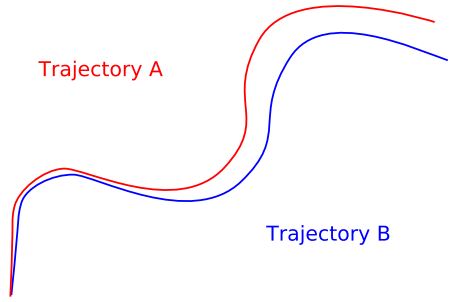
The work of Waegli and Skaloud is one of the rare examples falling into this specific field. They presented the analysis of skiers' trajectories [1, 2]. However, as their work focuses mainly on the extraction of such trajectories, their comparison methods were not highly developed.

1.3 Definition of the different trajectory types

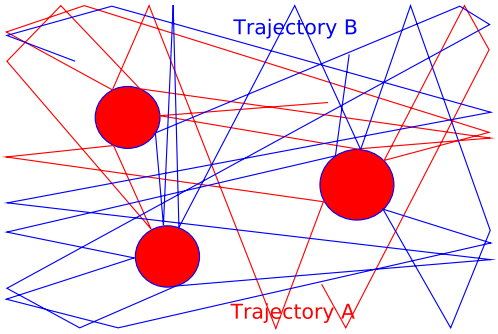
Trajectories are continuous functions from R to R^n, where n is the number of spatial dimensions. In two dimensions, this function, f, becomes: $[x, y] = f(t)$, where x and y are the spatial coordinates and t corresponds to time. Many different kinds of trajectories exist in real life. As it is not possible to propose a single optimal solution to treat all of these trajectories, this thesis focuses on three selected



(a) Trajectories with common frame and without drift



(b) Trajectories with common frame and with drift



(c) Trajectories without common frame

Figure 1.1: Examples of trajectories for all the cases that will be treated in this thesis

types of trajectories. Even if it reduces the scope of our research, it remains broad enough to include all the applications proposed previously.

- Trajectories with common frame and without drift (Figure 1.1(a))
  The agent follows a specific path and there are reference points on its path allowing it to recalibrate its trajectory (*e.g.*, driving on a road, a skier during a competition, a robot endowed with an absolute positioning system such as a GPS). In this case, the trajectories can be compared directly, as the spatial and temporal distances between comparable trajectory points are a measure of the trajectory differences.

- Trajectories with common frame and with drift (Figure 1.1(b))
  The moving agent has no absolute reference anchored to the environment. In this case, a drift can appear in the agent movement and thus the distances between the trajectory points are not the best measure of trajectory differences, as the major source of them is the drift, rather than the small variations due to differences in the agent's behavior. Analyzing the trajectories in small pieces, recalibrating the spatial and temporal reference between each piece represents a better way to process the data. An example scenario in which such type of trajectories can be generated is represented by a human trying to move straight on a flat surface with his eyes closed and without other input to serve as an absolute spatial reference. A further scenario could be a robot attempting to move to a specific place using only on-board odometric sensors and without having any specific guidance from cues perceived in the environment (*e.g.*, a wall which could be followed as in the first type of trajectories).

- Trajectories without common frame (Figure 1.1(c))
  The last case involves the study of an agent without a planned path and reacting to the environment. Examples might include a robot avoiding obstacles in a closed arena, the movement of a billiard ball, or a child during a game of blind man's bluff.

This nomenclature will be used throughout the remainder of the thesis.

1.4 Original contribution of the thesis

The quantitative comparison of trajectories has not been well developed as a field. The original contributions of this thesis to the domain are:

- the adaptation of the PDM to spatiotemporal data, such as trajectories,
- the definition of the inter-cluster distance as a quantitative measure of trajectory dataset differences,
- the development of general space-based sampling techniques,
- the modification of the CRW model to take into account more than one step of history,

- the extensive use of mobile robotic platforms (both simulated and real) as tool for validating trajectory analysis methods,
- and the application of trajectory analysis to problems in mobile robotics, including verification of simulation faithfulness to reality.

These different tools and methods allow for a quantitative comparison of trajectory differences; specifically, they prove useful for the comparison of motion behaviors and mobile systems as a whole, based on their trajectories.

1.5 Thesis organization

After this introduction, the core of this thesis will be presented as follows.

Chapter 2 presents a broad overview of the various sub-domains and current work in trajectory analysis. Next, Chapter 3 introduces the Point Distribution Model (PDM), and also shows its application to synthetic trajectories, allowing a better understanding of the way PDMs work.

As PDMs requires the same number of points per trajectory, Chapter 4 presents methods to transform trajectory datasets and make them compatible with PDMs. These methods are close to the way a human would do it naturally and show that simple solutions can be found for nearly all case studies and that sampling has a major influence on the PDM analysis results. As other techniques require also the same number of points per trajectory, these solutions are not limited to Point Distribution Models.

Chapter 5 will then present and compare different methods to analyze the three kinds of trajectories introduced previously. Methods presented include statistical tests, PDMs, HMMs, and CRW models.

The following chapters will then show the application of the proposed sampling and analysis methods to different case studies. In Chapter 6, the PDM method will be applied to the trajectories of a simulated mobile robot. This chapter shows the interests and limitations of the method and demonstrates the different factors influencing the PDM and its performance. Chapter 7 will then present more concrete case studies of PDM analyses of real agent trajectories. Finally, Chapter 8 will introduce two alternative methods, the GMM/HMM hybrid model and the CRW model. Their advantages and drawbacks will be clearly described in comparison to the PDM.

Appendix A is an important add-on, as it will introduce all the mathematical notions used throughout this thesis. It thereby provides the mathematical background necessary to understand this work. Finally, Appendix B will introduce the different acronyms used throughout this thesis.

Chapter 2

State of the art

This chapter will present a broad overview of the various sub-domains and current work in trajectory analysis. The underlying goal is to show the variety of applications relying on trajectory analysis methods, and also to list the different techniques used in this research field. Thus, the first section will present several domains of application, such as gesture analysis, surveillance systems, or behavioral identification. The second section will then present the different techniques in use, and will thoroughly describe their advantages and drawbacks.

2.1 Research area overview

Computer vision researchers provide during the last years nice tools to acquire the trajectories of a moving agent, using vision based tracking systems. Sanfeliu *et al.* presented a good overview of how motion analysis can be accomplished [3]. Nearly all the fields of research, which will be subsequently described, use vision-based tracking systems to extract the agent trajectories.

Another main contribution of the computer vision field was the developpement of the Point Distribution Model by Cootes *et al.*[4], a deformable template which will be extensively used in this thesis. The deformable templates are still widely used for vision tracking systems [5].

This section will first present works that are slightly linked to this thesis, and finish with the works closely related to it. The different references are regrouped into research areas.

2.1.1 Physical models

Physics developed perhaps the first fields of research about trajectory analysis. Old topics such as the equations of movement, ballistic or astronomy focus heavily on the analysis and the prediction of movement. Today there are still a lot of work realized in these fields, going from the optimization of trajectories to the moon [6] to the control of a fast-moving sailcraft trajectory near a sun [7].

Physical models are also used in climate or meteorologic applications [8, 9, 10, 11] in order to predict fire plume or cloud movements, pollution dispersion or the localization of clean air zones. Another application is marine surveillance and research. For instance, Meier [12] presented a methodology to improve ice pack trajectory modeling and Marghany [13] modeled oil spill trajectories.

The main goal of these works is to develop displacement models based on fluid dynamics or other physical laws. Therefore, the accomplished trajectory analysis is a comparison between the models and real life data, in order to evaluate the model quality rather than capturing the trajectories at highest levels of abstraction. Moreover, these multi-particle models are not able to represent trajectories of intelligent agents which perceive, compute, act, and decide in the real world.

Physical laws are also widely used in robotics to compute trajectories of robotic arms. These applications are mainly related to industrial production. For example, Munasinghe *et al.* propose a new control algorithm for industrial robot arms based on off-line trajectory generation with compensations to statics and dynamics [14].

2.1.2 Gesture analysis

The analysis of gestures is a major field in the analysis of trajectories. It focuses on the analysis of the relative displacements of numerous body parts, when performing specific actions or gestures. Fanti *et al.* developed probabilistic models to analyze joint movements [15], when Shin *et al.* used Beziez curves [16] and Huang *et al.* used Hidden Markov Models (HMM) and Principal Distribution Models to recognize gestures [17]. Multiple models where developed to focus on specific aspects of human motion, such as drawing [18] and gait [19]. Wu *et al.* searched for motion invariants and used Dynamic Time Warping (DTW) analysis of the curvature and torsion profiles and their respective derivatives, to measure gesture similarities [20]. In this way, they were able to classify gestures corresponding to drawn digits and letters. Urtasun, Fleet and Fua proposed an approach to 3D people tracking with learned motion models [21].

2.1.3 Traffic analysis

Another field of research is traffic analysis and collision prediction. The goal is to learn how to detect dangerous driving in order to ultimately reduce to number of road accidents.

Lauffenburger *et al.* extracted trajectory data of a car via multiple sensors such as a Differential Global Positioning System (DGPS), an Inertial Measurement Unit (IMU),... They made also simple trajectory comparisons between a novice and a expert driver [22].

Hu *et al.* worked on car trajectories. In order to simulate accidents, they carried out their experiments in an indoor model of a real traffic scene. They used an Artificial Neural Network (ANN) to model the trajectories and to predict the collisions [23]. They also used a Hierarchical Self-Organizing Neural Network to model the trajectories [24] and made a computation comparison with the Vector Quantization method used by Johnson and Hogg [25].

Oliver *et al.* worked on the modeling of driver behaviors with HMM or Coupled Hidden Markov Models (CHMM) [26]. They tried to recognize behaviors such as turning, stopping, starting, changing lane,...

2.1.4 Video surveillance

Surveillance and more generally people tracking is a major field of research. The security need of the last years leads to an increase of the resources involved (researchers, money, etc.). Vision systems are widely used and their goal is to recognize strange, dangerous, or illegal behaviors in order to improve a human surveillance work by focusing his attentiveness on the abnormal cases. While all the works presented below share such common goal, they present different methods to achieve it.

Owens, Hunter *et al.* have been working on automated vision systems to help Closed Circuit Television (CCTV) surveillance operators [27, 28, 29]. They used a hybrid system based on background subtraction in order to collect pedestrian trajectories. A Self-Organizing Map (SOM, also called Kohonen network) is then used on the data to search differences in the pedestrian trajectories compared to a large (>300) training set of trajectories. This method analyzes the trajectories piecewise (small segments) in order to classify the segments that are too different from the ones of the learning batch.

Grimson *et al.* used an adaptive tracking system to extract track data from the moving objects [30]. Trajectories are encoded as flow vectors (x,y,dx,dy). Object size and aspect ratio are also used for object recognition. Data clustering is done via an entropy minimization algorithm proposed by Wallace [31] or by multiple Gaussian modeling with a K-Means algorithm for the clustering.

Lee *et al.* modeled human trajectories for video surveillance and made a comparison between two models: a Hidden Markov Model (HMM) and a Least Common Subsequence (LCSS) model implemented on pedestrian trajectories in a city street [32].

In parallel to their work on traffic analysis, Oliver *et al.* modeled also human interactions with HMM or CHMM [33].

Collins *et al.* present in [34] the Video Surveillance and Monitoring (VSAM) project, realized in the Robotics Institute at Carnegie Mellon University. VSAM was a large project of monitoring pedestrians and cars on the campus using multiple cameras. It introduced a lot of different techniques about tracking, object recognition, and gait classification.

Cuntoor *et al.* used HMM to analyze and classify pedestrian trajectories at an airport [35]. Vaswani *et al.* worked on the same dataset and developed a technique based on the distribution of landmarks to detect anomalies [36].

Evans *et al.* built a surveillance system on the analysis of visual motion features to detect events and threats [37]

2.1.5 Activity labeling

An activity very close to video surveillance is activity labeling. In this case, the goal is not to detect abnormal or dangerous activities, but to be able to label all the different activities accomplished by the tracked agents (mainly pedestrians).

As an example, Remagnino *et al.* worked on the automatic annotation of pedestrian behavior in a car park [38, 39]. They use spatial data such as curvature, speed, location, heading, etc. to classify the pedestrian behavior using Bayesian Networks [40].

Bertini *et al.*, in their case, used Finite State Machines to model sport actions [41]. Each change

of state is triggered by player actions, such as "player enter a specific location" or "player shoot the ball". Their goal is to annotate highlights of different sports such as soccer, basketball or swimming. Different data are taken into account, such as the velocity of the ball, the player's speed and position on the field,... It is worth noting that Bertini *et al.* are actually doing behavioral modeling rather than trajectory analysis. However, we do believe that this is an interesting preliminary attempt toward a real trajectory analysis applied to sport performances.

2.1.6 Trajectory clustering

Like video surveillance, trajectory clustering is one of the major field of research about trajectory analysis. The difference with video surveillance and activity labeling is not so evident, as the tools and techniques used are very similar. The goal is however slightly different, as it intends to separate a dataset into clusters of similar trajectories.

Johnson and Hogg modeled the trajectories as a sequence of flow vectors. The probability distribution function of the flow vectors is approximated with Learning Vector Quantization (LVQ) [25]. They also model the probability distribution function by a Gaussian Mixture Model (GMM) [42].

Inspired by the first work of Johnson and Hogg, Stauffer *et al.* [43, 44] produced a robust tracker and used also Vector Quantization to cluster the trajectories.

Porikli [45, 46] proposed another way to learn trajectory patterns, using spectral clustering. He first modeled the trajectories with a HMM. A different HMM is learned for each trajectory. An affinity matrix between each trajectory and each HMM is then computed. Each component of the affinity matrix a_{ij} corresponds to the likelihood of the trajectory i with the HMM learned on trajectory j. Then, an eigenvector decomposition of the matrix is done in order to achieve a clustering of the trajectories. This method supports different numbers of points per trajectories. In [47], he proposed also a method using only the HMM parameters to cluster the trajectories, without using an affinity matrix.

Similarly, Alon *et al.* worked on pedestrian motion clustering using a HMM [48]. People from the same laboratory, Buzan *et al.* used Least Common Subsequence (LCSS) to cluster car or pedestrian trajectories, in order to track multiple objects with occlusion [49]. Piciarelli *et al.* used also LCSS to cluster human trajectories in a car park [50]. The clustering algorithm has a dynamic number of clusters rather than a pre-established one: if a trajectory does not fit into the boundaries of all the existing clusters, a new cluster is created for this trajectory.

Corina Sas worked on the analysis of people behaviors in virtually augmented environments [51, 52, 53, 54]. People were doing exploratory tasks in a virtually augmented building and thereafter needed to move as fast as possible to a given area. The resulting trajectories were clustered in two different ways. The first analysis was done with a statistical comparison of data such as the number of rotations, the number of consecutive rotations, the number of consecutive translations,... In this analysis, she compared regular and novice users of the virtually augmented environment, outlining differences with central tendency statistics. The second analysis was done with a Self-Organizing Map (SOM) implemented on the subdivision of the building in 28 areas.

Ng and Gong used Dynamic Time Warp distance to evaluate the similarities between trajectories

[55]. They worked on hand gestures and pedestrian movements. Likewise, Buchin *et al.* clustered subtrajectories acquired with a GPS on 50x50 km area, relying on the Fréchet distance and its discrete version [56]. Trajcevski *et al.* used also the Fréchet distance to compare trajectories, and mixed it with the Hausdorff distance [57]. They validated their new measure with a trajectory database acquired from objects moving on a road. Bashir, in his PhD thesis, used Principal Component Analysis (PCA) and HMM to cluster pedestrian trajectories and hand gestures [58]. He made also a fair introduction of the research field.

Finally, Efrat and Fan used the continuous DTW to compare handwritten signatures [59]. Jelle Van Hoof, in his master thesis [60], focused on curve comparisons. He used the Fréchet distance and the Hausdorff distance to evaluate the curve differences. These last two works are more related to curve fitting.

2.1.7 Semantic region detection

This research field develop techniques to find the different paths of motion in a scene, called the semantic regions. For example, on a road, the main motion paths will be the two lanes. It is different from the trajectory clustering, as the goal is not to have clusters of trajectories, but to define regions of different motion, even if those regions are closely related and often built from the trajectory clusters.

Fernyhough *et al.* developed a method to auto-create semantic regions for road and pedestrian applications [61]. They first built path models from the courses taken by the moving object and store them in a path database. Based on this path database and on the spatial occupation of the different paths, they extracted the different regions of interest.

Makris and Ellis used Gaussian Models to describe the scene entry points [62]. From all the acquired trajectories, they observed their entry points and regrouped them in the Gaussian models. McKenna and Charif used GMM to model the regions of inactivity and the trajectory entry points [63].

Wang *et al.* developed a method to automatically discover the principal paths of motion, based on the spatial distribution of the trajectory points and on the velocity of the moving agent [64]. They modeled also the entry and exit points of the different paths.

2.1.8 Animal and mixed animal-robot societies

This field of research tries to developed models of animal behavior. The models developed are not built from scene elements. They are based on Correlated Random Walk or similar probabilistic models. They do not intend to capture the different motion paths, but rather to get a general model of the wandering behavior.

Egerstedt *et al.* worked on the tracking of ants [65]. They used *Motion Description Language* to model ant movement. This modeling tool automatically infers aspects of ant behavior and interaction between the ants. Gordon worked also on the behavior of ants [66]. She based her analysis on the ant density in the different places of the experimental arena. Kohler and Wehner worked also with ants and made visual comparison of their motion paths [67].

Balch and Khan *et al.* developed also models of individual and group behavior using tracking data [68, 69, 70].

John A. Byers described animal movement using Correlated Random Walk dependent on three parameters: number of steps, step size, and distribution of random turning angles [71]. Jost *et al.* also used Correlated Random Walk to model cockroach behavior. Then, they implemented the parameters of this model into a micro-robot and compared the resulting behavior with the one of the cockroaches [72]. A more thorough description of motion behaviors was made by Garner *et al.* in [73]. Jeanson *et al.* worked also on the behavioral analysis of cockroaches [74].

Biro *et al.* worked on the trajectories of pigeons [75]. They used the Fasano-Franceschini extension in two dimensions of the the Kolmogorov-Smirnov test to compare the trajectory point distributions.

The last work does not focus on animal, but rather on human behavior. However, the intentions of this work is much more close to the work of the biologists, than the work done in video surveillance. Antonini *et al.* worked on a probabilistic model of a pedestrian behavior [76].

2.1.9 Robotics

A major field of research in robotics is to create robots acting like human. For example, Kajikawa *et al.* worked on the motion planning of a receiver humanoid-robot during hand-over operations. They modeled human hand trajectories and reproduced them with a humanoid robot. Finally they compared the human and robot trajectories [77, 78].

Another field of research is the behavior analysis of autonomous mobile robots. Nehmzow [79], Smithers [80] and Schöner *et al.* [81] showed the lack of quantitative performance measurement of robot-environment interaction. While they presented methods to model the behavior in a quantitative way using a dynamical system approach, no specific trajectory analysis framework was developed by these researchers. Nehmzow wrote a book describing how to use dynamical system approaches to analyze robot behavior [82]. Iglesias, Akanyeti, Nehmzow *et al.* proposed also the use of the NARMAX (Non-Linear Auto-Regressive Moving Average with Exogeneous Input) system identification technique to analyze robot behavior [83, 84].

Goldberg and Matarić used an Augmented Markov Model (AMM) to capture the interaction between a robot and its environment, which could be static (single robot) or dynamic (multiple robots sharing the same environment) [85, 86]. The AMM was built on the sensory inputs of each individual robot, and did not take into account any positioning or temporal information. Yan and Matarić further worked on the interaction dynamics between two or more individuals [87].

2.2 Description of existing models

As a lot of work was accomplished in the field of trajectory analysis, it is important to show the advantages and drawbacks of the previously introduced methods.

One kind of models are deterministic kinetic models. They can be used with passive objects (ballistic) or with completely controlled agents [6, 7]. As this thesis does not aim to predict trajectories, but instead

to analyze the difference between trajectory sets, exact models are not useful at all. Similarly, diffusion models for liquids or gases [8, 9, 10, 11, 12, 13] are also out of the scope of this thesis, as they focus on unintelligent moving agents.

Some researchers [15, 19, 76] have also built their own probabilistic models to analyze trajectories. However, they are closely bound to the targeted application and thus are not reusable in our case.

Artificial Neural Networks (ANN) are widely used in this field to classify trajectory sets [23, 24]. They can be used in a lot of declinations: single or multi-layers, different architectures, etc. Even if they can be quite efficient for classification purpose, they remain a black box and thus do not allow to easily link the classification results to the trajectory differences. Moreover, they are also highly non-linear. One type of neural network, the Self Organizing Map (SOM) or Kohonen network, is sufficiently interesting to be presented here a bit more thoroughly. This model, even if it remains a black box, produces a map from the trajectory dataset showing the different trajectory families. It is widely used [27, 28, 29, 54, 51] and is quite efficient to represent highly different trajectories. The link between the position on the resulting map and the trajectory differences remains difficult to apprehend. The last drawback of the ANN is that all the input vectors must have the same length. Thus trajectories with different number of points must be first resampled.

Closely related to the SOM, Vector Quantization (VQ or LVQ for Learning Vector Quantization) are also used for trajectory analysis [25, 43, 44, 54, 51]. The trajectory dataset is represented by a set of quantization vectors. All the trajectories are then represented by the nearest quantization vector. This method widely used for compression is quite efficient, but suffers from drawbacks similar to those of the ANN: if the trajectories have not the same number of points, they must be sampled. Moreover, trajectory far from a quantization vector suffer from a high approximation error. Thus, as a lot of data is lost in the quantization process, the link between the results and small trajectory differences is difficult to maintain.

Statistical tests are used to evaluate trajectory differences, such as the Fasano-Franceschini test in [75]. However, as the trajectories are considered as unordered sets of points, the temporal aspect of the trajectories is lost. More results and considerations with statistical tests will be presented in Chapter 5.

Bayesian Networks [38, 39], Finite State Machines [41] or Markov Models [34] are also used with trajectories. However, they are reduced to activity labeling, as they represent the transition between the different behavioral states of the moving agent, and not the trajectory itself.

Non-Linear Auto-Regressive Moving Average with Exogenous Input models (NARMAX) are used in [83, 84] to analyze robot behavior. However, they do not model the trajectories, but instead map the motor commands to the sensory inputs. Thus, they are great tools to identify the principal parameters of a controller, but are not suited to measure trajectory differences. Augmented Markov Models presented in [85, 86] serve also the same purpose.

Least Common Subsequence (LCSS or LCS) are used in [32, 49, 50]. They are a common computer science problem and are used to compare deoxyribonucleic acid (DNA) strings for example. They tends to find subsequences that are similar between two sequences. They are used to find trajectory parts that are similar between different trajectories. However, they do not take into account the temporal aspect. Dynamic Time Warping (DTW), used in [20, 55, 59], is a similar method encompassing temporal differences between the sequence elements. Both methods are efficient to classify gestures or really

different trajectories, but are not suited for our purpose.

Fréchet distance used in [56, 57, 60] and Hausdorff distance used in [57, 60] are very similar measures used to quantify the maximal distance between two curves. However, they only take into account the most separated points of each curve. As they do not encompass the other trajectory differences, they are too simple quantitative measures to make fair comparisons of trajectories.

Gaussian Models (GM) or Gaussian Mixture Models (GMM) are used to detect semantic regions [42, 63, 62, 64]. However, as they represent distributions of points and do not contain information about the temporal sequence, they are hardly usable to analyze trajectories in their original form. However, mixed with Hidden Markov Models, they become efficient solutions, as it will be later presented in this thesis.

Correlated Random Walk (CRW) models are used in some applications previously described [71, 72]. They will be also used in this this thesis and presented later more thoroughly.

Hidden Markov Model (HMM) or its declinations (CHMM, GHMM,...) are the main methods used with trajectories: to compare gestures [16, 17], car or pedestrian trajectories [33, 26, 32, 35, 48], or to cluster trajectory datasets [45, 47, 46, 58]. Their main interest lies in their applicability to nearly all kinds of data. The learning techniques are also well known and the method can be used with input vectors of different lengths (different number of points per trajectory). However, they are quite computationally intensive methods and do not fit optimally to the data features, as trajectories are rarely a Markovian process (memoryless). HMM will however be used in this thesis as a comparison to the PDM method or as complementary solution when PDM cannot be used.

Finally, all these models are using a large panel of different features, such as the spatial and temporal coordinates of the trajectory points, the curvature or the torsion of the trajectories, the different derivatives, their approximation by spline or Bézier curves, ... These features are not bound to the different models, but they have an important influence on the analysis. A discussion on the selected features will be presented later in this thesis..

2.3 Conclusion

This chapter offers a broad overview of the various sub-domains and current work in trajectory analysis. In particular, it presents work performed by computer vision scientists, who, over the last ten years, have been principally responsible for developing the field of trajectory analysis. This chapter also shows different techniques which may be used to analyze trajectories, coming mainly from signal-processing and machine-learning theory. This general description allows for the anchoring of this thesis, and the methods it introduces, in the more general field of trajectory analysis.

Chapter 3

Point Distribution Model

This chapter presents the Point Distribution Model (PDM) as a tool to compare trajectories. The PDM was introduced by Cootes [4]. As a deformable template, it is mainly used for the analysis of shape variations between images. Given that the spatial representation of a trajectory can be considered as a complex shape, we thought that the PDM could be applied to the analysis of trajectories. In his semester project [88], Florian Luisier made the first attempt to apply PDM to trajectories. From this first experience, we were able to create a new methodology [89].

3.1 PDM formalization

The trajectories have to be sampled with the same number of points in order to be analyzed. Each trajectory k can be represented as an ordered set of N points $\left\{\pi_1^k \cdots \pi_N^k\right\}$. Each point π_n^k is composed of a combination of its spatial components x_n^k, y_n^k or z_n^k, ..., and its temporal component t_n^k. In two dimensions, if both spatial and temporal data are used:

$$\pi_n^k = \left[x_n^k \; y_n^k \; t_n^k\right]^T. \tag{3.1}$$

Therefore, the trajectory τ_k can be expressed as :

$$\tau_k = \left[x_1^k \ldots x_N^k \; y_1^k \ldots y_N^k \; t_1^k \ldots t_N^k\right]^T. \tag{3.2}$$

A Principal Components Analysis (PCA) of the trajectory dataset is then computed. This tool is presented in Section A.6.1. If $\boldsymbol{P}$ is the resulting matrix of the eigenvectors,

$$\tau_k = \overline{\tau} + \boldsymbol{P} \cdot B_k \tag{3.3}$$

$$B_k = \boldsymbol{P}^{-1}(\tau_k - \overline{\tau}), \tag{3.4}$$

where $\overline{\tau}$ is the average trajectory. Equations 3.3 and 3.4 correspond to the affine transformation from the deformation space (B_k) to the trajectory space (τ_k), respectively to the projection from the trajectory space to the deformation space. As $\boldsymbol{P}$ is constructed as an orthonormal basis, this transformation is a change of referential. To be more explicit, the PCA transforms the trajectory space in two steps. First,

it subtracts the average trajectory from the dataset so as to keep only the deviations from the mean trajectory. Second, it changes the spatial referential in order to have the maximal variability in the first dimension and a decreasing variability in the following dimensions.

If all the three components are used (x_n^k,y_n^k, and t_n^k), the maximal dimensionality of the dataset will be $r = argmin(3N, k-1)$, when all the trajectories are independent. If there are linear combinations between the trajectories, the dimensionality of the dataset will be reduced.

The cumulative energy, defined in Equation A.41, can be used to evaluate how much variability is contained in the first dimensions. It allows the dimensionality reduction of the deformation space to a subspace containing the major part of the trajectories variability. 90% to 95% of the variability are often used, and they often represent less than 10 dimensions.

An axis of the deformation space is called a deformation mode. To establish a spatial representation of a deformation mode, a position must be chosen on the corresponding axis in the deformation space. As this position is on one axis of the referential frame, only one component of the corresponding vector will be non-null. As a result, the transformation of this position in the trajectory space becomes:

$$\delta_r = \overline{\tau} + P_r \cdot a, \tag{3.5}$$

where a is the chosen position of the r^{th} deformation axis and P_r is the r^{th} column of $\boldsymbol{P}$. If multiple positions are selected on the deformation axis, the differences between the corresponding trajectories (δ) show how the mode influences the trajectories. To display the effects of a mode, good values of a can be either ± 1 or the minimal and maximal values observed in the dataset for the specific mode.

3.2 Trajectory PDM examples

The mathematical description of the PDM is perhaps not so easy to understand. Furthermore, it is perhaps difficult to concretely visualize what the PDM really does. Therefore, we will provide some examples of PDM analyses on artificially generated trajectories.

The dimensionality of the trajectories increases linearly with the number of points. A trajectory of twenty points corresponds to forty dimensions if only x and y coordinates are used, and to eighty dimensions if x, y, z, and t are used. The PDM is quite interesting, as it allows the visualization of the differences between the trajectories.

3.2.1 Linear trajectories

For the first examples, the trajectories will be purely linear. The top left graphic of Figure 3.1 shows the twenty-one trajectories and their sampled points. These trajectories are just segments. As the trajectories are sampled with lines orthogonal to the x axis, $x = [0, 1, 2, \ldots, 20]^T$ for all trajectories. y is proportional to x: $y = a \cdot x$ with $a \in [-1, -0.9, \ldots, 0.9, 1]$. The average trajectory is a horizontal segment and the whole set of trajectories has just one degree of freedom: the slope of the trajectories. If we look at the result of the PDM analysis (top right graphic of Figure 3.1), the trajectories lie on the first axis of the deformation modes and no trajectory contributes to the second and following deformation modes.

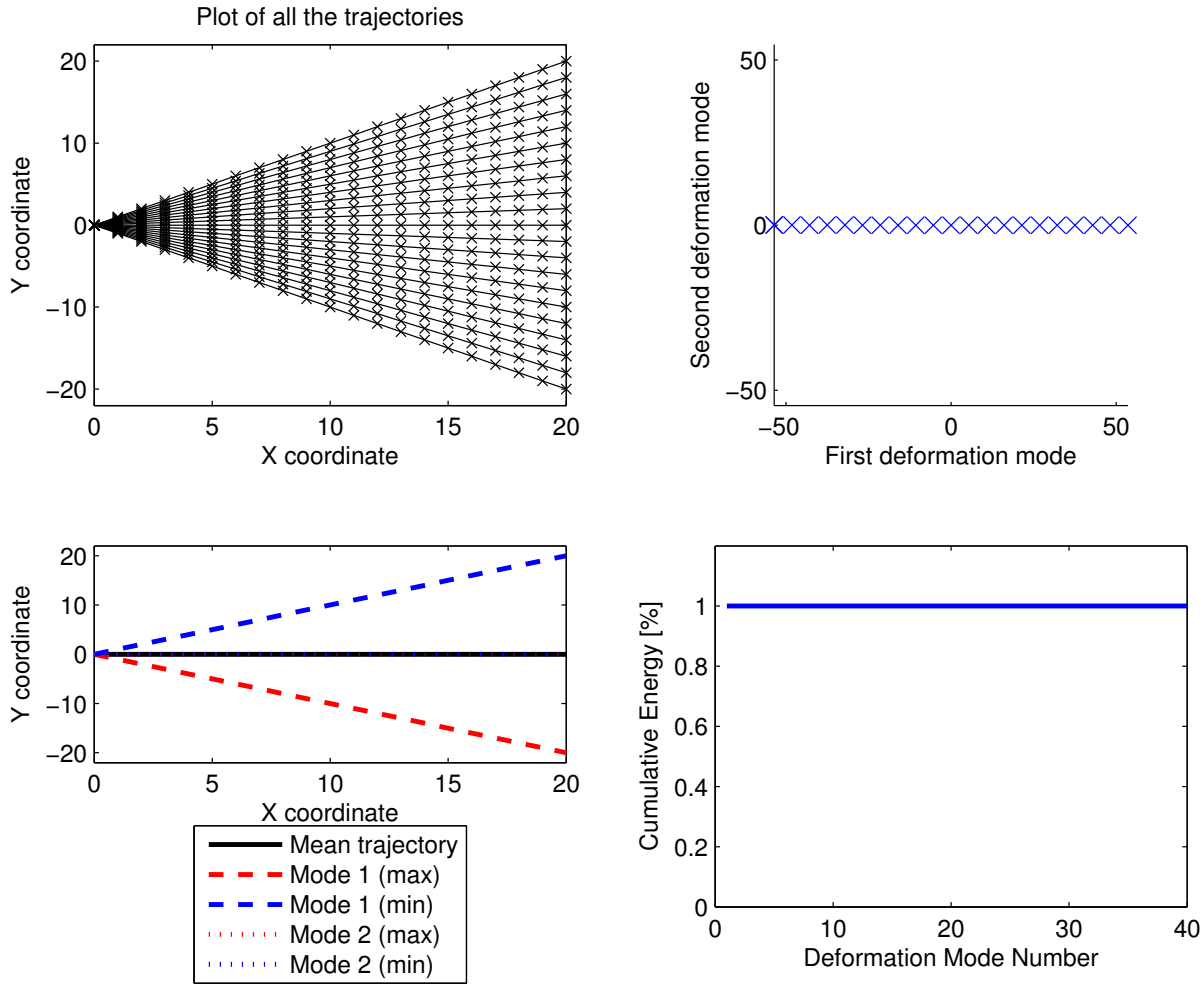


Figure 3.1: PDM analysis applied to artificial trajectories. The twenty-one trajectories are purely linear: $y = a \cdot x$ with $x = [0\ 1\ 2 \ldots 20]^T$ and $a \in [-1\ \ -0.9 \ldots 0.9\ 1]$. The top left, top right, bottom left, and bottom right graphics show respectively the trajectories, their transformation in the PDM space (only the first two modes/dimensions), the spatiotemporal influence of the first two deformation modes, and the cumulative energy of the deformation modes

As the whole dataset has just one degree of freedom, it is not surprising that only one mode is needed to represent it. It can also be observed that the points are equally separated in the PDM space.

Some computations will help understand how this works. As the trajectories are encoded like in Equation 3.2, the trajectory τ_a corresponds to the vector:

$$\tau_a = [0\ 1 \ldots 20\ 0\ a \ldots 20a]^T, \tag{3.6}$$

and the average trajectory is:

$$\overline{\tau} = [0\ 1 \ldots 20\ 0\ 0 \ldots 0]^T. \tag{3.7}$$

Thus, the difference between a trajectory and the average trajectory is:

$$\tau_a - \overline{\tau} = [0\ 0 \ldots 0\ 0\ a \ldots 20a]^T. \tag{3.8}$$

As the dataset contains just one degree of freedom,

$$\tau_a - \overline{\tau} = P_1 * b_a^1, \tag{3.9}$$

where b_a^1 is the first element of vector B_a, all the other elements being equal to zero. Moreover, as P_1 is by definition a unitary vector,

$$b_a^1 = ||\tau_a - \overline{\tau}|\,| = \sqrt{\sum_{i=1}^{20}(ai)^2} \approx a \cdot 53.6. \tag{3.10}$$

These values can be found in the top right graphic of Figure 3.1: the cross at the farthest position on the right corresponds to the trajectory built with $a = 1$. If we look at the spatial representation of the deformation modes (bottom left graphic of Figure 3.1), we can see that the first deformation mode corrects the slope of the trajectory and that the second deformation mode is null. As for the cumulative energy (bottom right graphic of Figure 3.1), it jumps from 0 to 100% with the first mode, as the whole dataset can be explained with just one dimension.

3.2.2 Linear trajectories with noise

To generate the second trajectory example, a Gaussian noise of unitary variance and zero mean ($\mathcal{N}(0, 1)$) was added to the trajectories of Figure 3.1. Figure 3.2 shows the trajectories and the results of the PDM analysis. Similarly to the prior experiment, the trajectories in the PDM space are more or less collinear, with some noise added to the point locations. As for the spatial representation of the deformation modes, the first mode corresponds mainly to the slope of the trajectories and the following modes represent the Gaussian noise. The cumulative energy is represented for more than 98% by the first mode and more than ten additional modes are needed to explain the other 2% (Gaussian noise).

3.2.3 Only noise

The present example is similar to the previous one, but with null slope trajectories. Thus, only the Gaussian noise ($\mathcal{N}(0, 1)$) is significant. Figure 3.3 shows the trajectories and the results of the PDM analysis.

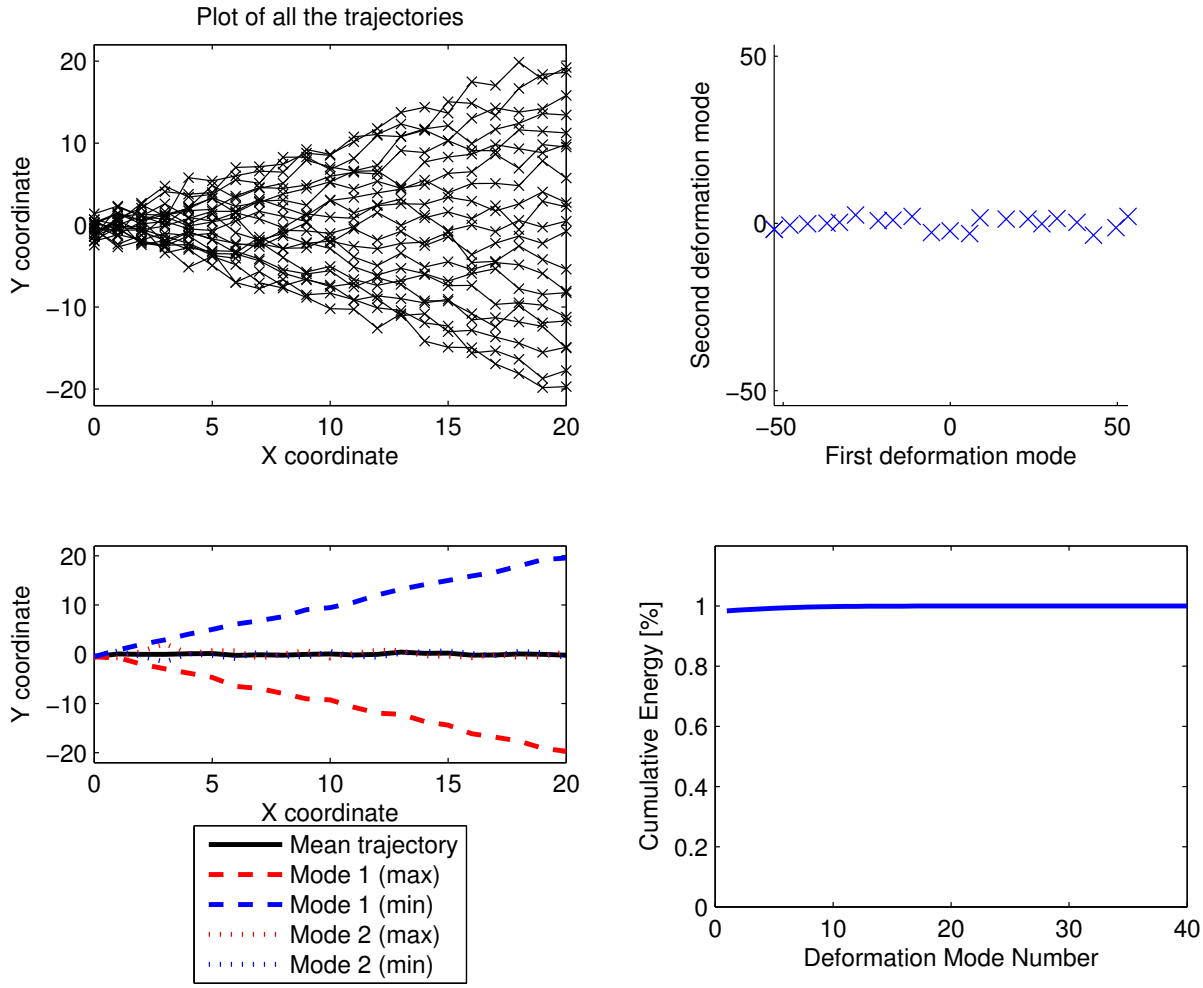


Figure 3.2: PDM analysis applied to artificial trajectories. The trajectories shown here are the result of adding a Gaussian noise ($\mathcal{N}(0,1)$) to the trajectories of Figure 3.1. The top left, top right, bottom left, and bottom right graphics show respectively the trajectories, their transformation in the PDM space (only the first two modes/dimensions), the spatiotemporal influence of the first two deformation modes, and the cumulative energy of the deformation modes

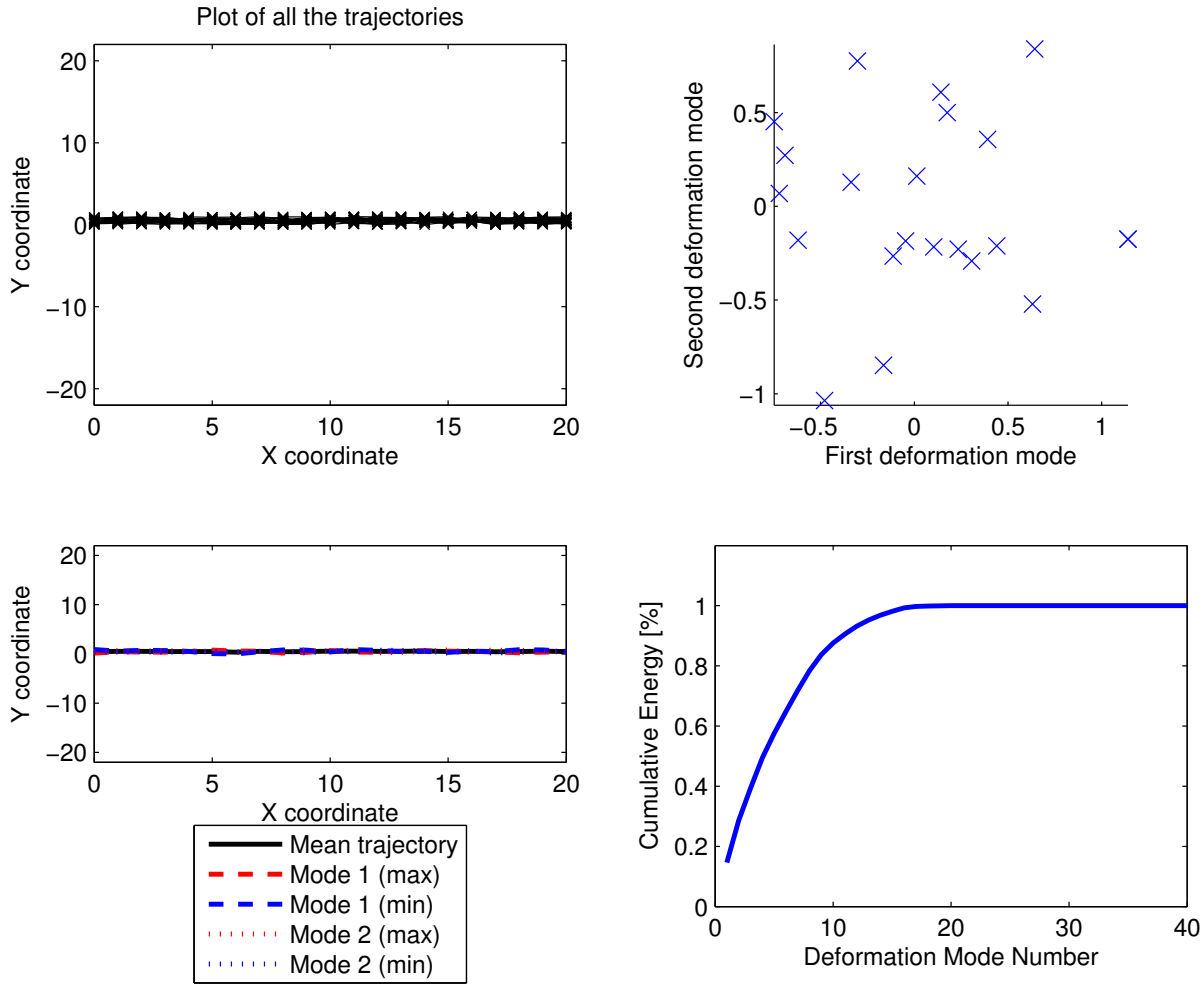


Figure 3.3: PDM analysis applied to artificial trajectories. The y coordinates of the trajectories corresponds to a Gaussian noise ($x = [0\ 1\ 2 \ldots 20]^T$ and $y \sim \mathcal{N}(0, 1)$). The top left, top right, bottom left, and bottom right graphics show respectively the trajectories (the same range as in the other plots were kept for better comparison), their transformation in the PDM space (only the first two modes/dimensions), the spatiotemporal influence of the first two deformation modes, and the cumulative energy of the deformation modes

In this case, the trajectories are randomly distributed in the PDM space. However, the contributions to the different modes are much smaller. The spatial representations of the modes are highly variable as they must describe the noise. Also, the cumulative energy is increasing very slowly and more than fifteen modes are needed to explain 98% of it. These results are not surprising, as it is difficult to find correlation in multivariate Gaussian noise.

3.2.4 Two linear sets with noise

To build this last example, we created two sets of trajectories similar to the ones in Figure 3.2. A positive value of 5 was then added to all the y coordinates of the first set and a negative value of -5 to the y coordinates of the second set. The resulting trajectories and their analysis with a PDM are shown in Figure 3.4. As we can see in the spatial representation of the modes, the first mode is mainly linked to the slope of the trajectories and the second one corrects the vertical position of the trajectories. It is also noticeable that the two distributions are perfectly separable in the first two modes of the PDM. Moreover, the cumulative energy shows that more than 95% of the variance is explained by the first two deformation modes.

3.3 Spatiotemporal trajectories

A third dimension, such as time, can easily be included in the analysis.

3.3.1 Linear trajectories

Figure 3.5 represents the extension to three dimensions of the first example in two dimensions (Figure 3.1). A set of $K = 21$ trajectories is created with $N = 100$ points per trajectory. Supposing that $n \in [1 \dots N]$, the n^{th} point of the k^{th} trajectory, π_n^k, is determined by the following equation:

$$\begin{aligned} \pi_n^k &= \left[x_n^k \; y_n^k \; t_n^k\right]^T, \\ x_n^k &= a_k \cdot t_n^k, \\ y_n^k &= -a_k \cdot t_n^k, \\ t_n^k &= \frac{n}{10}, \end{aligned} \tag{3.11}$$

where $a_k \in [-1 \; -0.9 \; \dots \; 0.9 \; 1]$. The resulting analysis is very similar to the results of the corresponding example in two dimensions. As there is only one degree of freedom in the set, there is also only one dimension of interest in the deformation space, the other dimensions being null.

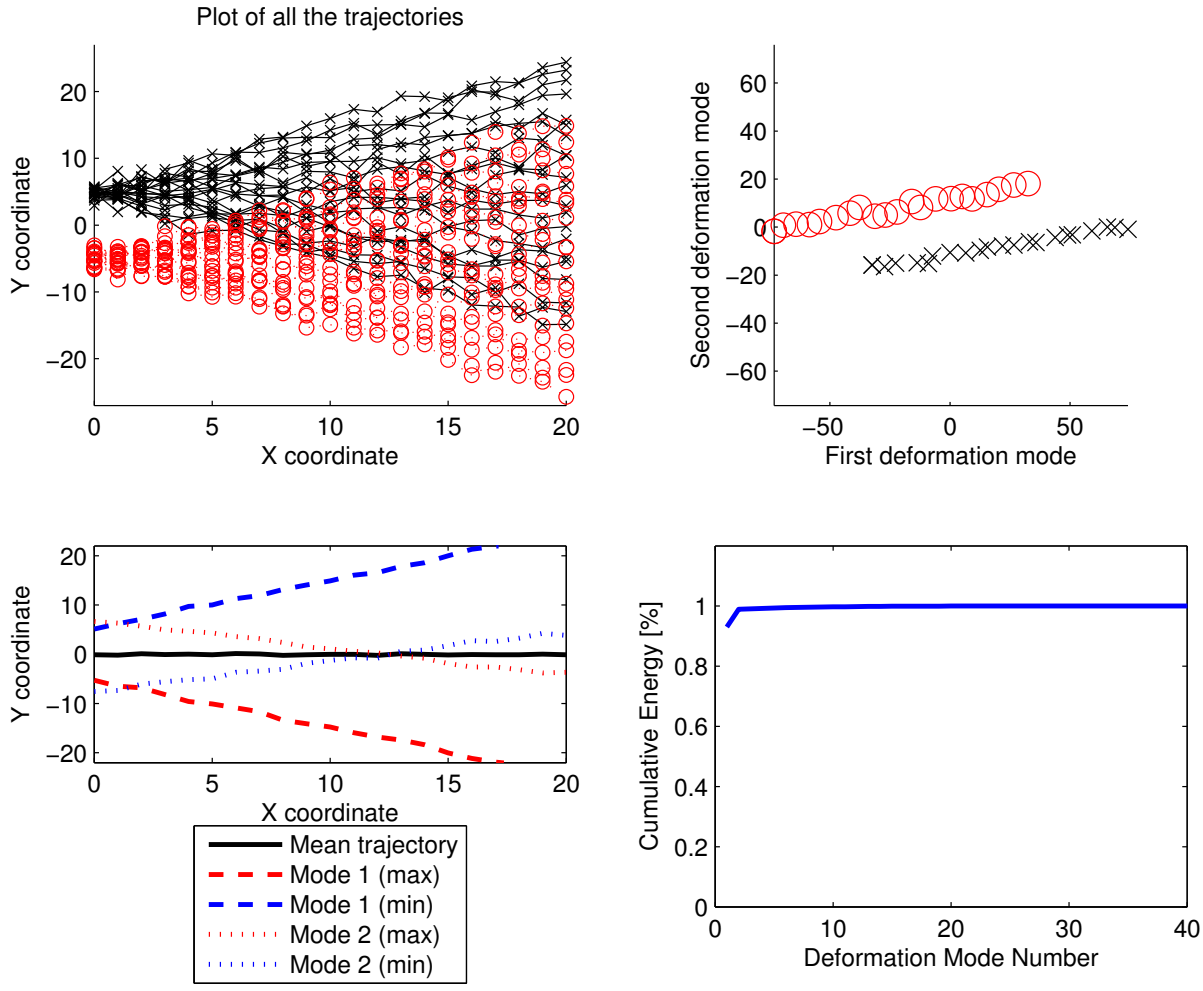


Figure 3.4: PDM analysis applied to two sets of artificial trajectories. The two sets were generated as in the example illustrated by Figure 3.2, but y was added a constant value of ± 5. The top left, top right, bottom left, and bottom right graphics show respectively the trajectories, their transformation in the PDM space (only the first two modes/dimensions), the spatiotemporal influence of the first two deformation modes, and the cumulative energy of the deformation modes

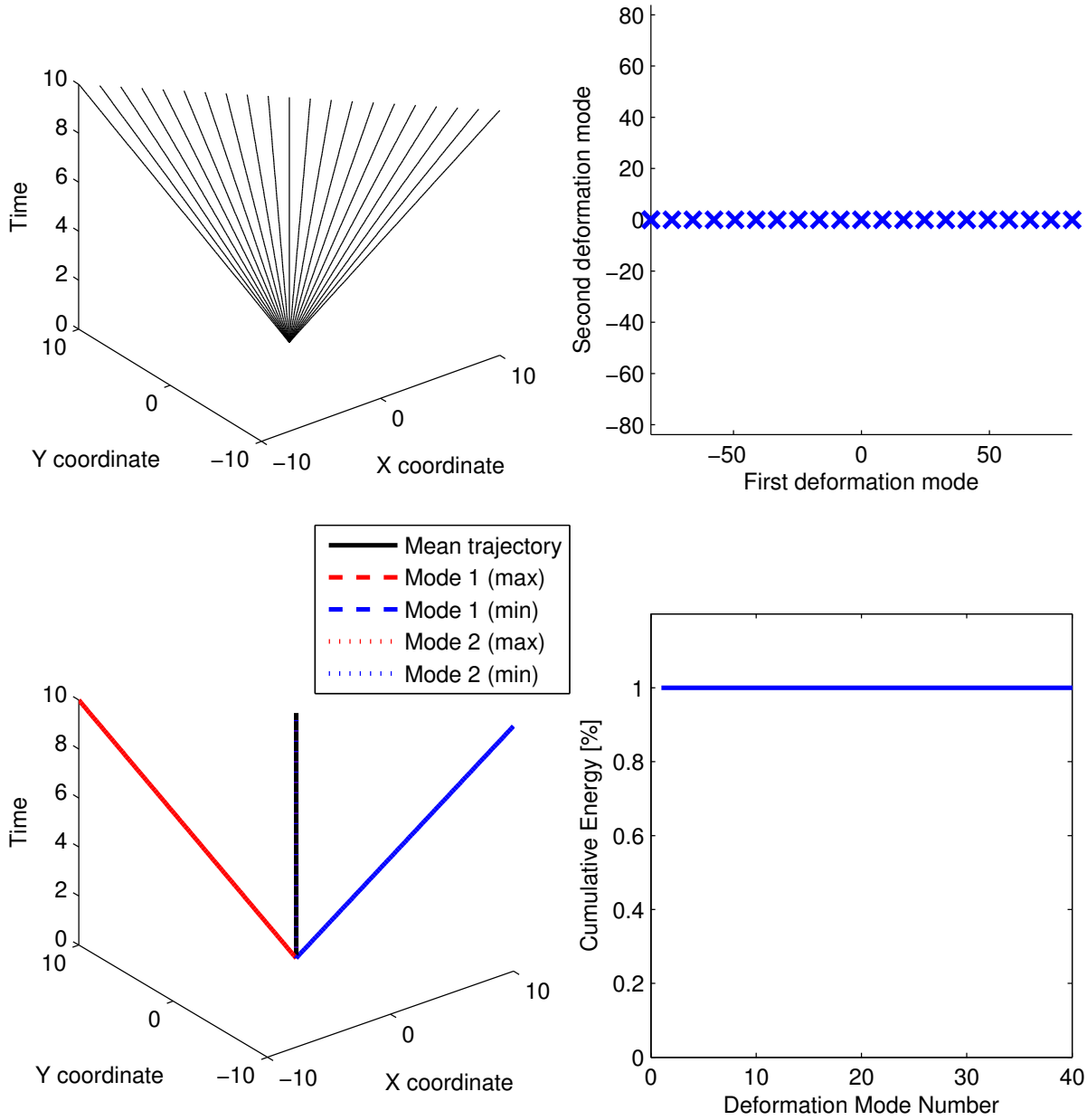


Figure 3.5: PDM analysis applied to spatiotemporal trajectories. The twenty-one trajectories are purely linear: $y = -a \cdot t$ and $x = a \cdot t$, with $t = [0\ 0.1\ 0.2 \ldots 10]^T$ and $a \in [-1\ \ -0.9 \ldots 0.9\ 1]$. The top left, top right, bottom left, and bottom right graphics show respectively the trajectories, their transformation in the PDM space (only the first two modes/dimensions), the spatiotemporal influence of the first two deformation modes, and the cumulative energy of the deformation modes

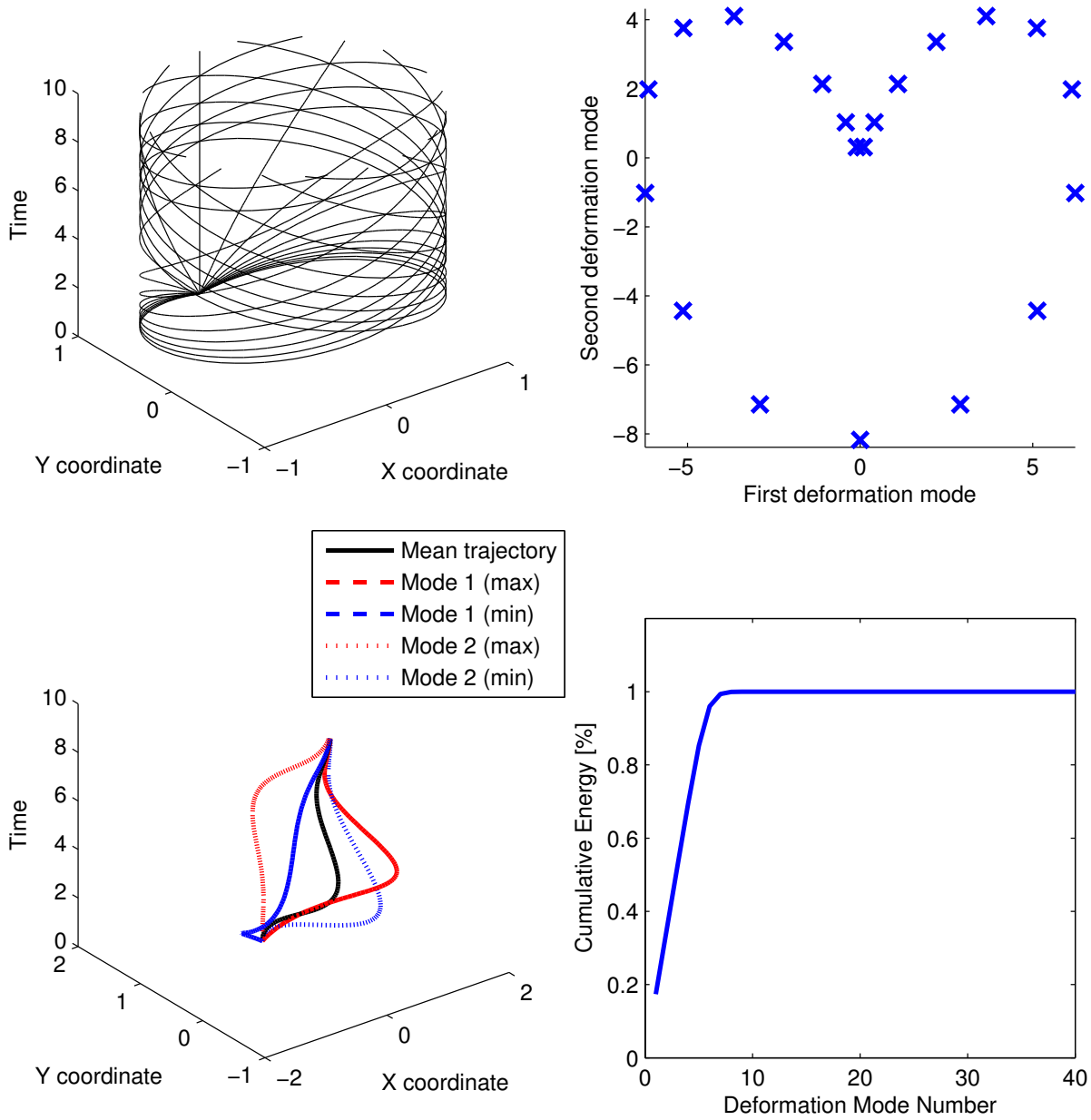


Figure 3.6: PDM analysis applied to spatiotemporal trajectories. The twenty-one trajectories are nonlinear: $y = -sin(a \cdot t)$ and $x = cos(a \cdot t)$, with $t = [0\ 0.1\ 0.2 \ldots 10]^T$ and $a \in [-1\ -0.9 \ldots 0.9\ 1]$. The top left, top right, bottom left, and bottom right graphics show respectively the trajectories, their transformation in the PDM space (only the first two modes/dimensions), the spatiotemporal influence of the first two deformation modes, and the cumulative energy of the deformation modes

3.3.2 Non-linear trajectories

The second example of spatiotemporal trajectories shows a non-linear system, determined by the following equations:

$$\begin{aligned} \pi_n^k &= \left[x_n^k \; y_n^k \; t_n^k\right]^T, \\ x_n^k &= sin\left(a_k \cdot t_n^k\right), \\ y_n^k &= cos\left(-a_k \cdot t_n^k\right), \\ t_n^k &= \frac{n}{10}, \end{aligned} \tag{3.12}$$

the other parameters being the same as in the previous experiment. These equations correspond to a movement along a circle of unitary radius, with a variable speed. The resulting trajectories are helicoids. Figure 3.6 shows the trajectories and the results of the PDM analysis. The spatiotemporal influence of the deformation modes becomes very difficult to link to the dataset and the mode values are difficult to understand. As sin and cos functions are non-linear, the resulting modes are no longer on a line. Moreover, even if there is only one underlying degree of freedom (a_k), the system cannot be reduced to only one dimension as non-linear functions are used. As a result, around 10 modes are needed to make a linear approximation of the sin and cos functions, as one can see in the cumulative energy plot.

3.3.3 Clusters separation

The third example shows how two kinds of trajectories are separated in the deformation space. Two sets of twenty helicoidal trajectories are generated, in a way similar to the previous experiment (Equation 3.12). For the first set, $a = 1$ and for the second set $a = -1$. In both cases, Gaussian noise ($\mathcal{N}(0, 0.1)$) is added to all coordinates (x, y, and t). Thus, there are two types of helicoids and each type contains twenty trajectories whose variations are only due to the noise. Figure 3.7 shows the results of the PDM analysis. The top right plot shows that the separation of the two sets is well done and that one mode is sufficient to explain the difference between the sets. The explanation is quite obvious. Each average trajectory of each set can be expressed as a vector in $3N$ dimensions. Thus, an axis linking these two vectors and forming the line going through their ends can be found. As the distance between the two vectors' ends is much larger than the added Gaussian noise, the axis of maximal variance will be very close to this axis. Considering the spatiotemporal influence of the deformation modes also makes it much easier to understand their current shape than in the previous experiment. Finally, the plot of the cumulative energy shows that most of the variance is contained in the first deformation mode.

3.3.4 Scale influence

The spatiotemporal example described in this section is just a slight modification of the previous one. The scale of the temporal axis has been multiplied by 1000, which would be the same as modifying the axis unit from seconds to milliseconds. As a result, the variance in the temporal dimension is multiplied by 10^6. Thus, the variance of the temporal dimension has a much greater impact on the PCA than the

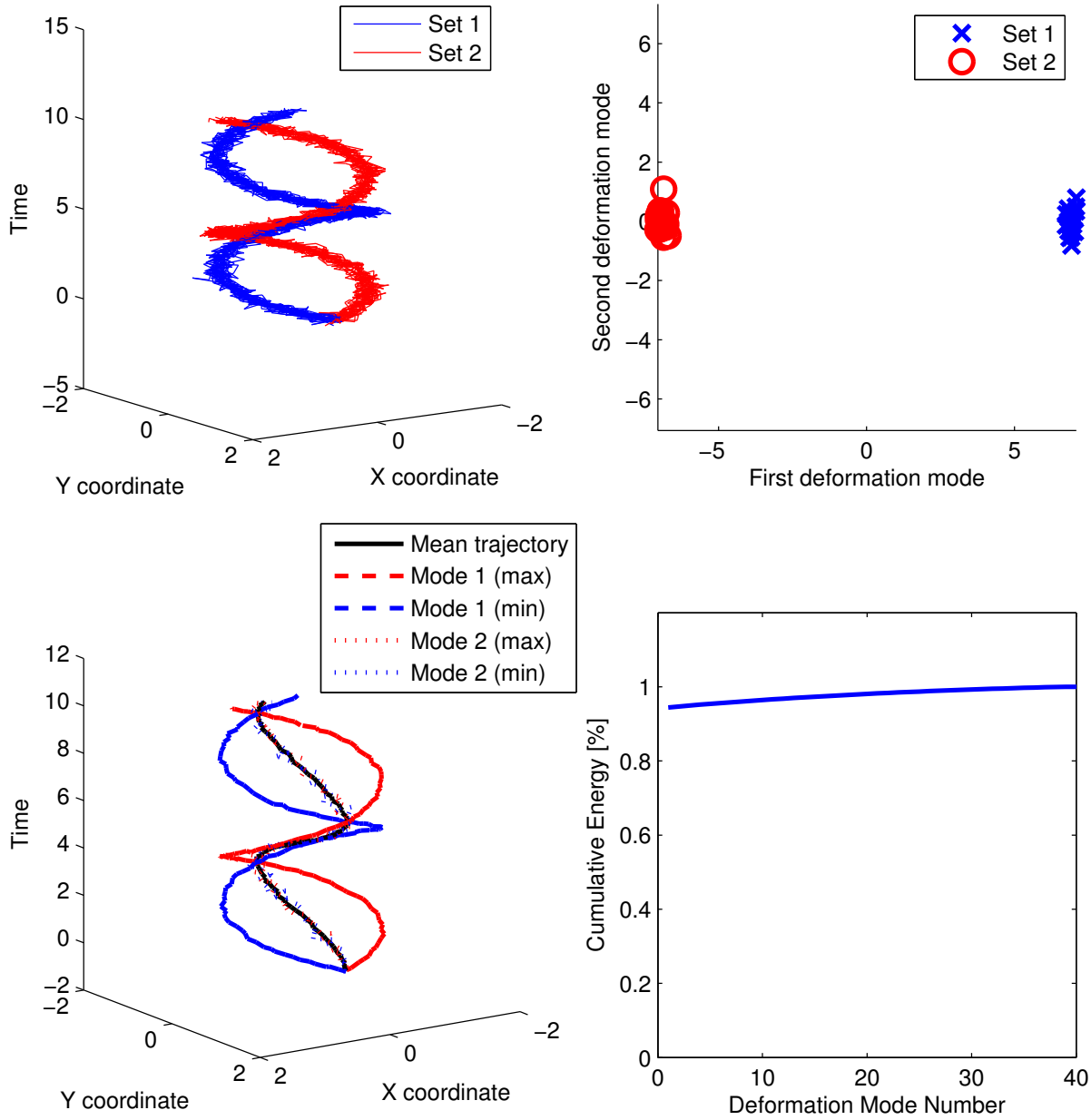


Figure 3.7: PDM analysis applied to two sets of twenty spatiotemporal trajectories. The first set corresponds to the same kind of helicoids as shown in Figure 3.6, with $a = 1$ and Gaussian noise being added to all coordinates ($\mathcal{N}(0, 0.1)$). The second set is created with $a = -1$ and with the same kind of added noise. The top left, top right, bottom left, and bottom right graphics show respectively the trajectories, their transformation in the PDM space (only the first two modes/dimensions), the spatiotemporal influence of the first two deformation modes, and the cumulative energy of the deformation modes

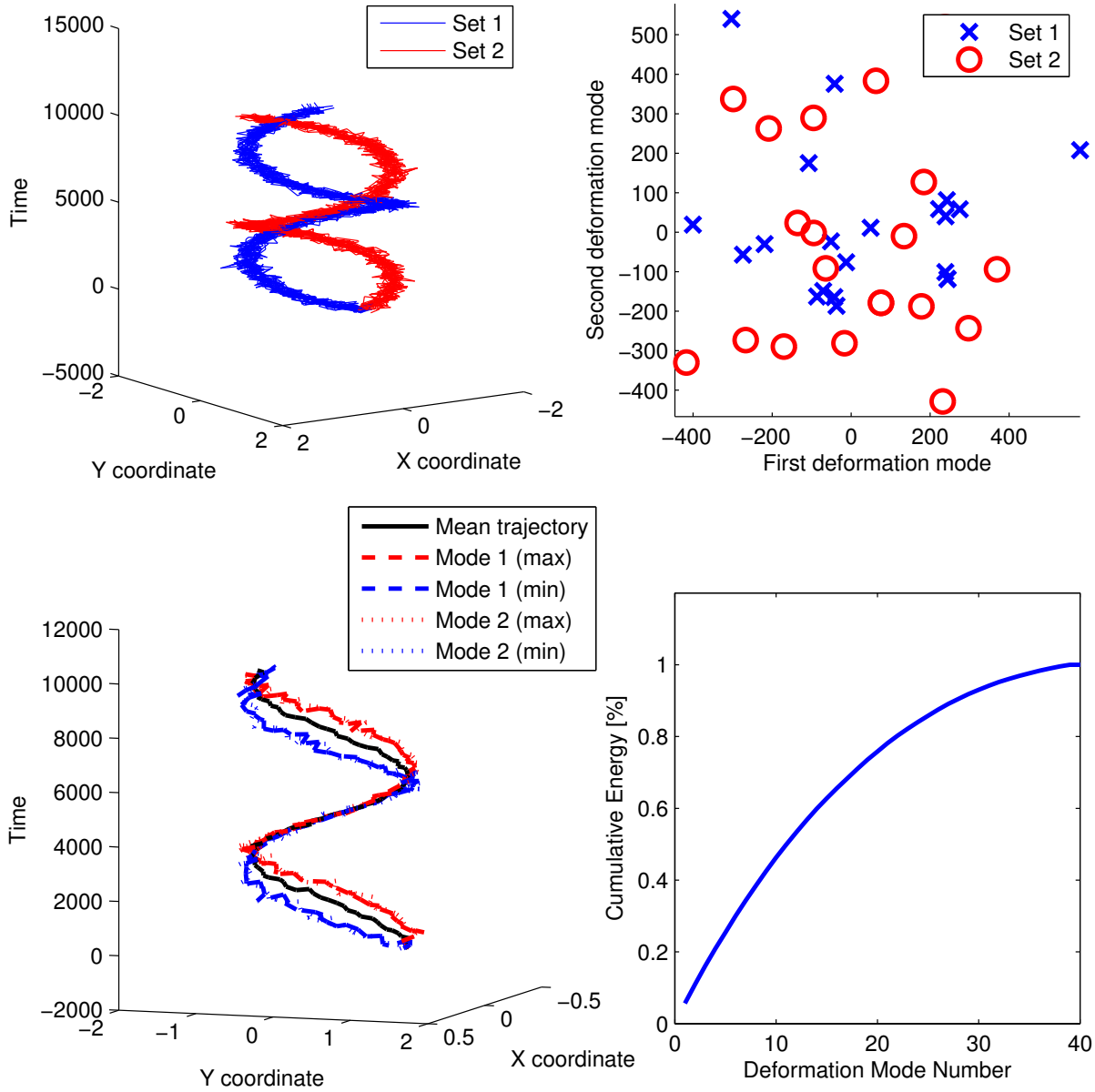


Figure 3.8: PDM analysis applied to two sets of twenty spatiotemporal trajectories. The two sets are nearly identical to Figure 3.7. Only the scale of the temporal axis has been changed, as it was multiplied by 1000, which could correspond to a change of temporal units (*e.g.*, from seconds to milliseconds). The top left, top right, bottom left, and bottom right graphics show respectively the trajectories, their transformation in the PDM space (only the first two modes/dimensions), the spatiotemporal influence of the first two deformation modes, and the cumulative energy of the deformation modes

variance on the spatial axes. Figure 3.8 shows the trajectories and the corresponding PDM analysis. Contrary to the previous example, the two sets of trajectories are no longer separable considering only the first 2 modes. Looking at Equation 3.12, one can see that the t coordinate is independent of a. As a result, both sets of trajectories have more or less the same temporal positions, excepting the trajectories made of Gaussian noise. As this noise is multiplied when the temporal scale changes, the variance caused by this noise is greater than the variance caused by the spatial differences between the sets. The PCA first focuses on the temporal noise and is therefore no longer efficient to separate the two trajectory sets.

One way to bypass this problem is to divide each coordinate by the variance of the whole dataset on the corresponding axis. However, when we are working with similar units, a possible greater variance on one axis will be reduced by this technique. It is thus not always optimal. It is crucial to understand this problem and to take it into account when different types of data are mixed.

3.4 Conclusion

This chapter has shown that PDMs can be used to model trajectory datasets. As the examples were synthetic, acquisition and sampling has been left out of the problematic. Thus, the presentation focused on the way the PDM works and how it can be used to easily apprehend the difference between trajectories. The importance of the cumulative energy parameter has been described and the transformation of the deformation modes into the spatial domain has been shown, so as to help the reader to understand how the PDM transforms the trajectories into the space of deformation modes. The principal advantages and drawbacks have also been introduced, such as the dimensionality reduction, the classification, or the problem arising from handling different kinds of data. The next chapters will present solutions as well as examples of applications of this model to real data.

Chapter 4

Trajectory sampling

A trajectory is in essence a continuous function of the position of an agent moving in time and space. However, the intent of this thesis is not to compare trajectories as continuous functions, but as sequences of points. Therefore the underlying continuous movement of the physical system will not be approximate. Thus, the sampling of the trajectories becomes an important part of their analysis. As trajectories can be composed of thousands of points, it is important to choose points describing the trajectory well enough. Moreover, as a lot of analysis methods (such as the PCA) require the same number of points per trajectory, resampling methods are needed.

There are two main ways of sampling trajectories: the time-based and the space-based methods. Data acquisition produces a first sampling of the trajectories. Most of the time, the sampling is time-based as it is based on an internal clock (processor, camera, ...). For example, if the trajectory is acquired with a vision-based tracking system, the position of the agent will be extracted from each frame and the trajectory points will be acquired at the frame rate of the camera. Similarly, a GPS or another electronic device often provides a time-based acquisition. In the signal processing of sound, time-based sampling is also the standard. Pure space-based sampling is more difficult to achieve, as nearly all digital devices are equipped with a digital clock. However, the use of optical barriers acquiring the time when the agent goes through them, corresponds to a space-based acquisition. Similar space-based methods could be developed for the acquisition.

It is important to point out that for both space-based and time-based acquisitions, the spatial and temporal positions of the agent are saved at the acquired points. Thus the difference between time-based and space-based acquisitions is not linked to the kind of data we are working with, but to the way these data are acquired. Afterward, these data can always be resampled spatially or temporally.

This chapter presents different solutions applicable to a large panel of trajectories. It will be divided between the examination of the solutions for trajectories generated on a circuit and two case studies which can be easily extended to other trajectories.

Figure 4.1: Top view of an experiment in a circuit with a mobile robot driving on it

4.1 Circuit

Many of the experiments were performed with a mobile robot driving continuously on a closed circuit. Figure 4.1 shows the setup of one of the experiments done in a circuit. In this case, each lap made by the agent was considered as an individual trajectory. Thus, if the agent did K circuit laps, we will get K trajectories of the same agent.

4.1.1 Time-based sampling

To resample these laps, the first solution was to apply a time-based sampling. Thus each lap was divided with the same number (n) of sampled points, separated by the same time interval ($\frac{Duration_{Lap}}{n}$). Figure 4.2 shows two trajectories sampled with twenty points. Each trajectory corresponds to a different behavior. The first behavior has a constant forward speed during the lap, resulting into more or less equidistant sampled points. On the contrary, the second behavior has a varying speed: the agent drives faster on the lower part and slower on the upper part of the circuit. Thus, the distance between the sampled points is much shorter on the upper part than on the lower part.

The advantages of time-based sampling lie in its simplicity and its applicability: this method can be easily applied to all kinds of trajectories and is very fast to compute. However, Figure 4.2 shows its major drawback : if the trajectories are not generated with an agent moving at a constant velocity, sampled points not at all at the same place on the circuit will be compared. This in turn results in a much more complicated and unintuitive analysis of the trajectories.

Moreover, as all the points are separated by the same time interval, the temporal values of the sampled points (t_i) are no longer of any interest. All these values are on one line and $t_i = \frac{Duration_{Lap} \cdot i}{n}$. Figure 4.3 shows the temporal profiles of four trajectories. Only one parameter remains in those temporal profiles: the lap duration, which corresponds to the slope of the line. As a comparison, Figure 4.4 shows the temporal profiles resulting of a space-based sampling of the same trajectories. The speed variations can

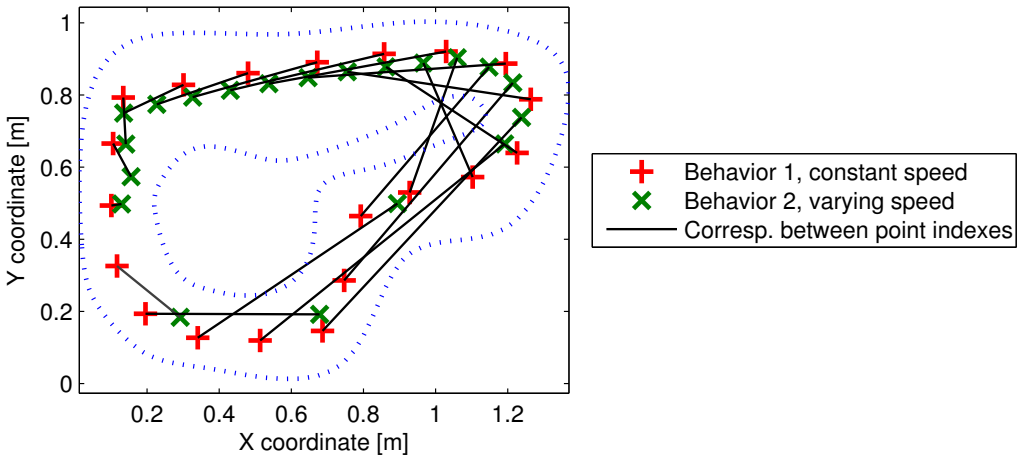


Figure 4.2: Points resulting of a time-based sampling applied to two trajectories. The first trajectory is the result of an agent moving with a constant forward speed. In the second trajectory, the agent is moving faster on the lower part of the circuit and slower on the upper part

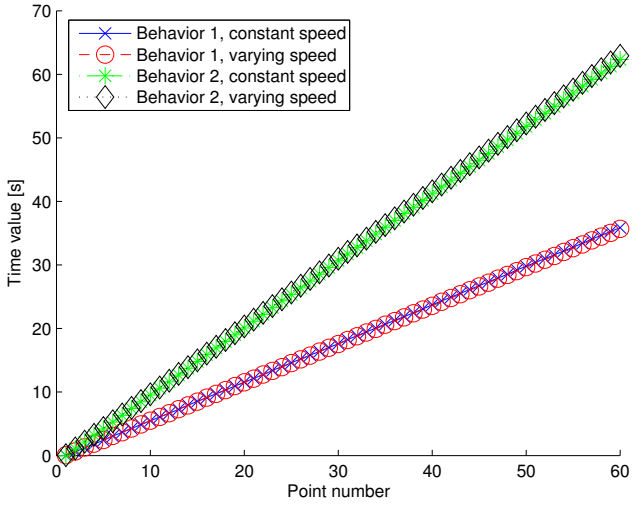


Figure 4.3: Temporal profiles of four trajectories resulting of a time-based sampling. This sampling completely removes the temporal variations between the points: those produced with or without a constant speed are on the same line

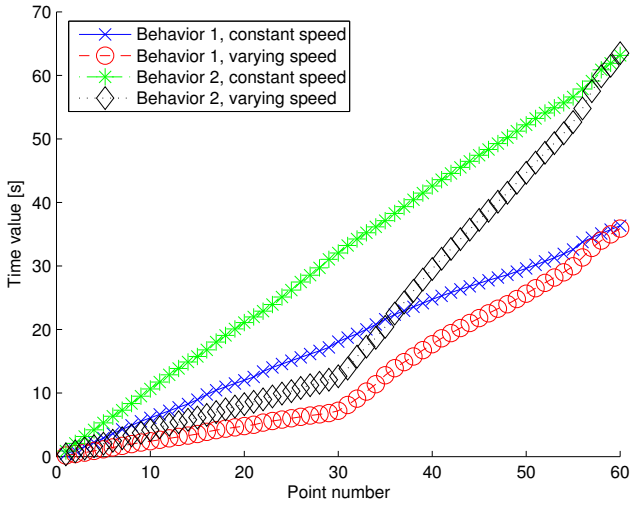


Figure 4.4: Temporal profiles of four trajectories resulting of a space-based sampling. The temporal variations between the sampled points are kept

be easily spotted.

4.1.2 Space-based sampling

To solve the problems of time-based sampling, we developed space-based methods that sample the trajectories of an agent moving in a circuit. The first experimental setup used for this project encouraged us to develop this kind of analysis. Radio-guided miniature cars were driven by human pilots who sometimes missed a curve, leading to a crash and to a complete mess of the trajectory at a specific place on the circuit. When space-based sampling was used, the influence of the crashes was limited to the sampled points of the crash area. On the contrary, with a time-based sampling, the crash lead to a delay influencing all the sampled points.

Humans do space-based sampling naturally when they observe moving agents (e.g. at a specific time, the car goes through the crossroads, the robot arrives into a specific area, or the pedestrian passes through the door). Comparing trajectories near specific landmarks is our underlying intention.

4.1.2.1 Sampling with orthogonal lines

A first solution to the problem of space-based sampling is to use a reference trajectory and to sample all the other trajectories with lines orthogonal to this reference. Figure 4.5 shows how this method can be applied to trajectories generated in a circuit. It is a simple method that can be applied to multiple cases if the trajectories to sample have a similar shape. However, it is not perfect. The choice of the reference trajectory can influence a lot the resulting sampled points. Moreover, two major problems can arise during the sampling. The first one is represented in Figure 4.6(a). As the orthogonal lines can cross one another between the reference trajectory and the trajectory to sample (dashed line), the order

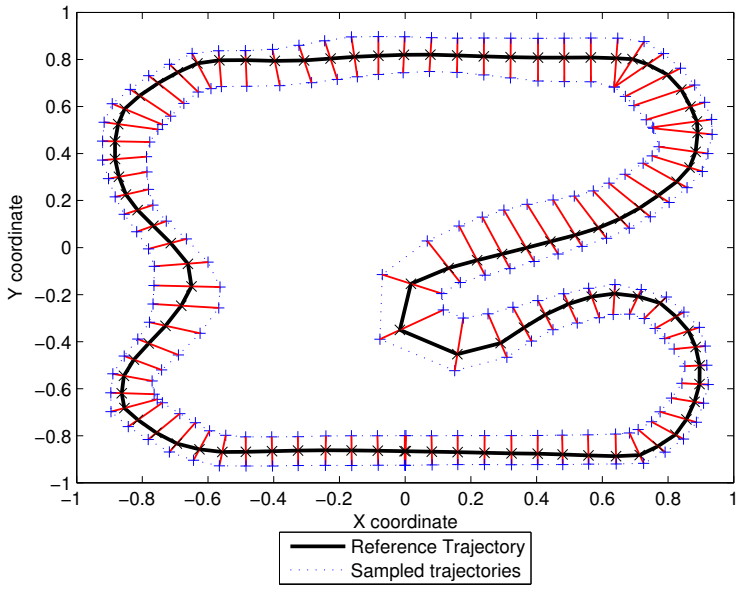


Figure 4.5: Two trajectories, one on the inner part of the circuit and the other one on the outer part, are sampled using segments orthogonal to a reference trajectory

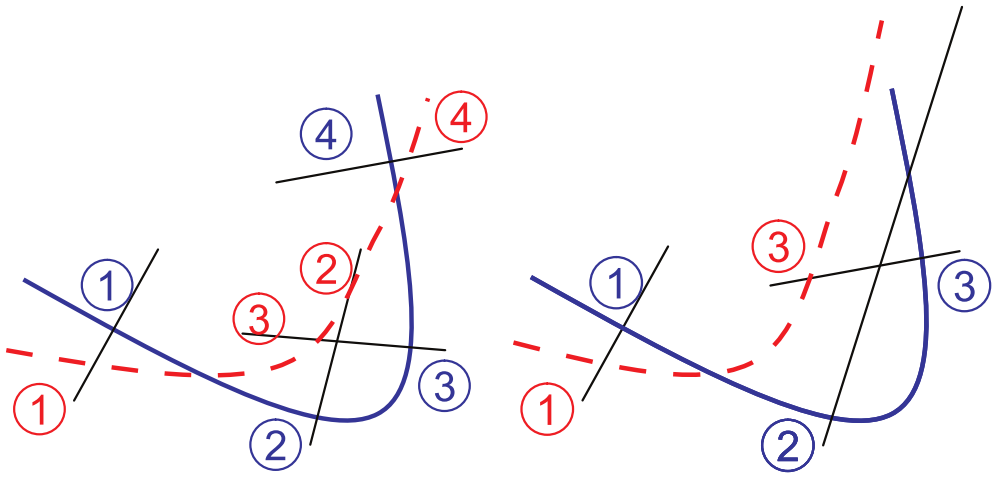


Figure 4.6: These graphics show two problems happening with orthogonal lines. On the left, the order of two sampled points (2-3) is inverted. This can lead to strange temporal profiles. On the right, there is no intersection between the orthogonal line and the trajectory to sample (dashed line). Thus, one sampled points (2) cannot be computed

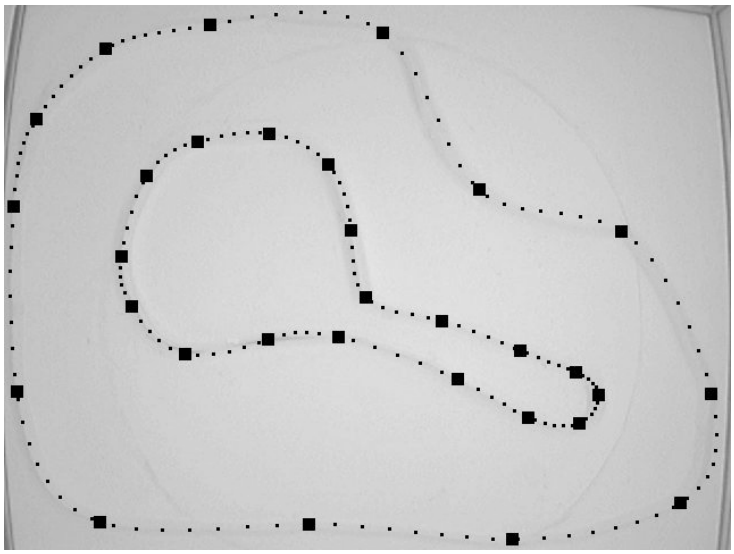


Figure 4.7: Inner and outer walls of the circuit modeled with cubic splines

of the sampled points can be inverted. It leads to strange temporal profiles, where the agent seems to go backward. The second problem is shown in Figure 4.6(b): in some specific cases, there is no intersection between the orthogonal line and the trajectory to sample. In order to solve this problem, one trick is to use a trajectory allowing to sample all the trajectories of the dataset as a reference. However, it sometimes happens that no trajectory in the dataset is suitable to this purpose.

4.1.2.2 Gate sampling

To avoid the problems due to sampling with orthogonal lines, we developed another method, using the circuit walls to create gates between them. If the gate goes from one wall to the other, it will necessarily cross all the trajectories of the dataset. Moreover, if the generated gates are not crossing each other, the order of the sampled points will not be inverted.

The gates are generated in two steps: first, thousands of gates linking the circuit walls are created and second, the most suitable gates are selected. The precise procedure for generating the final gate distribution is as follows.

The inner and outer walls, which are manually extracted from the image of the circuit, are both modeled with multiple cubic splines (Figure 4.7). The number and positions of the cubic splines controll points of the cubic splines were chosen manually to approximate as much as possible the wall curves. In case of simulations, the walls are directly generated with multiple cubic splines. Then, gates are created between these two walls. Figure 4.8 shows one tenth of the gates created for one part of the circuit. Even if it seems intuitive to draw them as they are, generating them automatically is quite complex: the walls are not parallel, and building a gate orthogonal to the two walls is therefore not possible. We could however compute a middle curve between the two walls and make the gates orthogonal to this curve, but it can happen that two successive gates cross each other in a sharp turn, leading, in some cases, to

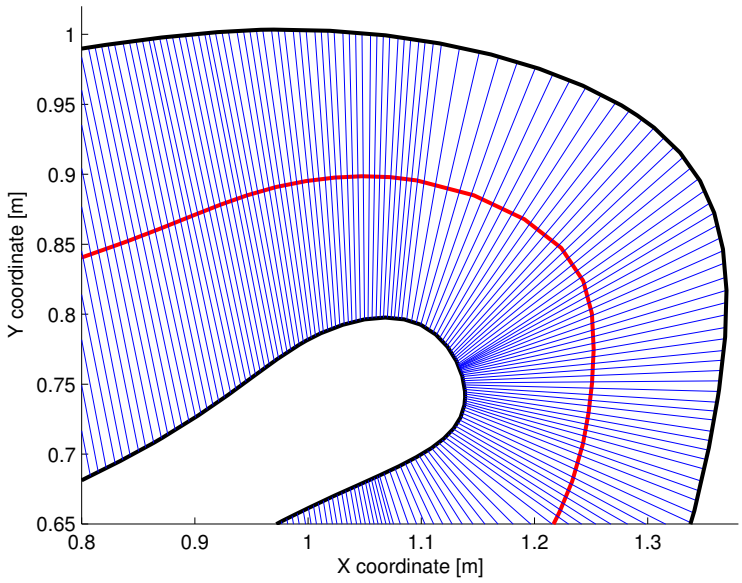


Figure 4.8: Gates created betweem the two walls. Only 10 % of the gates are plotted. We can see the two walls, the gates and the multiple cubic splines approximation of the gate centers

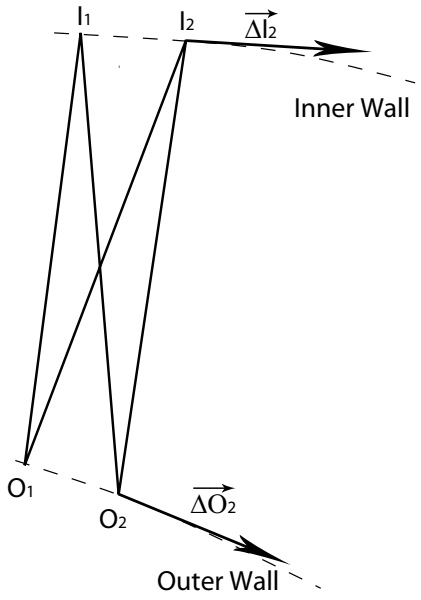


Figure 4.9: If $[I_1O_1]$ is the first gate, the next most suitable gate will be chosen between $[I_1O_2]$,$[I_2O_1]$ and $[I_2O_2]$

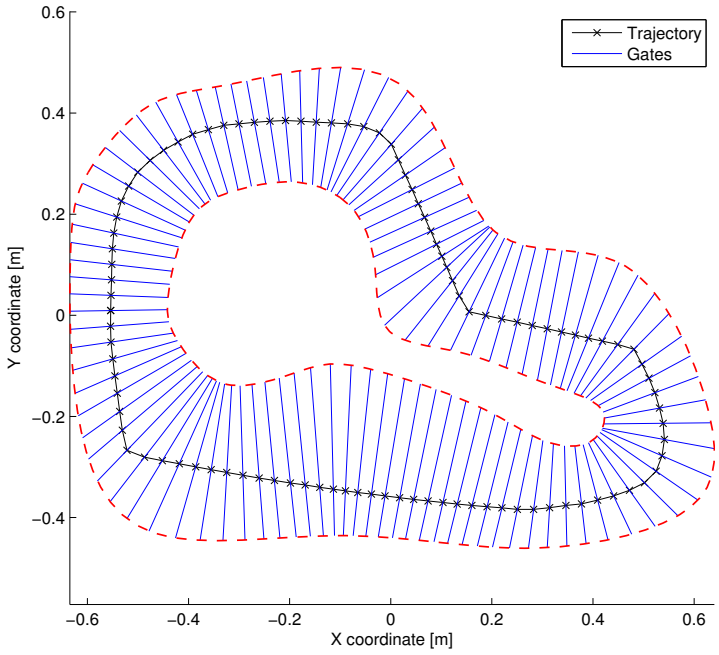


Figure 4.10: Gate sampling: the trajectories are sampled at the intersections with the gates. There are one hundred equidistant gates

an inversion of the order of the sampled points. Thus, we chose a method which generates thousands of points on both walls, using the multiple cubic splines models. The first gate is created between the closest points of the two walls. The next points on both walls are then used to create the following gate. Figure 4.9 illustrates this heuristic process. If the first gate is the segment $[I_1O_1]$, the next possible gates are $[I_1O_2]$, $[I_2O_1]$ and $[I_2O_2]$. We choose the one that gives the largest value to the following condition:

$$\Gamma_{[I_n,O_m]} = \frac{\overrightarrow{I_nO_m} \times \overrightarrow{\Delta I_n}}{\|\overrightarrow{I_nO_m}\| \cdot \|\overrightarrow{\Delta I_n}\|} \cdot \frac{\overrightarrow{I_nO_m} \times \overrightarrow{\Delta O_m}}{\|\overrightarrow{I_nO_m}\| \cdot \|\overrightarrow{\Delta O_m}\|} \cdot \frac{1}{\|\overrightarrow{I_nO_m}\|^{\alpha}}, \tag{4.1}$$

where $\overrightarrow{\Delta I_n}$ stands for the tangential vector to the inner wall at point I_n, and $\overrightarrow{\Delta O_m}$ for the tangential vector to the outer wall at point O_m. As Equation 4.1 shows, three factors are maximized: the orthogonality between the gate and the outer wall, respectively the inner wall, and the inverse of the gate length. The parameter α of this equation have been visually optimized on multiple circuits and a good value was $\alpha = 3$.

As thousands of gates are created, the most interesting ones have to be selected. A selection method has thus been developed: it uses a multiple cubic splines approximation of the curves represented by the gate centers (Figure 4.8). We call this curve $\psi(s) = (x(s), y(s))$ for $s \in [0 \dots S]$, with 0 and S being the initial and final points. This curve can be divided into steps of the same length:

$$Step_{distance} = \frac{1}{n} \int_0^S \sqrt{x'(s)^2 + y'(s)^2} ds, \tag{4.2}$$

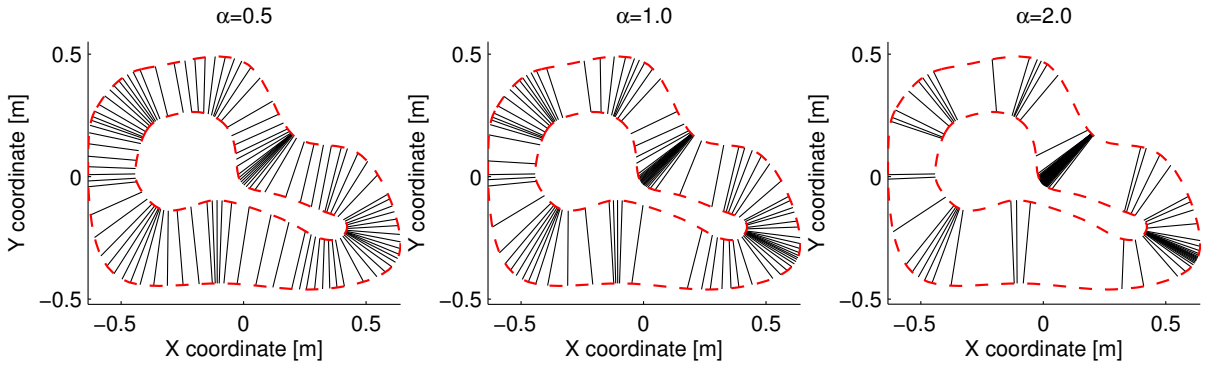


Figure 4.11: Generation of a hundred gates with more gates in the curves (Equation 4.4, for three values of α, see plot for the actual numerical values)

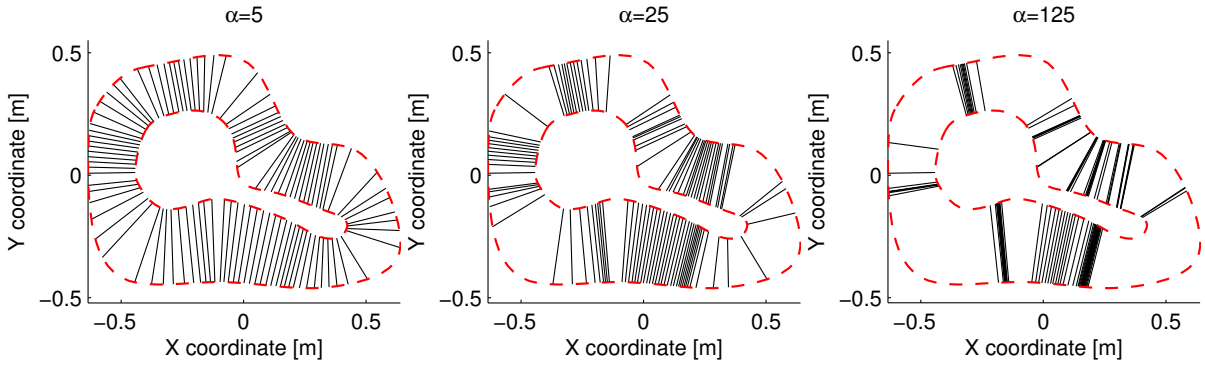


Figure 4.12: Generation of a hundred gates with more gates in the straight parts of the circuit (Equation 4.5, for three values of α, see plot for the actual numerical values)

where n is the desired number of steps (or gates) and $x'(s)$ the first derivative of $x(s)$: $\left(\frac{dx(s)}{ds}\right)$. The curve is thus divided into steps of equal length. The gates that are the closest to each step end are then selected. An example of this method of gate selection is shown in Figure 4.10. Another method is to increase the number of gates in the turns. The sharper is the turn and the higher is the spatial density of the sampling gates in that region. It is based on the curvature $\kappa(s)$ of $\psi(s)$:

$$\kappa(s) = \frac{|x(s)'y(s)'' - x''(s)y'(s)|}{\left(x'(s)^2 + y'(s)^2\right)^{\frac{3}{2}}}, \tag{4.3}$$

$$Step_{curvature} = \frac{1}{n}\int_0^S \sqrt{x'(s)^2 + y'(s)^2}\kappa(s)^\alpha ds, \tag{4.4}$$

where α is a scalar and $x'' = \frac{d^2x(s)}{ds^2}$. Increasing α increases the number of gates in the curves as it is shown in Figure 4.11, and for $\alpha = 0$, this method produces equidistant gates.

The last method was complementary to the previous one and designed to increase the number of gates in the straight parts of the circuit:

$$Step_{straightness} = \frac{1}{n}\int_0^S \sqrt{x'(s)^2 + y'(s)^2}\left(\arg\max_{s\in[0,S]}(\kappa(s)) - \kappa(s)\right)^\alpha ds \tag{4.5}$$

where α is a scalar. Increasing α will increase the number of gates in the straight parts of the circuit, as it is shown in Figure 4.12. It is not possible to use $\frac{1}{\kappa(s)}$ in the Equation 4.5, as $\kappa = 0$ when the circuit is straight, which is often the case.

Once the gates are selected, the trajectories are sampled at their intersections with the gates. In Figure 4.10, the crosses correspond to these intersections. They are computed with a linear interpolation between the two nearest trajectory points on both sides of the gate. For a specific gate, as all sampled points are on the same segment (gate), it is redundant to keep both x and y coordinates, as x can be written as a linear combination of y, and conversely. Thus, only two values are stored: the interpolated temporal coordinate t and the position on the gate g, a value between 0 when located at the intersection with the inner wall and 1 when located at the intersection with the outer wall. As the trajectories are always between the two walls, values lower than 0 or greater than 1 are impossible.

The three methods to select the gates and the corresponding value of α will define the local density of the sampled points. This density must be chosen according to the application and to its objectives. For example, if we are more interested by the agent reactions in the curves, increasing the density of sampled points at these locations is important.

4.2 Light tracking setup

In the semester project of Aïsha Hitz [90], we analyzed the trajectories of a mobile robot moving to a light from a common starting point and with the same initial orientation. Figure 4.13 shows the experimental setup. This environment is different from a circuit because it lacks walls. However, even without walls, there is an underlying structure of the trajectories determined by the lighting template, as they always

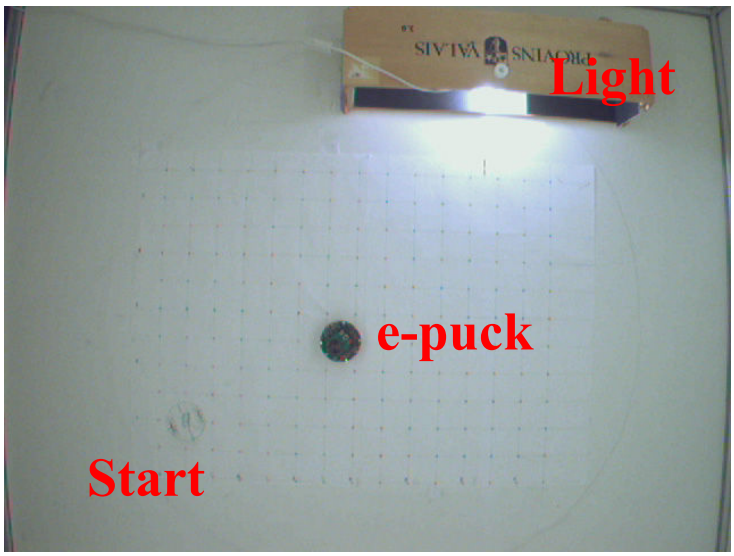


Figure 4.13: Another experimental setup is used: an e-puck mobile robot moving in direction of a light

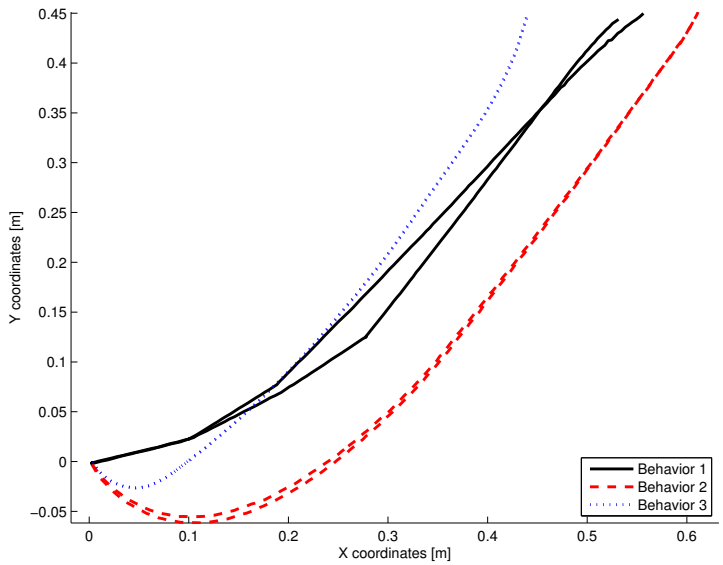


Figure 4.14: Six trajectories generated on the setup presented in Figure 4.13. The e-puck robot has three different behaviors driving it and two trajectories per behavior are plotted. For the third behavior, the two trajectories are so similar, that it is difficult to differentiate them

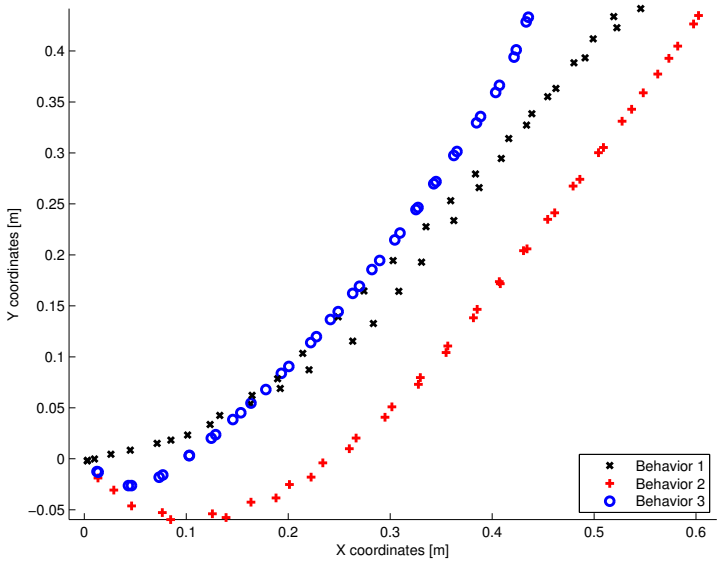


Figure 4.15: Time-based sampling of the trajectories of Figure 4.14. Each trajectory was sampled with twenty points equally separated in time. Problems of correspondence between the sampled points corresponding to the same behavior are clearly observable

have the same starting and ending area. Figure 4.14 shows six trajectories generated using this setup. The robot had three different behaviors while driving to the light, which are clearly identifiable. The initial number of points per trajectory is quite variable, going from 116 to 145. To compare the trajectories, the same number of points per trajectories is needed. Thus, methods similar to those used for circuits were developed, using the trajectory duration or length, parallel lines, or concentric circles.

4.2.1 Time-based sampling

The time-based sampling method used with this setup is exactly the same as that developed for the circuits (see Section 4.1.1). Each trajectory is divided into steps of the same duration. Figure 4.15 shows the trajectories of Figure 4.14 sampled with this time-based technique. The trajectories with the same behaviors makes it clear that the spatial correspondence between the sampled points is very bad, as a sampled point of one trajectory can be located very far from its corresponding sampled point of the other trajectory. For example, the two trajectories of the third behavior have actually very minor spatial differences. However, the time-based sampling method introduced an important spatial difference between the sampled points. To solve this problem, space-based sampling methods need to be developed.

4.2.2 Space-based sampling

Methods close to those used for the circuit have been developed. Gate sampling, as presented in Section 4.1.2.2, cannot be used, as there are no circuit walls to anchor the gates. The other techniques described in Section 4.1.2.1 can be transposed directly to the setup, but would lead to the same problems described

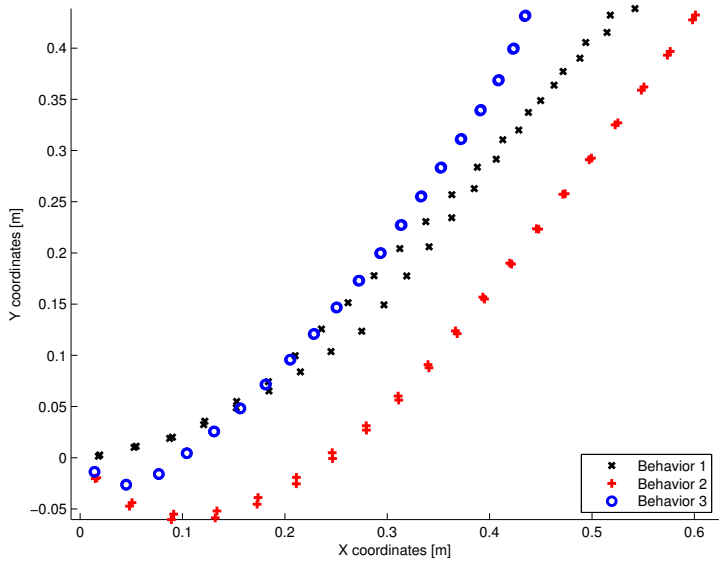


Figure 4.16: Space-based sampling of the trajectories of Figure 4.14 based on the trajectory length. Each trajectory is represented by twenty sampled points. The correspondence between these points is much better than the one achieved with the time-based sampling

previously. Thus, new methods were developed for this experimental setup.

4.2.2.1 Trajectory length

A solution to sample the trajectories is to divide them into parts of the same length. In a formal way, if $f(s) = [x(s)\ y(s)\ t(s)]$ is a function describing the trajectory, with $s \in [0, S]$, the total length of the trajectory is:

$$length_{trajectory} = \int_0^S \sqrt{\frac{dx(s)}{ds}^2 + \frac{dy(s)}{ds}^2}\, ds. \tag{4.6}$$

Thus, for a desired number of sampled points (n), the resulting k^{th} sampled points will be $P_k = [x_k\ y_k\ t_k] = f(s_k)$, where s_k satisfies the condition:

$$\int_0^{s_k} \sqrt{\frac{dx(s)}{ds}^2 + \frac{dy(s)}{ds}^2}\, ds = \frac{(k-0.5)}{n} \cdot \int_0^S \sqrt{\frac{dx(s)}{ds}^2 + \frac{dy(s)}{ds}^2}\, ds, \tag{4.7}$$

where $k \in [1 \ldots n]$. Figure 4.16 shows the sampled points resulting of this technique applied to the trajectories of Figure 4.14. This method, which is very simple and can be applied to absolutely all trajectories, shows less problems of correspondence than the time-based method previously described. There is however a drawback when one of the trajectories is clearly longer (or shorter) than the others on a very small part only. Such an extension (or shortening) changes the length of the trajectory and the positions of its sampled points will thus be different from those of the other trajectories. Such difference could be the result of an obstacle or of a driving error that leads to a complete turn to drive back on

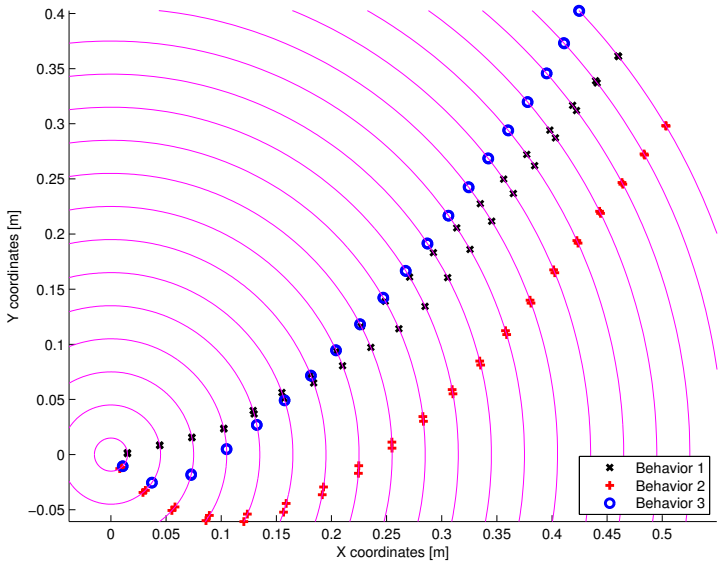


Figure 4.17: Space-based sampling of the trajectories of Figure 4.14 based on circles. Each trajectory is represented by twenty sampled points. These points are the intersection between the trajectory and a set of concentric circles used for the sampling. The correspondence between the sampled points is fairly good

the desired pathway. For this setup, the agent had a smooth movement and thus, there were no such problems.

This spatial method was presented with two spatial dimensions, but it can be easily extended to three dimensions. In the method description, the trajectories are defined as continuous functions, but linear interpolation can be used to apply this technique to trajectories defined by a set of points.

4.2.2.2 Sampling based on circles

Another method, creating virtual gates where the trajectories are sampled, can also be applied. It is much more similar to the gate sampling method used with circuit. As there are no circuit walls anymore, these gates are anchored to the area covered by the experiment. Concentric circles centered on the starting point have been used, as this point is common to all the trajectories. Figure 4.17 shows the twenty circles used for the sampling and the resulting sampled points. To find these sampled points, two successive points are determined, one inside the sampling circle and the other one outside of it. Then, the intersection between the segment described by these two points and the sampling circle is computed. The resulting sampled points have a good spatial correspondence.

The interests of this method lies in the fact that even if a trajectory is very different from the others on a specific part of it, only the sampled points near this part will be moved. This is why this technique is a bit more interesting than the one based on the trajectory length. For this setup, the circles were taken with a regular increase of the radius, as the trajectories were quite orthogonal to them. However,

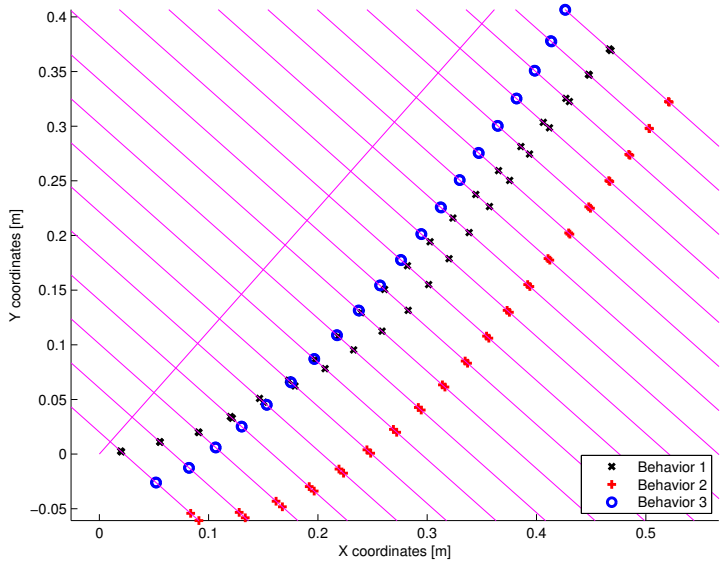


Figure 4.18: Space-based sampling of the trajectories of Figure 4.14 based on parallels lines. Each trajectory is represented by twenty sampled points. These points are the intersection between the trajectory and a set of parallel lines used for the sampling. The correspondence between the sampled points is fairly good

more circles could be used for the sampling at places where the trajectories become less orthogonal to the circles, similarly to the sampling techniques used with circuits, where the number of sampling gates could be increased in some regions. This method can also be used with three spatial dimensions, the sampling circles becoming spheres.

4.2.2.3 Parallel lines

The last method used, which is very similar to the one just presented, is based on a set of parallel lines. A segment as parallel as possible to the trajectories is manually chosen and divided into parts of equal length. Then the lines orthogonal to the segment passing through the subdivision are computed. The sampled points then correspond to the intersections of the trajectories with these lines. Figure 4.18 shows the example of trajectories sampled with this technique. The advantages and drawbacks are quite similar to those of the circle-based method. On this setup it is however as a bit less efficient than the circle-based technique given that at the starting points of the trajectories the gates are less orthogonal to the sampling lines. Thus, there is much more distance between two sampled points, which do not represent well the first part of the trajectories generated with the second behavior. This method can be used with three spatial dimensions, the lines becoming planes.

4.3 Conclusion

This chapter has presented two trajectory generation setups. Even if only two case studies have been examined, the methods used can be easily extended to a large panel of generated trajectories. Other applications will be shown in the next chapters.

Time-based sampling techniques offer solutions easy to implement. However, the resulting sampled points are most of the time difficult to compare directly, as speed variations have a major influence on the location of the sampled points. As a result, space-based sampling techniques are presented. The most interesting solutions define barriers in the environment. The trajectories are then sampled at their intersections with these barriers. However, these solutions have an important drawback. If the agent goes back and forth through a barrier, an important amount of information gets lost during the sampling, as only the first intersection with a barrier is kept.

This chapter has also demonstrated that sampling is a crucial part of the analysis and that the mix of spatial and temporal data is complicated to handle. If it is not performed correctly, the sampled points loose their spatial correspondence and the resulting analysis will fail to provide useful information. In addition, having too many sampled points will increase data size and leads to an increase of the computation requirements. On the contrary, information is lost when too few sampled points are considered. As a result, the application and the a priori knowledge of the trajectories must be taken into account when choosing a sampling method and tuning its parameters. If an equivalent of the Nyquist theorem for spatiotemporal data could be found, it would allow for the exact determination of the number of sampled points necessary for the analysis.

Chapter 5

Trajectory analysis

In scientific literature, the comparison of trajectories is mainly qualitative and performed visually. A qualitative comparison is made from the plot of the trajectories and is considered as a sufficient measure. The lack of standard methods to compare trajectories can explain this common practice. However, using a quantitative measure of the trajectory differences would allow for an in-depth analysis eventually leading to scientifically better supported conclusions. Therefore, this chapter presents methods to follow a more systematic an quantitative approach to trajectory comparisons. Still, the comparison of two trajectory remains a difficult task. A trajectory is a complex mix of spatial and temporal data and thus, it is difficult to find a single value to represent the differences between trajectories. In the following sections, multiple methods are presented, using spatial, temporal or spatiotemporal data.

The different methods will be separated between the three types of trajectories presented in the introduction. Quite complex methods will be presented, but it is still important to remember that simple solutions are sufficient in most situations. In some cases, the difference between the trajectory sets will be so large, that nearly all the proposed methods will be able to differentiate them. However, if a method fails to see the difference between two trajectory sets, it does not mean that they are equal. Depending on our application, more complex methods and more trajectory data (spatial or temporal data, their derivatives, curvature, etc.) need to be used to insure that a sufficient number of features are taken into account for the comparison.

5.1 Trajectories with common frame and without drift

The trajectories analyzed in this case come from an agent that is guided by an external element, for instance, a car on a road, an autonomous robot using a GPS-based navigation system, a pedestrian walking down a corridor,... In theses cases, the difference between the trajectories is bounded by this external element. For example, the trajectory of a car on a road is bounded by the road dimension.

To compare this kind of trajectories, multiple data can be used:

- Area between two trajectories
- Trajectory point distributions

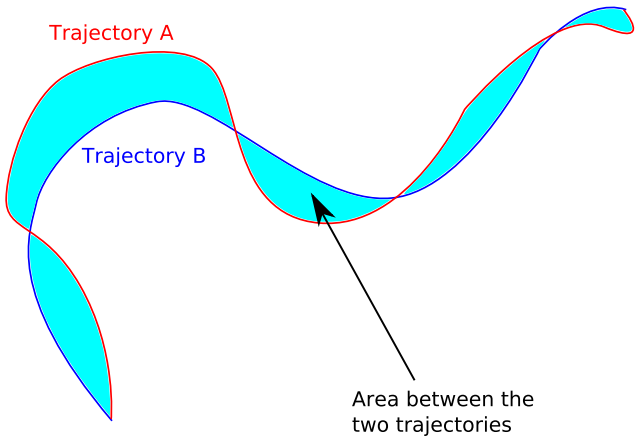


Figure 5.1: This example shows two trajectories and the respective area between them. This area can be used to measure the difference between the trajectories

- Position of sampled points

5.1.1 Area between two trajectories

If we consider a trajectory as a spatial function, the area between the two functions can be used as a measure of their difference. Figure 5.1 shows an example with two trajectories and the area between them.

If $f(t)$ and $g(t)$ are the two functions defining the trajectory, this area A is defined as:

$$A = \int_{t_0}^{t_1} ||f(t) - g(t)||dt \tag{5.1}$$

This integral may be difficult to compute and it can also be difficult to define the trajectories as functions. However, a numerical approximation of this area can be easily computed.

As an example, this method can be applied to the data presented at the *International Conference on Intelligent Robots and Systems* (IROS, Chapter 7.1.1). In Figure 5.2, three trajectories are shown. The first trajectory is the reference, the area difference with the second trajectory is 0.011 m^2 and the area difference with the third trajectory is 0.112 m^2. As a further example, we computed all the area between the reference trajectory of Figure 5.2 and all the trajectories generated during the experiments for the two behaviors used. There are four datasets for the first behavior (1a,1b,1c,1d) and four datasets for the second behavior (2a,2b,2c,2d). Figure 5.3 shows the resulting box plots (Appendix sct.boxplot). This plot shows that the trajectories of the second behavior have a greater area separating them from the reference trajectory than the trajectories of the first behavior. As the reference trajectory was generated with the first behavior, it is the intended result.

This metric is simple and gives a good hint of the difference between two trajectories. However, using it to measure the difference between two trajectory sets is not possible, as an area can only be

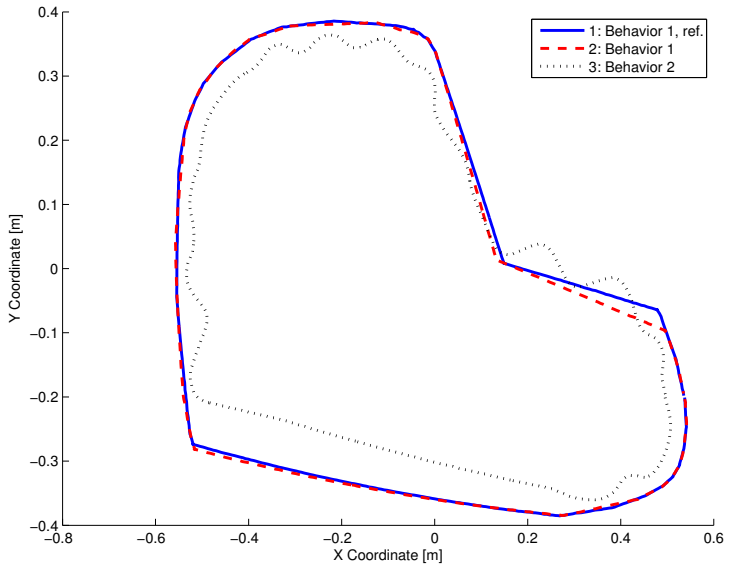


Figure 5.2: 3 trajectories provided as examples. The first two are generated with the same behavior and the third is generated with another behavior. There is 0.011 m^2 between the first (solid blue) and the second trajectories (dashed red), and 0.112 m^2 between the first (solid blue) and the third trajectory (dotted black)

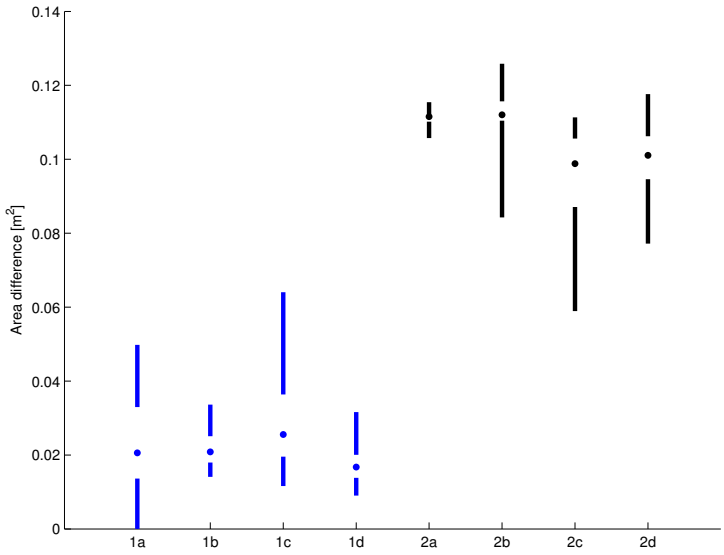


Figure 5.3: The area between the reference trajectory of Figure 5.2 and 4 sets of trajectories generated with each behavior: $1a \ldots 1d$ for the first behavior and $2a \ldots 2d$ for the second one

computed between two trajectories. Moreover, the measure does not give us an indication about the regions where the major difference between the trajectories lies. Finally, computing this value for 2D trajectories is quite easy, but doing it in 3D becomes really tricky.

5.1.2 Analysis of the trajectories as point distributions

The spatial coordinate distributions of trajectory points can be used to compare two trajectories. For example, Nehmzow used the Mann-Whitney U test to compare two trajectories in his book [82].

5.1.2.1 Coordinates analyzed independently

The first way to compare trajectories will be to compare their distributions on each axis independently. We used four statistical tests to measure the difference between the distributions: the Student's t-test, the Mann-Whitney U test, the F-test, and finally the Kolmogorov-Smirnov test. All these tests are presented in Appendix A. The three tests have a null hypothesis (H_0) that the distributions are similar. Rejecting the null hypothesis means that there is a statistical difference between the two trajectories. For the 3 trajectories presented in Figure 5.2, Table 5.1 shows the results of the comparison of the x coordinates using the four statistical tests. It results that the t-test and the U test are not really able to separate the two behaviors. Only the Kolmogorov-Smirnov test is able to classify the controllers. The F-test shows however large difference of P-Value.

Table 5.1: Result of the Student's t-test, the Mann-Whitney U test, the F-test and the Kolomogorov-Smirnov test, when comparing the distribution of the x coordinates of the trajectories presented in Figure 5.2. *Acc.* stands for that the null hypothesis, H_0, being accepted. For all the tests, it means that no statistical difference of the tested characteristic (mean, variance, or cumulative distribution function) can be observed, at a significance level of 95%. *Rej.* stands for the rejection of this null hypothesis

	t-test		U test		F-test		K-S test	
Trajectories	H_0	P-val.	H_0	P-val.	H_0	P-val.	H_0	P-val.
T1 and T1	Acc.	1	Acc.	1	Acc.	1	Acc.	1
T1 and T2	Acc.	0.86	Acc.	0.83	Acc.	0.94	Acc.	0.97
T1 and T3	Acc.	0.90	Acc.	0.51	Acc.	0.06	Rej.	~ 0
T2 and T3	Acc.	0.74	Acc.	0.39	Acc.	0.07	Rej.	~ 0

Figure 5.4 shows the resulting P-value when comparing the reference trajectory shown in Figure 5.2 with four sets of trajectories generated with both behaviors. The results of the t-test and the U test show that these tests have a poor classification power. Only the F-test and the K-S test seem to provide better results.

When performing this kind of analysis, the main problem is that the different coordinates are treated independently. However, the spatial coordinates are really not independent: most of the time, knowing the x coordinate of the robot will clearly give a hint about the y position of the robot. Figure 5.5 shows

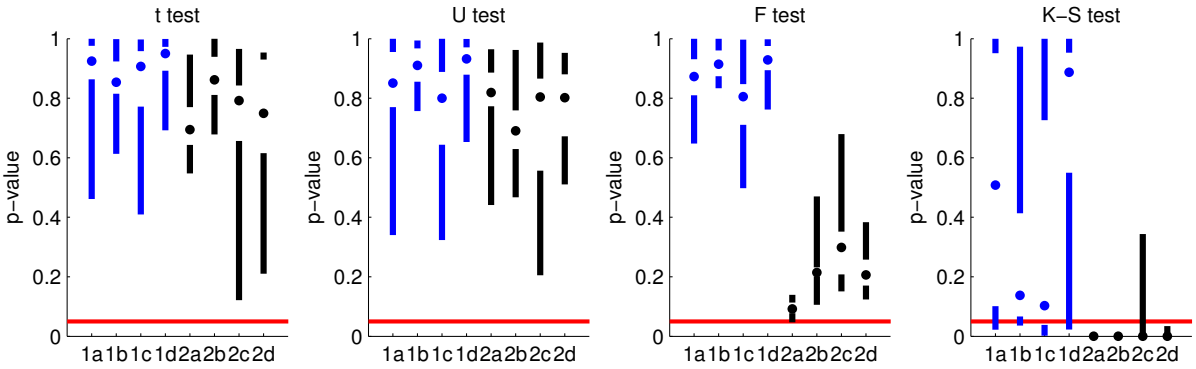


Figure 5.4: Box plot of the resulting P-value of the t-test, the U test, the F-test and the K-S test, when comparing the reference trajectory of Figure 5.2 and 4 sets of trajectories generated with both behaviors: $1a \ldots 1d$ for the first behavior and $2a \ldots 2d$ for the second one

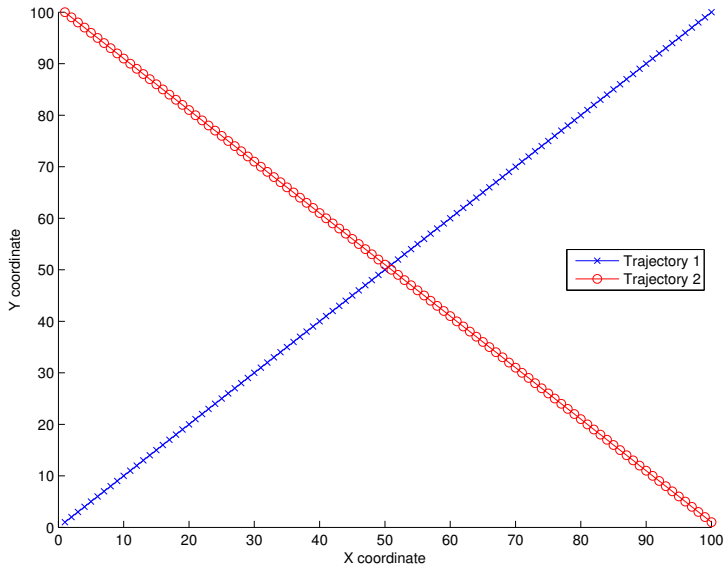


Figure 5.5: Two linear trajectories that are clearly different. However, with an independent analysis of the trajectory coordinates, the t-test, the U test and the K-S test will fail to see a difference between the two trajectories

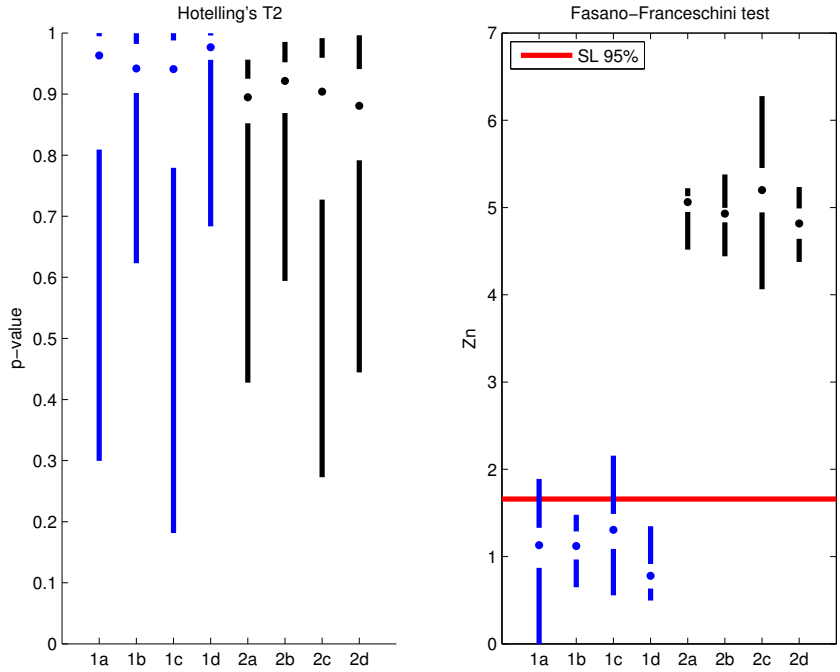


Figure 5.6: Box plot of the results of the Hotelling's T^2 and the Fasano-Franceschini test, when comparing the reference trajectory of Figure 5.2 and 4 sets of trajectories generated with each behavior. For the Fasano and Franceschini test, the plot shows the resulting values of the test. The critical value (1.66) for a significance level of 95% is also plotted as an information

an example of a trajectory difference that cannot be analyzed this way. If one of these univariate test is applied to the trajectory coordinates, all test will fail to see a difference: the hypothesis that there is a difference is rejected with a P-value of 1.

5.1.2.2 Multivariate analysis of the trajectories

A way to bypass the problem presented just before, is to analyze the distributions of the trajectory coordinates together (Multivariate analysis). The multivariate extension of Student's t-test, Hotelling's T^2 test, or an extension of the Kolmogorov-Smirnov test to multiple dimensions, such as Fasano and Franceschini algorithm, presented in Section A.3, can be applied to the trajectory points. However, with the trajectories presented in Figure 5.5 the Hotelling's T^2 fails to see a difference between the trajectories with a p-value of 1. As it compares the distribution means, there is no difference to observe. On the other hand, the Fasano-Franceschini test easily spots the difference, with $Z_n = 3.53$ when the critical value of Z_n for a significance level of 95% is 1.32.

Figure 5.6 shows the results of the Hotelling's T^2 and the Fasano-Franceschini test, when comparing the reference trajectory of Figure 5.2 with four sets of trajectories generated with both behaviors. As the Hotelling's T^2 compares only the distribution means, it does not show really good results in the separation of the two behaviors. However, as the Fasano-Franceschini test compares the shapes of the

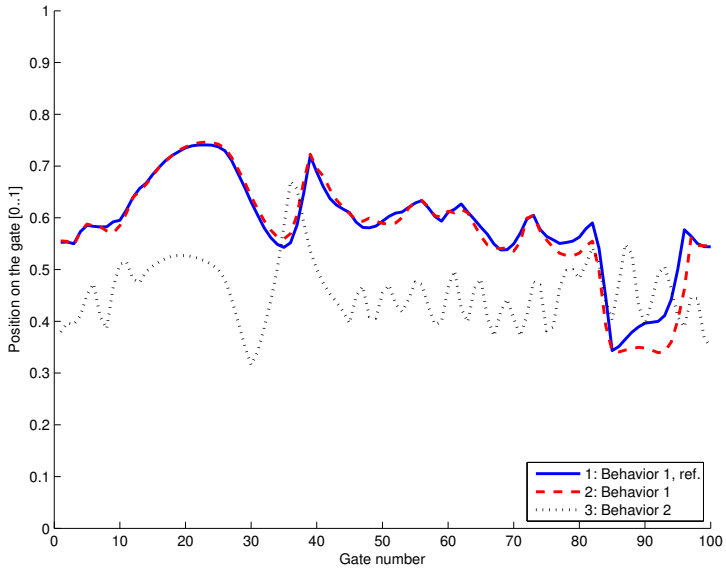


Figure 5.7: Position on the sampling gates of the 3 trajectories plotted in Figure 5.2. We can clearly see the spatial differences between the two behaviors

distributions, it is really efficient to separate the two behaviors. Thus, it is a good solution to compare trajectories using only the distribution of their points. As a counterpart, it requires quite a lot of computation. Moreover, as only the distributions of the points are analyzed, the point order have no influence on the results, even if this order is quite an important characteristic of a trajectory, as it describes the direction of motion.

5.1.3 Trajectory analysis using sampled points

Another intuitive way would be to compare the trajectories in specific regions. It is quite similar to the way people analyze the trajectories: “At the beginning, the man was on the right side of the street. When he arrived at the bridge, he crossed it.” In this example, the beginning and the bridge are locations where the trajectories have to be compared. The major problem is to find the correct regions where we want to compare the trajectories. It is certain that this selection is really dependent on the goals of the analysis.

Chapter 4 has presented multiple methods to sample the trajectories. The goal is to have the same number of points for all the trajectories to compare their ordered sets of points. Thus the i^{th} point π_i^k of the k^{th} trajectory can be compared with the i^{th} point π_i^l of the l^{th} trajectory.

To show an example of analysis, we can use the IROS data presented in Section 5.1.1. The data are sampled with 100 equidistant gates, as introduced in Section 4.1.2.2. Similar results can be obtained using other sampling methods. Figure 5.7 shows the spatial profiles of the three trajectories plotted in Figure 5.2 and Figure 5.8 shows their temporal profiles.

A measure of the difference between two trajectories similar to the area between the trajectories is the sum of the absolute difference between the positions on the gates. If π_i^k and π_i^l are the positions on

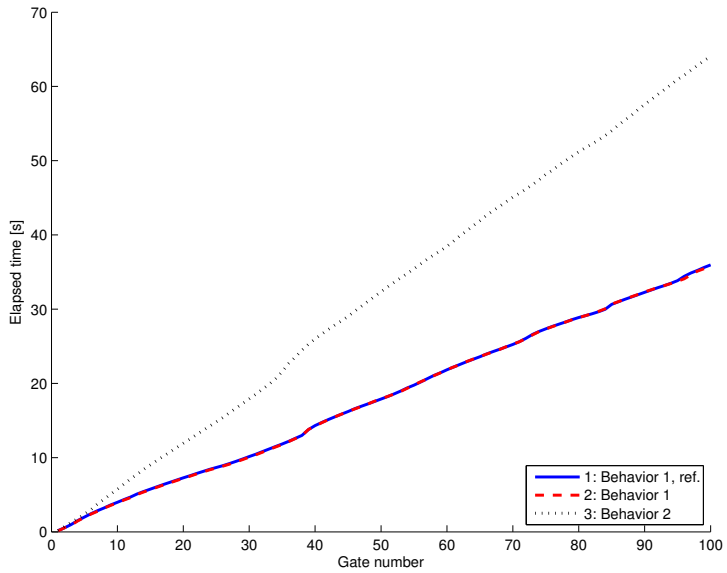


Figure 5.8: Temporal profile of the three trajectories plotted in Figure 5.2. Another difference between the two behaviors can be observed

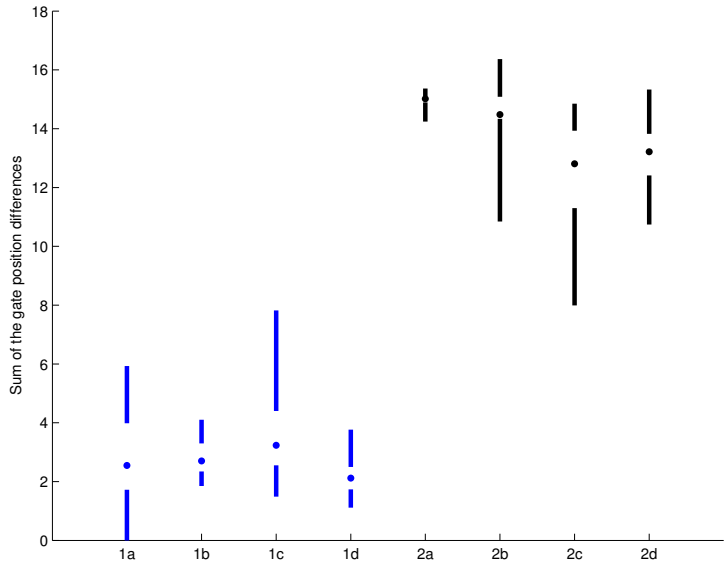


Figure 5.9: Box plot of the sum of the difference between the gate positions of the reference trajectory of Figure 5.2 and 4 sets of trajectories generated with each behavior

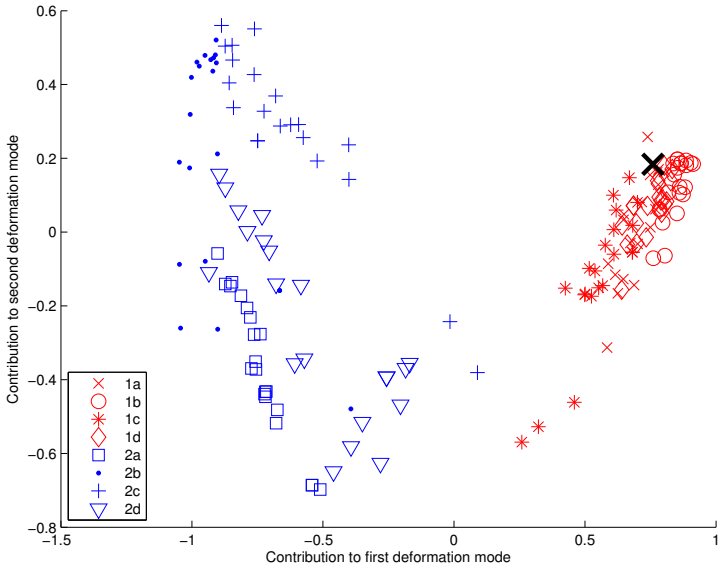


Figure 5.10: PDM analysis of four sets of trajectories generated by the two behaviors presented in Figure 5.2 (the black X corresponds to the reference trajectory). There are 20 trajectories per set for a total of 160 trajectories. The PCA was done using the whole dataset. A clear separation of the two behaviors is observable

the i^{th} gate of the k^{th} and l^{th} trajectory respectively, then the difference D is:

$$D(k,l) = \sum_{i=1}^{N} |\pi_i^k - \pi_i^l| = D(l,k) \tag{5.2}$$

where N is the number of sampled gates. For the trajectory shown in Figure 5.2, this difference is 1.49 between trajectory 1 and 2, and is 15.1 between trajectory 1 and 3. Figure 5.9 looks similar to Figure 5.3. The measure presented in Equation 5.2 is computed for the same 4 sets of trajectories for both behaviors. All the measures of difference are relative to the first trajectory of Figure 5.2. The results are approximately the same, except for a scale factor of ~ 100. It is not really astonishing as we are more or less computing an approximated integral of the distance between the trajectories.

5.1.4 Trajectory analysis using Point Distribution Models

As presented in Chapter 3, a Point Distribution Model (PDM) can be used to compare trajectories. As an example, the same trajectories as in the previous sections can be used. Figure 5.10 shows the resulting contributions to the first two deformation modes of the PDM. Each behavior was represented by 4 sets of 20 trajectories, for a total of 160 trajectories. The results show that there is a clear separation in the two dimensions of maximal variance. Thus, as it has already been stated with other statistical tools before, there is a noticeable difference between the two behaviors. 67% of the cumulative energy lies in the first dimension and 77% in the first two dimensions.

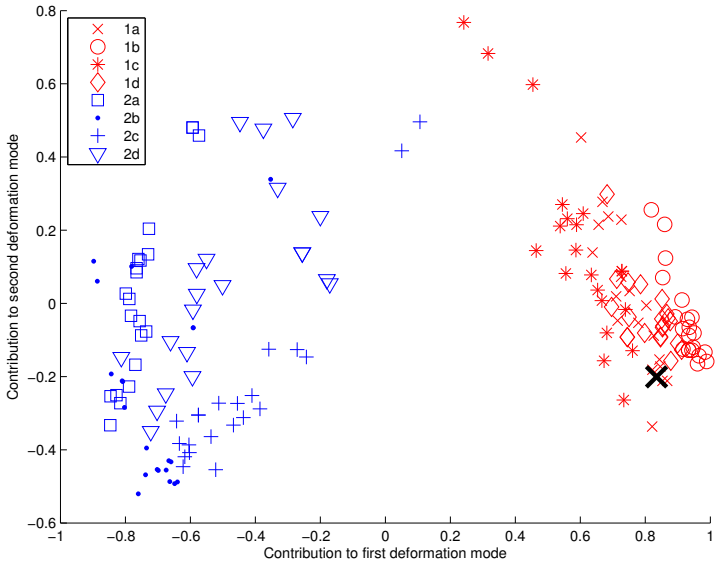


Figure 5.11: PDM analysis done with the same trajectories used for Figure 5.10. The PCA was computed using only the first set of each behavior (2x20 trajectories). Then the corresponding contributions to the modes where computed for the other trajectory sets. Compared to Figure 5.10, the PCA analysis is really similar and thus stable

To show the stability of the PDM, the PCA was computed with only the first set of each behavior. The contribution to the first two deformation modes were then computed for the other trajectory sets using the resulting eigenvectors. Figure 5.11 shows the resulting deformation modes. Except for an inversion of the second axis (PCA searches for the axes of maximal variance and thus the direction is not determined), the resulting contributions are quite the same as in Figure 5.10. This method is thus not too dependent of the input dataset.

5.1.4.1 Linear discriminant analysis

It is possible to replace the Principal Components Analysis by a Linear Discriminant Analysis (Section A.6.2) in the Point Distribution Model. The resulting model will thus focus further on the separation of the behaviors instead of their variance. Figure 5.12 shows the resulting contribution to the first two LDA dimensions. A better separation is achieved using a LDA compared to a PCA.

However, if the LDA is computed with only the first set of each behavior, the resulting separation is much more unstable. Figure 5.13 shows the resulting contributions of the trajectories coming from all eight sets, when computing the LDA with only the first set of each behavior. The separation for these specific two sets is nearly perfect, as each set of 20 trajectories is reduced to one visible point. However, the other six sets are not so well separated. This figure shows how much the LDA is sensitive to the input dataset: it will find the best axis to separate the two trajectory sets, but not always the best axis separating the two behaviors.

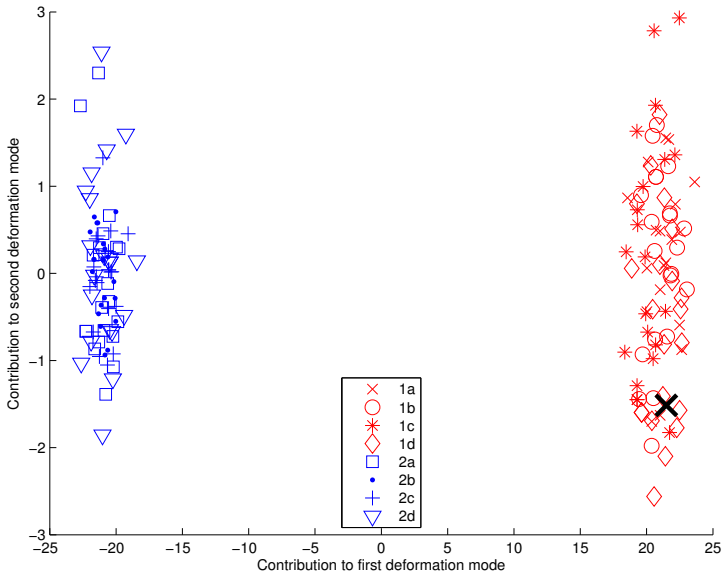


Figure 5.12: LDA applied to the same dataset used in Figure 5.10. The separation achieved with this method is much better than the one obtained with a PCA

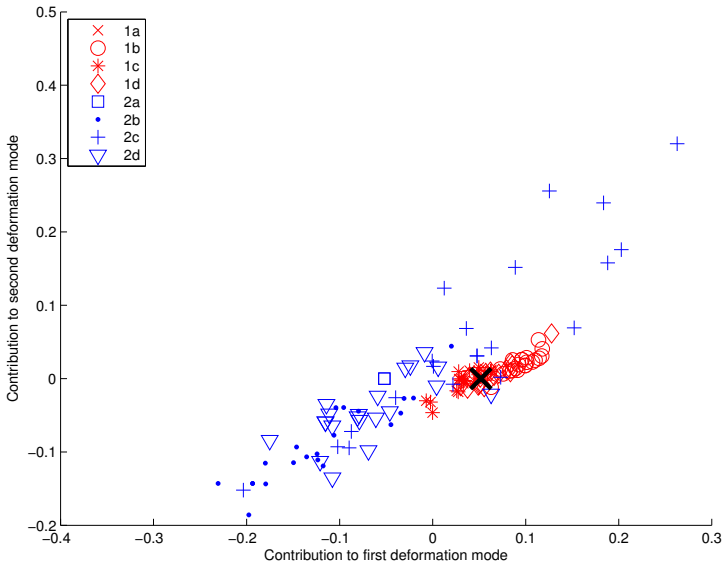


Figure 5.13: LDA analysis applied to the same trajectories used in Figure 5.12. The LDA was computed using only the first set of each behavior (2x20 trajectories). Then the corresponding contributions to the modes where computed for the other sets of trajectories. The results show how much the LDA is sensible to the original dataset and how unstable it can be

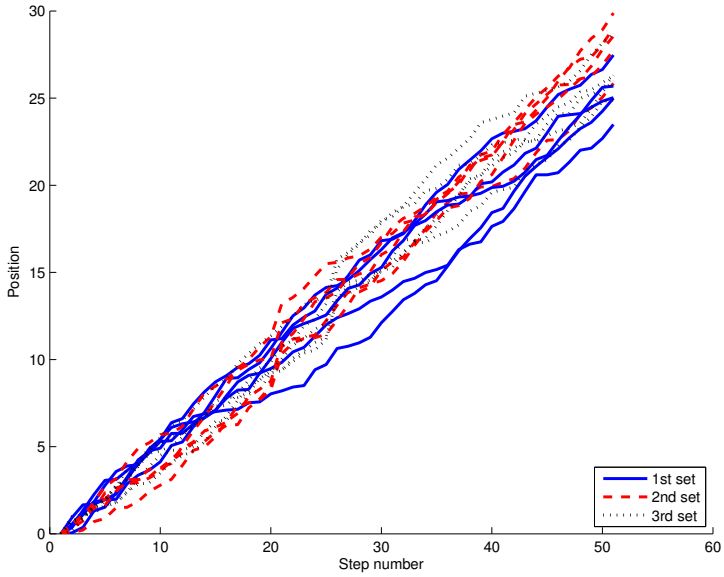


Figure 5.14: Example of profiles of three different behaviors. The difference is not easy to point out

5.2 Trajectories with common frame and with a drift

In the case were the agent is not guided continuously based on an absolute reference, the positioning error is added on the whole trajectory. If all the trajectories are starting at the same position, the difference between the trajectory points will increase along the trajectories to be maximal at their end. If the comparison of the trajectories is based on the maximal difference, it will ignore nearly completely the first part of the trajectories to focus on their last part. To maintain a good comparison among trajectories, a PDM analysis cannot be done as before.

Trajectories of a mobile robot using only on-board, egocentric navigation capabilities (e.g. odometry) is one example of scenario where such kind of trajectory can be generated. However, a more common example happens when temporal profiles are compared. If one agent is faster than the other, like in Figure 5.8, this difference of speed is added on the whole trajectories and the difference at the end will be extremely large. As a consequence, the PDM will mainly focus on the last part of the trajectories.

Figure 5.14 shows a completely fictional example. Three kinds of profiles were created. They can correspond to the temporal profile of a mobile robot driven by three different controllers. These controllers are quite noisy and thus are difficult to differentiate. However, for the second and the third sets, the controller is really losing time between two sampled points. The second dataset has a delay at the 20^{th} step when the third is slowing at the 25^{th} step. These delays are quite difficult to observe in the plot, as they are completely drowned out by the trajectory noise.

Theses profiles can be analyzed with a PDM. Figure 5.15 shows the location of the different profiles in the space of the first two modes of the PDM. The difference between the three dataset is also difficult to spot in the space of the PDM.

To reduce the influence of the error accumulation on the whole trajectory, the derivate of the profiles

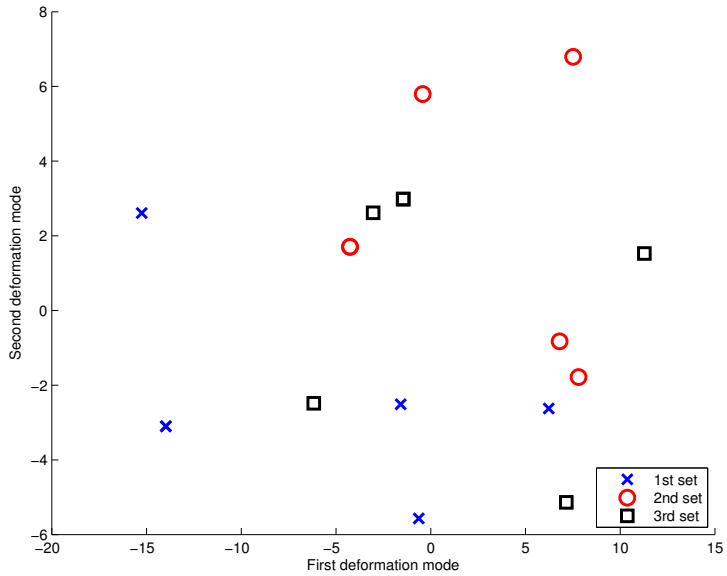


Figure 5.15: PDM analysis of the profiles plotted in Figure 5.14. The difference between the three datasets is difficult to spot

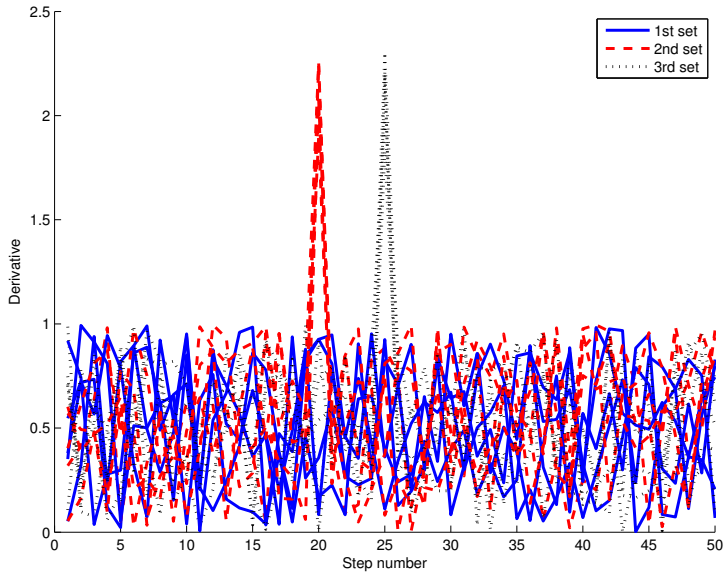


Figure 5.16: Derivative of the profiles shows in Figure 5.14. The singularities of the second and third dataset are clearly visible in this plot

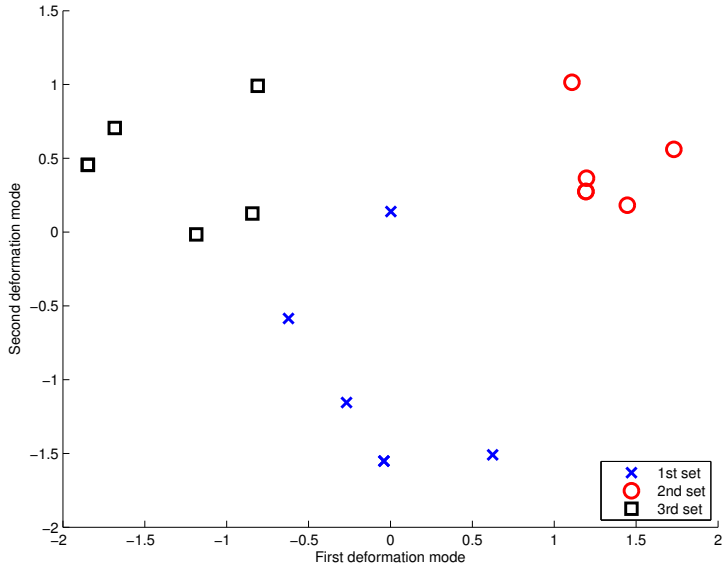


Figure 5.17: PDM analysis of the profile derivatives. Contrarily to the previous PDM analysis, the difference between the dataset is pointed out in this plot. The three datasets are thus clearly separated

could be used. Figure 5.16 shows the derivatives of the profiles introduced previously. The delays added to the second and third datasets are clearly visible. Likewise, in the PDM analysis, these differences will pop up. Figure 5.17 shows the location of the profile derivatives in the space of the PDM. The three datasets are clearly separated.

The objective of this example is to outline the difference between profiles where the variations are cumulated or not along the trajectories. In the previous sections, the presented trajectories were corrected by the environment and thus had no cumulated errors. Thus, a direct comparison of the profiles was possible. However, when trajectories with drift are handled, it is more efficient to work on the profile derivatives.

5.3 Trajectories without common frame

Trajectories without common frame are the most unusual. In this particular case, the agent has no pre-established path (i.e. plan) to move from one location to another, but is just roaming around and reacting to the environment. This kind of movement corresponds to a mobile robot avoiding obstacles or to the trajectory of an undirected billiard ball. Some randomized search strategies can also be considered in this category.

Figure 5.18 shows the trajectories of 4 e-puck mobile robots moving in a square arena of 0.72x0.72 meters. Two different behaviors were encoded to drive the robots. A pair of robot (1-2) was driven by the first controller and the other pair (3-4) was driven by the second one. Both behaviors were avoiding the arena walls and the others robots. They possess similar straight lines, but produces different curves, more or less sharp, depending on the controller. The resulting movements are examples of trajectories

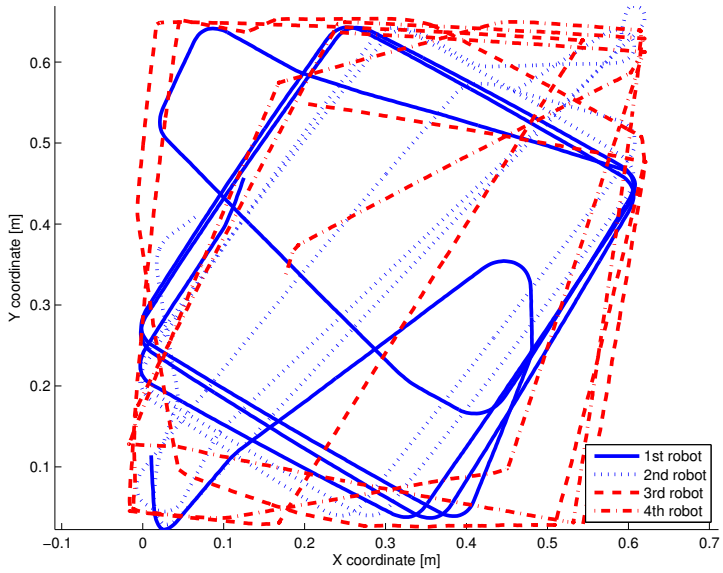


Figure 5.18: Trajectories of 4 mobile-robots avoiding each other and the walls of a square arena (0.72x0.72 meters). A pair of robot (1-2) was driven by one controller and the other pair was driven by another controller (3-4). From the trajectories, one controller produces sharper turns and the other more smoother ones

without common frame.

To compare the different trajectories, the techniques described previously cannot be used. In this case, the Correlated Random Walk (CRW) model presented in Section A.9.3 can be used to represent the trajectories.

5.3.1 First method to extract the CRW distributions

To create a CRW model, the distribution of step lengths and angles between the step must be extracted from the trajectory points. The first method used considers each segment between two successive trajectory points as a step. If there is a constant time interval between two trajectory points (like with vision-based tracking systems), the step length can be directly related to the agent speed. Figure 5.19 describes how this method works.

In our case, this method was working quite well, excepting some localization errors caused by the tracking system. Figure 5.20 shows the problem: a small error of agent localization results in a large error in the angle between two steps. As a result, a solution was implemented to remove this tracking noise. When the distance between two successive points is under a given threshold, a zero is stored for the step length and the angle between the steps. This solution is logical as a robot moving so slowly can be considered as motionless. Thus, small localization errors will no more ends in huge errors for the angle distribution.

Figure 5.21 shows the distribution of step lengths and step angles analyzed together. As the two

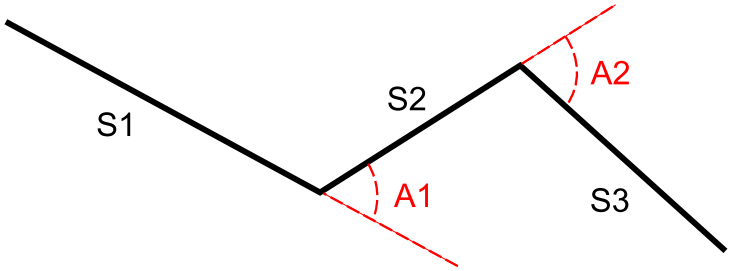
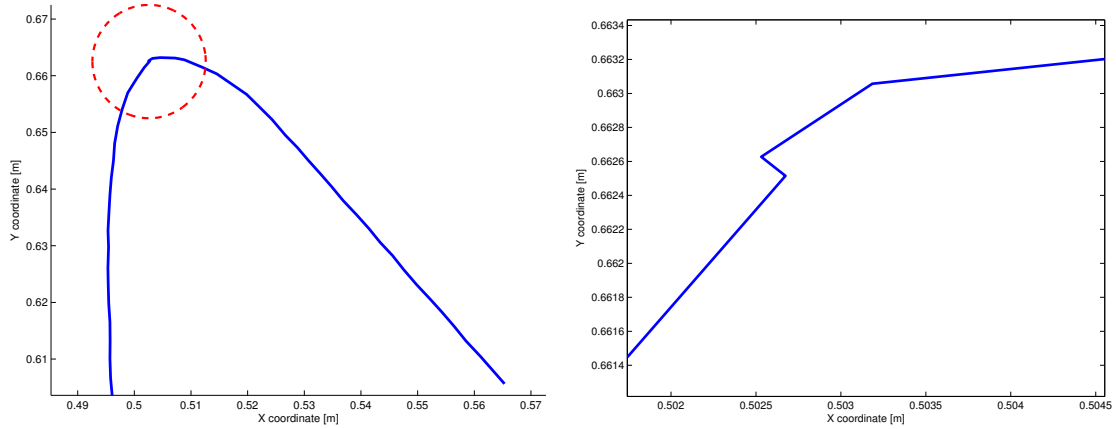


Figure 5.19: First method to extract CRW distributions. Each segment (S_i) between two successive trajectory points is considered as a step. The length of these steps and the angles between them (A_i) are stored



(a) Tracking error (b) Close up

Figure 5.20: Example showing the influence of a tracking error. When the agent is not moving, or moving slowly, the approximation of the angle between the steps can be really influenced by a tracking error. The right plot is a zoom of the area circled in the left plot

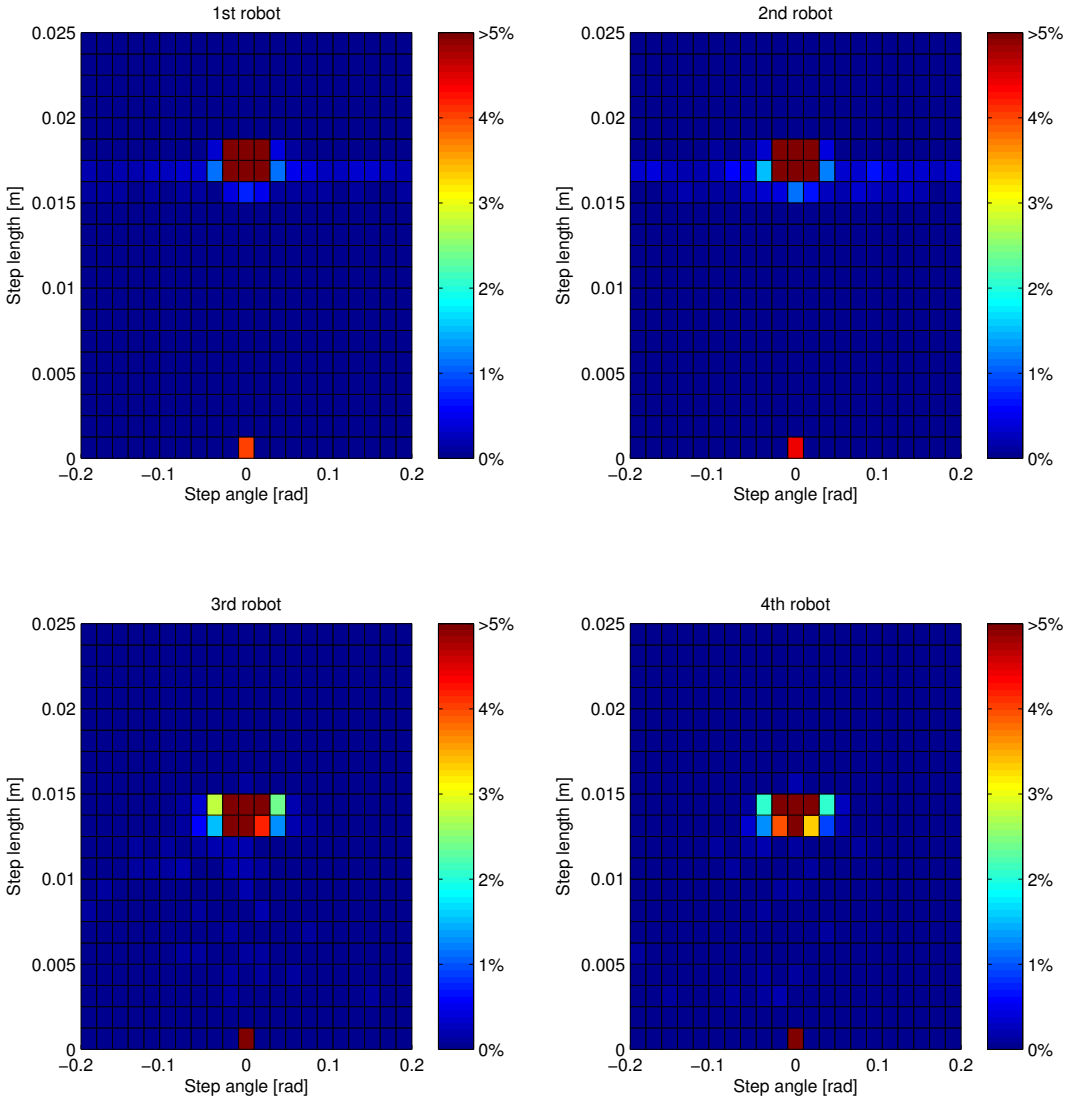


Figure 5.21: Bidimensional histogram of the step lengths and step angles, for the four trajectories shows in Figure 5.18. The number of occurrence per subdivision was divided by the total number of occurrence. Thus, each subdivision presents a proportion of occurrence, similar to a PDF function. The color axis corresponds to this proportion of occurrence

Table 5.2: Maximal difference of cumulative distribution function (CDF) between the four histograms shown in Figure 5.21. All results are in percentage

	1^{st} set	2^{nd} set	3^{rd} set	4^{th} set
1^{st} set	0	4.6	95.4	95.3
2^{nd} set	4.6	0	94.6	94.5
3^{rd} set	95.4	94.6	0	22.8
4^{th} set	95.3	94.5	22.8	0

distributions are correlated, it is important to analyze them together. In these two plots, the difference between the two controllers can be spotted. For both controllers, majority of steps are straight (i.e. the distribution as a peak around $angle = 0$), whose length can be related to the controller speed. It can be observed that the first two robots are faster, as their average step length is a bit longer.

As the method proposed before was used to remove the tracking errors, all steps smaller than 2.5 mm were put to a length of zero, similar to their respective angles. Thus, for the four robots, there is quite a large fraction of steps in the bin $[0, 0]$. This fraction is higher for the last two robots. As they are turning on the spot, they will produce additional steps below the threshold mentioned above.

The difference between the histograms can be measured in a similar way as the Fasano-Franceschini extension to two dimensions of the Kolmogorov-Smirnov test presented in Section A.3.3. Table 5.2 shows the maximal difference of CDF measured for the four histograms of Figure 5.21. The two sets of controller are clearly visible.

5.3.2 Second method to extract CRW distributions

Another method was tested to extract the CRW distributions. Figure 5.22 illustrates it. The longest step between two trajectory points is chosen, as long as the distance to this segment from the points in-between is smaller than a chosen error. The resulting trajectory corresponds to a polyline approximation. This method removes the noise produced by the tracking. However, the length of the steps can no longer be related to the agent speed, but rather to the environmental features.

Figure 5.23 shows the distributions extracted from the trajectories of the same four robots, using the second method. There is also a big difference between the histograms of both controllers. The first two robot produces a lot of small turning segments. As their movement has a lot of curves, these curves are approximated by a lot of steps. On the contrary, as the last two robots are turning on the spot, they do not produce these small steps.

The maximal step length can be related to the arena. In a smaller arena, the steps will be smaller, and in a larger arena, they will be longer, if the number of robots is identical. This kind of hints could not have been observed with the first method of distribution extraction.

For the histograms of Figure 5.23, the measures of difference are shown in Table 5.3. The two sets of controller are also visible, even if the difference is less clear than with the previous method of step extraction.

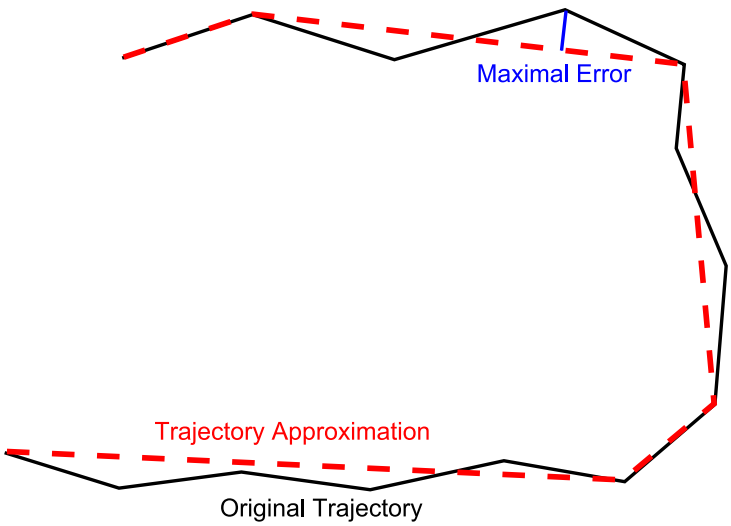


Figure 5.22: Second method to extract CRW distributions. A maximal approximation error must be selected in advance. Then the longest segment is chosen between two trajectory points, where the points in-between have a smaller distance to this segment than the desired error

Table 5.3: Maximal difference of CDF between the four histograms shown in Figure 5.23. All results are in percentage

	1^{st} set	2^{nd} set	3^{rd} set	4^{th} set
1^{st} set	0	10.0	38.8	32.3
2^{nd} set	10.0	0	43.6	38.7
3^{rd} set	38.8	43.6	0	12.6
4^{th} set	32.3	38.7	12.6	0

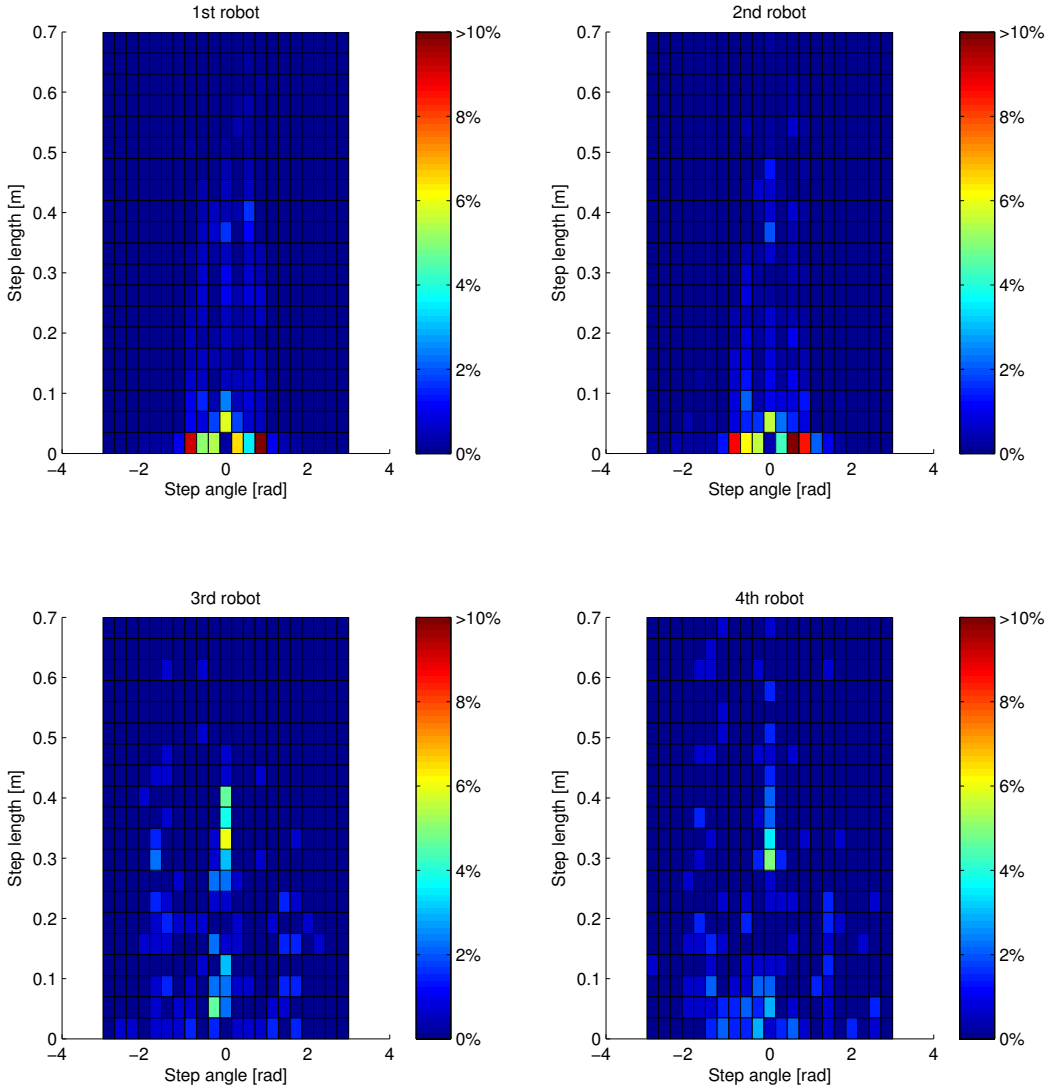


Figure 5.23: Bidimensional histogram of the step lengths and step angles, for the four trajectories shown in Figure 5.18. The second method was used to extract these distributions

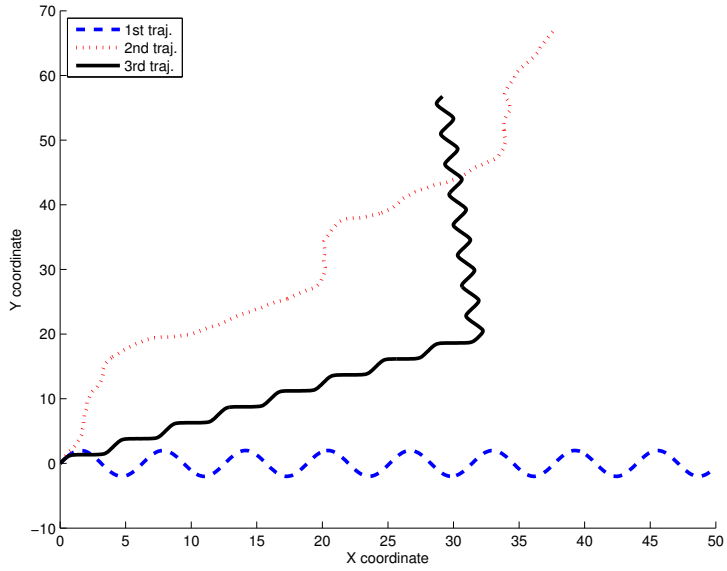


Figure 5.24: Example of trajectories whose difference cannot be observed with the CRW model.

5.3.3 Limits

As seen in the previous example, the CRW model can be used to observe differences of trajectories without common frame. However, as only the step distributions are stored, the way they are successively organized is lost. Three trajectories were created with the same series of step lengths and steps angles, but not in the same order. Figure 5.24 shows them and the difference between the two *organized* trajectories and the more random one is clearly visible. However, as they come from the same distribution, the CRW model will fail to see differences between them.

5.4 Conclusion

This chapter described several techniques and methodologies to compare trajectories. There is no optimal solution for all the analyses. Simple statistical tests are clearly the fastest and easiest solutions to implement. However, they remain quite limited as they do not take into account the temporal sequence of the trajectories.

The Point Distribution Model is shown as a good solution to reduce the dimensionality of the problem, allowing a better understanding of the trajectory differences and a simplification of the classification process. However, it keeps its own limitations and for example the Linear Discriminant Analysis can be more efficient in some cases, in particular when the goal is to maximize the separation between trajectory sets.

Correlated Random Walk models were presented as an alternative solution to analyze trajectories without common frames that cannot be modeled by Point Distribution Models. Even if it can be applied to all kind of trajectories, it remains limited, as it does not capture the relationship between successive

steps. If this relation does not exist in reality, it would be the perfect model. However, as it is rarely the case, a good chunk of information is lost.

Finally, it is important to keep in mind the objectives when choosing the analysis technique. For example, statistical tests only compare characteristics of point distributions, such as the mean, the variance or the CDF. On the contrary, the PDM takes into account the temporal sequence of the trajectory points. Even if the resulting analysis includes more trajectory characteristics, only the analysis objectives can define if they are needed or not.

Chapter 6

PDM analysis of simulated trajectories

This chapter aims to study the different parameters influencing the PDM comparison of trajectories. As there is a large amount of parameters characterizing a given experimental scenario, all these experiments were realized in simulation to speed up the acquisition process of trajectory data. Moreover, the simulation allows us also to focus only on selected parameters, as undesired influences can be kept out.

Two kinds of trajectories are studied in this chapter, both having a common frame and characterized or not by a drift. The first section studies trajectories generated in a circuit. As the variation are constrained by the circuit width, no drift can appear. The second section presents the trajectories of a robot relying only on odometry to localize itself. Thus, the variations are cumulated along its path and drifts can be observed.

6.1 Trajectories with a common frame and without a drift

This section focuses on the use of the PDM (Chapter 3) to classify the trajectories of a mobile robot driving on a circuit. The goal is to analyze how modifications of the environment (circuit), the hardware (sensor, motors,...) and software (controller) of the robot influence the resulting trajectories. The main results were presented at the International Conference on the Simulation of Adaptive Behavior (SAB) [91]. The parameters influencing the PDM analysis and its limitations are also studied hereby.

6.1.1 Experimental setup

As a lot of trajectories were needed, the experiences were carried out in Webots [92], a realistic simulator, to speed up their generation. This software reproduces individual sensors and actuators with noise, non-linearities, and dynamic effects such as slipping and friction. A differential-drive mobile robot, reproducing the Khepera II platform, was driving on a circuit created manually. Figure 6.1 shows the simulated robot between both walls of the circuit. The Khepera robot is endowed with eight infrared proximity sensors, which are represented in the figure. A few circuits were created, but only the results generated on two different simulated circuits will be presented here, as the other circuits do not add relevant information.

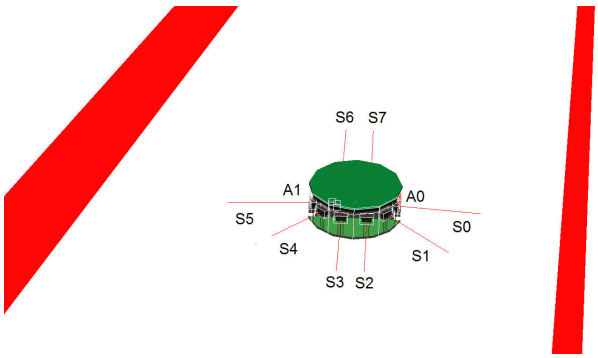
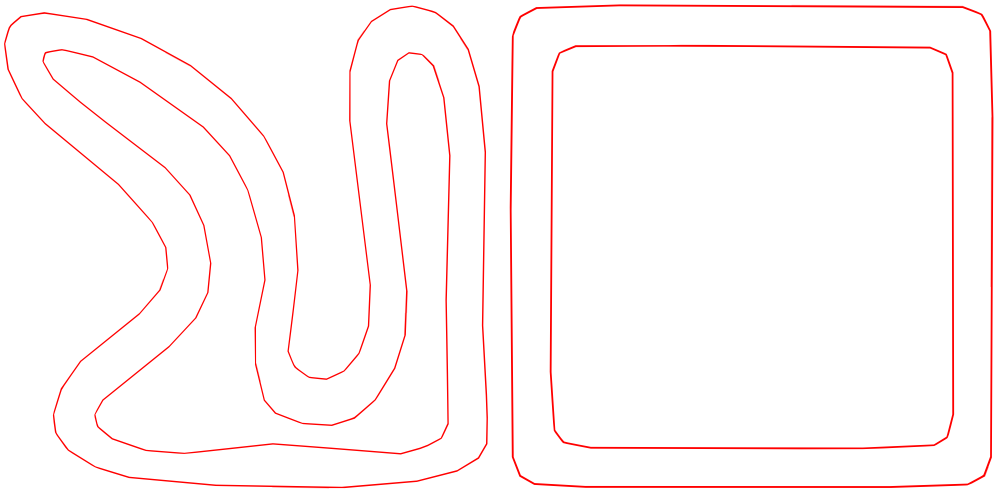


Figure 6.1: Image of the simulated mobile robot (Khepera), with its 8 infrared sensors (S0 to S7) and two wheels (A0 and A1). The two walls of the circuit can also be seen



(a) (b)

Figure 6.2: The two circuits simulated in Webots (1 on the left, 2 on the right)

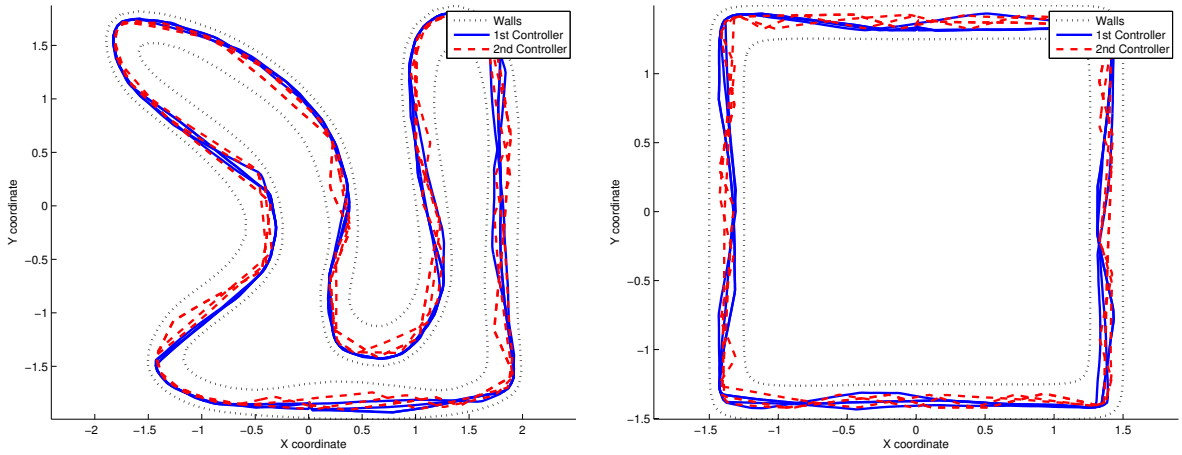
Table 6.1: Description of the two controllers used for the experiments

Controller 1	Controller 2
If $\sum_0^2 S_i < T \Rightarrow \{^{A_0=V}_{A_1=-V}$ Else if $\sum_3^5 S_i < T \Rightarrow \{^{A_0=-V}_{A_1=V}$ Else $\{^{A_0=V}_{A_1=V}$	$S_l = \sum_0^2 S_i$ $S_r = \sum_3^5 S_i$ $A_0 = V \cdot \left(1 + \frac{K \cdot (S_l - S_r)}{2 \cdot (S_l + S_r)}\right)$ $A_1 = V \cdot \left(1 + \frac{K \cdot (S_r - S_l)}{2 \cdot (S_l + S_r)}\right)$
$S_0 \ldots S_5$ are the robot sensors as shown in Figure 6.1 (back sensors S_6 and S_7 are not used in either of the controllers) A_0 and A_1 are the robot actuators as shown in Figure 6.1 T is a constant threshold value V is a parameter modifying the robot's overall speed K is a parameter modifying the robot's reactivity	

The two circuits are shown in Figure 6.2. They are quite different with an overly simplified circuit made of four straight lines and four curves, and another more complex circuit. This huge difference intends to show the influence of the circuit configuration on the PDM analysis. However, there are common features, such as being characterized by only one lane and a closed loop.

We simulate the robot moving continuously within the two circuits, extracting each lap as a separate trajectory. Since a lap does not begin and end at the same point, it adds variability to the trajectories. Two different reactive controllers were implemented to drive the robot on the circuits. The two controllers move it around the track at the same average speed, but using different methods to avoid the walls. The first controller was rule-based ("if a wall is too close in front, turn away from the wall, otherwise go straight"). The second controller is essentially a Braitenberg vehicle and linearly adjusts its trajectory as a function of its proximity to a wall on the left or the right side. The robot was moving clockwise in both circuits. Both controllers continuously calculate the perception-to-action loop every 32 ms. The description of the two controllers can be found in Table 6.1. The two controllers were chosen for their simplicity, but the analysis has very few limitations and could easily be used with more complex controllers.

From this description, we can see that Controller 1 is characterized by essentially two discrete behaviors: go straight or turn in place. Controller 2 makes much smoother turns and its overall behavior changes as a function of the distance to the left or right wall. Figure 6.3 shows three trajectories per controller on each of the two circuits. Slight differences can be seen between the two controllers; for example, Controller 2 makes more zigzags than Controller 1 (even if its turns are smoother, it turns more often than Controller 1). At first glance, it is not so easy for the human eye to differentiate between the raw trajectories.



(a) (b)

Figure 6.3: 6 trajectories (3 for each controller) of the robot's movement simulated on the 2 circuits

6.1.2 Trajectory sampling and modeling

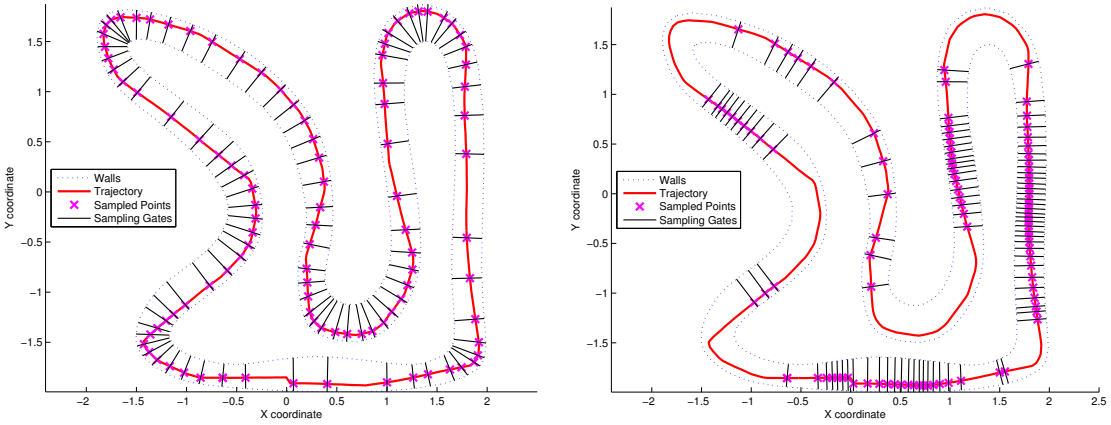
In order to apply the *Point Distribution Model* presented in Chapter 3, each trajectory must be sampled with the same number of points. We used the three methods using gates presented in Section 4.1.2.2. To simplify the notation, the method with equidistant gates is called the sampling method A (SM-A), the other based on the circuit curvature is called (SM-B) and the last based on circuit straightness is called (SM-C). SM-B puts more gates in the curves (as shown in Figure 6.4(a)), and SM-C places more gates in the straight sections (Figure 6.4(b)). These sampling methods allow us to very easily modify the number of gates per circuit. The performance of the three placement strategies will be compared hereafter.

The points resulting of the trajectory sampling are then used to make a PDM analysis. The following results focus on purely spatial analysis. Thus, only the positions on the sampling gates were used to create the multidimensional input vectors of the PDM.

6.1.3 Analysis using the first two modes of the PDM

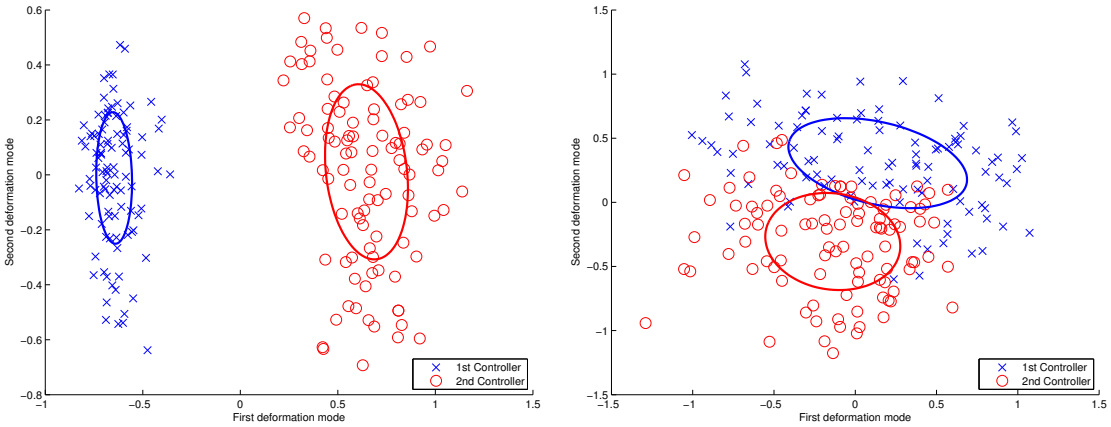
To show the performance of the PDM for clustering controller trajectories, we acquired 200 trajectories (100 for each controller presented in Table 6.1) on the two circuits (Figure 6.2(a) and 6.2(b)). The trajectories were then re-sampled with 100 points per trajectory, using the gate selection criterion with more gates in the curves (SM-B, $\alpha = 0.5$). Then, we modeled all the trajectories using the PDM. Figure 6.5 shows the locations of the 200 trajectories in the space formed by the first two modes of the PDM for each of the two circuits; two clusters can be easily differentiated. The clear separation of the controllers shows the benefit of the PDM modeling of the trajectories. The intrinsic variance of the controllers is smaller than the distance between them. Therefore the trajectories can be clustered and hence classified.

The separation of the first 2 clusters is narrower for the second circuit than the first. The lack of curves and the predominance of straight lines reduce the number of obstacles and therefore the number of controller reactions from which the PDM transformation can extract its data. As their straight movements



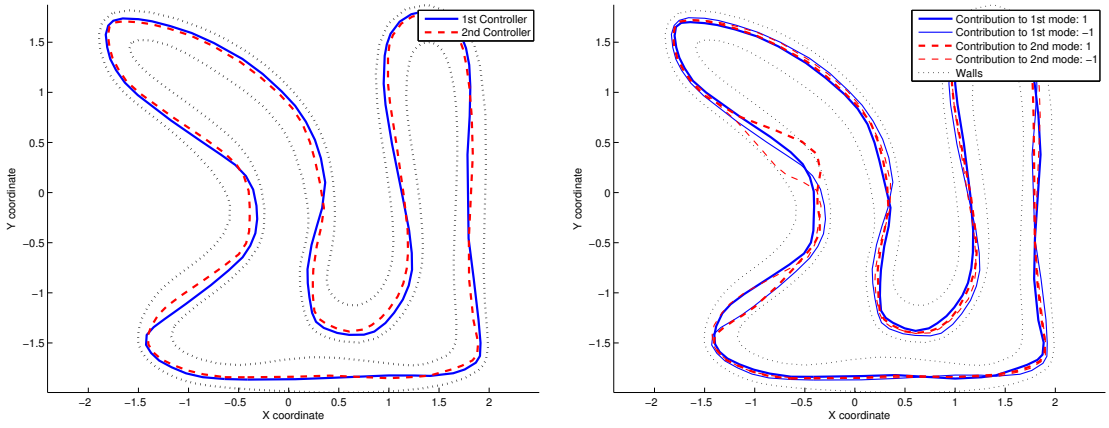
(a) (b)

Figure 6.4: Sampling of the trajectory with gates as orthogonal to the wall as possible, with more gates in the curves (SM-B, left, $\alpha = 0.5$) and more gates in the straight lines (SM-C, right, $\alpha = 50$)



(a) (b)

Figure 6.5: Projection of all the trajectories in the space of the first two deformation modes of the PDM for the first (left) and the second (right) circuit. The ellipse of unitary Mahalanobis distance is also plotted for each controller



(a) (b)

Figure 6.6: On the left, prototype trajectory of the two controllers on the first circuit. On the right, synthetic trajectories resulting from a contribution of ± 1 to one of the first two modes (first circuit)

are equivalent, it becomes more difficult to detect differences and therefore classify the two controllers. If we calculate the inter-cluster distance as presented in section A.5.2, $d = 8.0$ for the first circuit and $d = 1.9$ for the second circuit.

6.1.4 Prototype trajectories

Figure 6.6(a) displays the mean of each controller's trajectories. These trajectories are called prototypes of each controller. Slight differences appear at certain locations in the circuit; these are the places where the controllers can be differentiated.

Figure 6.6(b) shows the synthetic trajectories resulting from a positive or negative contribution of value 1 to the first or the second deformation mode for the first circuit. They corresponds to points situated at the coordinates $[1, 0]$,$[-1, 0]$,$[0, 1]$ and $[0, -1]$ in Figure 6.5(a) translated to the trajectory space. We can see that the first mode is spread on the whole circuit. There is however a big deformation after the lower left curve. The second mode is mainly influenced by the trajectory parts lying in the two straight sections on the left side of the circuit.

Another view can help us to see which parts of the circuit influence the first mode. We can plot the values of the first vector of the transformation matrix $\boldsymbol{P}$ (Equation 3.3) at the location of their respective sampling gates. Figure 6.7 shows this plot, where the values of P_1 were plotted. These values were also multiplied by the length of their respective gates. As the PDM was built on the positions on the sampling gates, this multiplication allows to go back to spatial dimensions.

This plot shows that the main variance is located near the lower curves of the circuit (on the right on this particular plot). And as it was observed before, the variance is spread over the circuit.

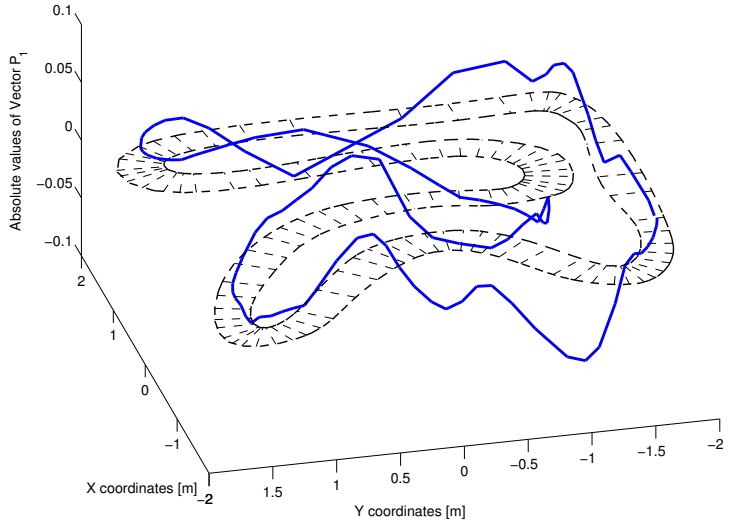
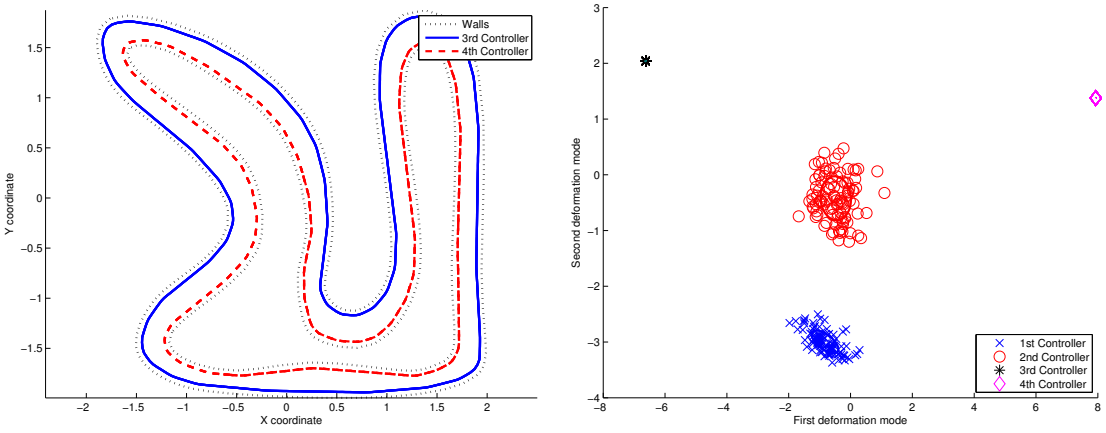


Figure 6.7: Plot of the first vector values of the transformation matrix (P_1) at the location of their respective sampling gates. These values were multiplied by the length of their respective gate, to have a spatial representation. The circuit gates and walls are also plotted.



(a) (b)

Figure 6.8: On the left, 6 trajectories (3 per controller) generated with two new controllers following the left and the right wall respectively. On the right, projection in the space of the first two deformation modes of the PDM of 400 trajectories (100 for each of the previously used controllers and 100 for each of the new controller). The ellipse of unitary Mahalanobis distance is also plotted for each controller

Table 6.2: Inter-cluster distance measured between the clusters of Figure 6.8(b)

	C1	C2	C3	C4
C1	0	8.7	35	70
C2	8.7	0	19	25
C3	35	19	0	10^4
C4	70	25	10^4	0

6.1.5 Alternate controllers

To show that the analysis is not working only with these two controllers, we created two others. They are driving the robot with a wall following behavior, one following the left wall and the other the right one. Figure 6.8(a) shows 6 trajectories (3 from each controller). As the movements are really repeatable, the differences between the trajectories of the same controller are really hardly noticeable on this plot. Figure 6.8(b) shows the result of the PDM analysis applied to 400 trajectories: 200 that we already used before (Figure 6.5(a)) and 100 trajectories for each of the new controllers. On this last plot, all the four controllers are clearly separated. Moreover, the two new controllers with much more repeatable movements show less dispersion in the space of the PDM modes and are nearly reduced to one point. Thus, less spatial variation induces less variation in the space of the PDM modes. Finally, the first two modes are clearly sufficient to easily classify the trajectories. The inter-cluster distance between each cluster is reported in Table 6.2.

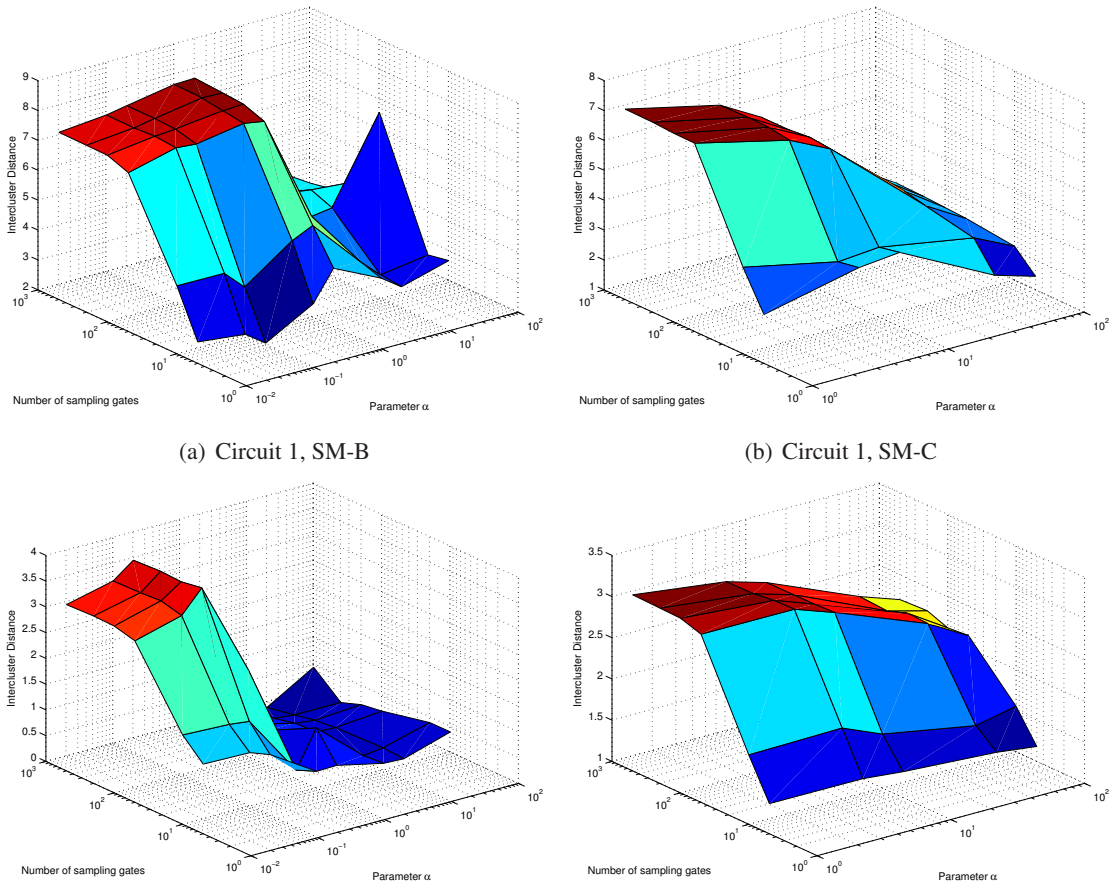
6.1.6 Sampling influence on the clustering

The sampling has a major influence on the separation of the clusters. Two parameters where thoroughly explored: the number of sampling gates and the location of these gates.

Increasing the number of gates will directly increase the size of the data we are working with. However, a too small number of gates will reduce the information too much to have a clear separation. Thus, an optimum has to be found between the needed computation power and the desired cluster separation. To study the gate location influence, we apply the three sampling methods (SM-A, SM-B and SM-C) to the trajectories generated on the two circuits presented before. We used also multiple values of the parameter α (Equations 4.4 and 4.5) to increase the number of sampling gates in the curves and in the straight parts of the circuit.

Figure 6.9 shows the inter-cluster distance as a function of the number of gates and as a function of the parameter α for sampling methods SM-B and SM-C. From these four plots, we can see that increasing the number of gates increases the distance between the two clusters. However, there is an apparent threshold around 100 gates for both sampling methods and for both circuits. Above this number, increasing the number of gates does not seem to increase the cluster separation.

Increasing the parameter α reduces the spreading of the gates. With a high value of α, the gates will be only in the sharp curves of the circuit or in the really straight parts of the circuit. Therefore, there is a large drop of information, as the gates are located nearly at the same place and are thus highly



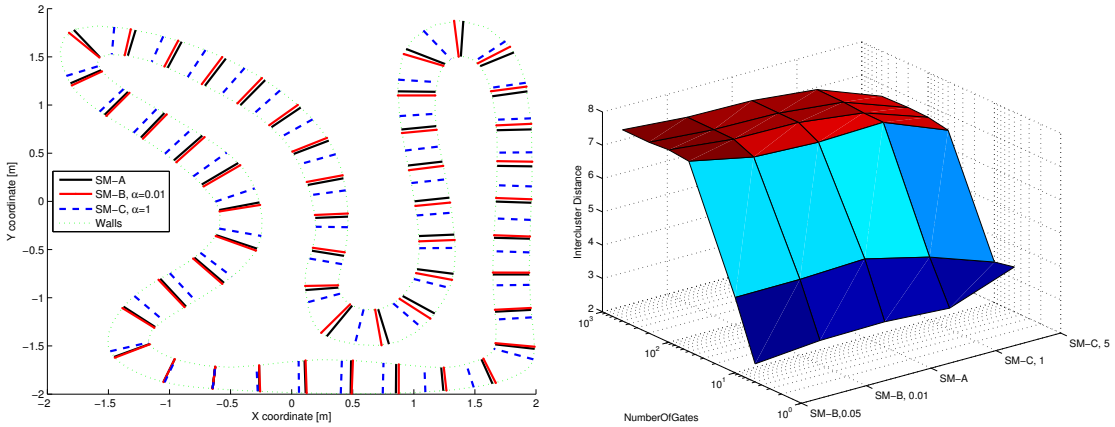
(a) Circuit 1, SM-B

(b) Circuit 1, SM-C

(c) Circuit 2, SM-B

(d) Circuit 2, SM-C

Figure 6.9: Influence of the number and position of the sampling gates on the cluster separation. The distance between the cluster is computed exactly in the same way as in Section 6.1.3. Only the number of gates and the parameter α of the sampling methods have been modified. For SM-B, increasing α increases the number of gates in the curves and for SM-C, it increases the number of gates in the straight lines. Increasing the number of gates increases the cluster separation, but increasing the parameter α reduces the inter-cluster distance



(a) (b)

Figure 6.10: On the left, sampling gate location for small values of α. The three sampling methods select gates nearly at the same location. On the right, the inter-cluster distance for small values of α for the three sampling methods. We can observe that the results are quite similar

redundant. It explains thus the great decrease of the inter-cluster distance when α becomes too large. When α is close to zero, SM-B and SM-C seems to gives the same results. It is not surprising if we look at Equations 4.4 and 4.5, as the equation becomes the one of SM-A:

$$Step = \frac{1}{N}\int_0^S \sqrt{x'(s)^2 + y'(s)^2}ds. \tag{6.1}$$

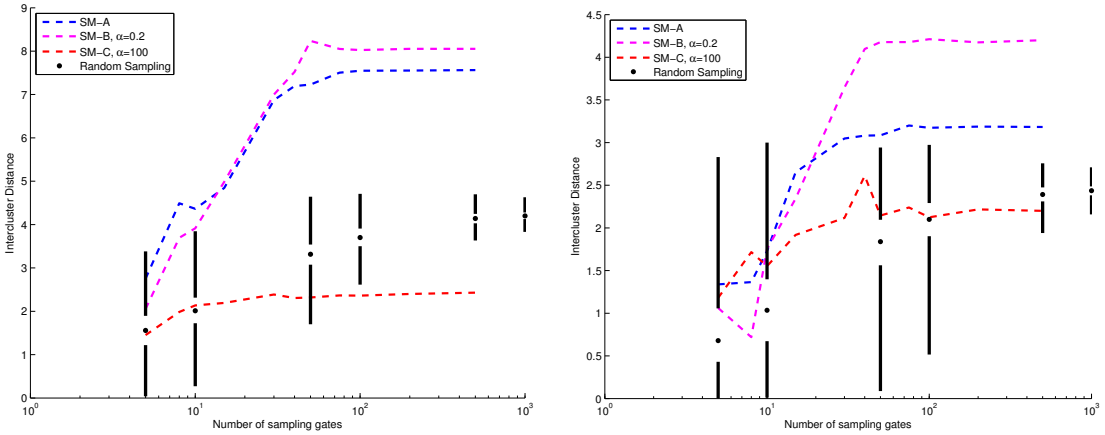
Figure 6.10 shows the location of the sampling gates and the inter-cluster distance for small values of α. We can clearly see that in this case, the three sampling methods provide really similar results.

Finally, for both circuits, we can observe that for SM-B, the optimal value of α is not the smallest one. Putting more gates in the curves, especially when they are few gates, increases the separation of the two clusters. If we look at the description of the controllers at the origin of these clusters, their difference exists only when they meet a wall. Without obstacles, both drive straightforward at the same speed. As the robot will mainly encounter walls in the curves, it is not surprising that taking more information in the curves leads to a better separation of the two controllers.

These observations are highly bound to the circuit shape and to the controllers. However, too few sampling gates leads to data scarcity and results most of the time in a worse separation of the controllers. On the contrary, increasing the number of gates increases the information at our disposal and leads to a better separation of two controllers in most cases.

6.1.7 Random sampling

To evaluate the performance of our method to choose the sampling gates, we made a comparison with a purely random selection of them. As it is described in Section 4.1.2.2, a very large number of gates is first generated on the circuit (8617 and 2940 for the first and the second circuit, respectively). Then, a



(a) Circuit 1 (b) Circuit 2

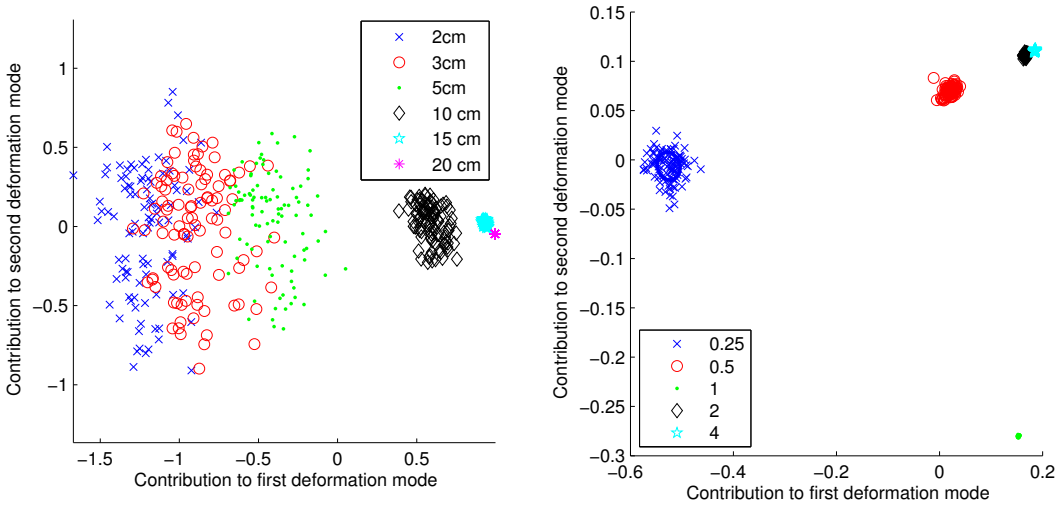
Figure 6.11: Box-plots of the resulting inter-cluster distance, when choosing the gates randomly. The results achieved with the three standard sampling methods are plotted here as a comparison.

specific number of gates (e.g. 100) are selected using SM-A, SM-B or SM-C. In this specific case, for each of the selected numbers of gates (5, 10, 50, 100 and 500), the gates were selected randomly 1000 times. Figure 6.11 shows the resulting distance between the two clusters. The inter-cluster distances for the three sampling methods presented before are also plotted as a comparison. The random selection of the sampling gates is really not a good solution and is clearly worse than the deterministic solutions presented before. It can be observed, that increasing the number of gates clearly increases the separation between the clusters and clearly reduces the dispersion of the results. For a small number of gates (~ 5), the average result from the 1000 random runs, is not really worse than the deterministic methods of selecting the gates. Even, some random results can be clearly better than the standard methods. It is not really surprising, as the results of these methods are not good for a small number of gates. Finally, a random selection of the gates can choose, by pure luck, the same gates as Method SM-A, SM-B or SM-C. Thus, if a biggest number of random experiments where done, the best random results would be better, or at the minimum equal, to the results achieved with the deterministic methods.

6.1.8 Variation of the robot hardware and software parameters

To complete the performance evaluation of our method, we analyze the influence of possible system design choices on the resulting analysis: the controller reactivity, the overall robot speed, and the sensory range. For these experiments, we report only results obtained using Controller 2 for sake of clarity. Referring to Table 6.1, the reactivity corresponds to K and the overall speed to V.

Figure 6.12(a) shows the analysis of the variation of the sensory range from 2 to 20 centimeters. The real range of the sensors (5 cm) is shown with the small lines in Figure 6.1. For ranges of 10 centimeters and above, the robot will almost always be able to see both walls at the same time. Therefore, the variability of the controller will decrease significantly, as the robot will follow a path in the middle of the lane. Naturally, the greater the sensory range, the smaller the variability of the controller. The different



(a) (b)

Figure 6.12: On the left, analysis of the variation of the sensory range from 2 to 20 cm. On the right, analysis of the variation of the controller weights (K), for a sensory range of 20 cm

sensory ranges are mainly separated using the first deformation mode in Figure 6.12(a): one variation axis (sensory range) corresponds to one dimension for classifying the different clusters.

Figure 6.12(b) shows the clustering of the reactivity of the controller. The factor K varies between 0.25 and 4, the reactivity increasing with K. If K is small, the robot avoids the wall with more inertia and thus oscillates much more. As as result, the dispersion of trajectories in the mode space is much greater, as can be seen in Figure 6.12(b). With a sensory range of 5 centimeters and $K = 0.25$, the robot was not able to avoid colliding with the walls anymore. Therefore, this experiment was made with a sensory range of 20 centimeters ($V = 1$).

Figure 6.13 shows the result of the variation of the overall robot speed factor V. Similarly to the previous experiment, with the robot overall speed increased ($V = 4$), the robot was not able to avoid walls with a sensory range of 5 cm. Therefore, this experiment was performed with a sensory range of 20 cm. As with a human driver, an increase in the overall speed means that the robot pass closer to the walls before avoiding them, resulting in larger oscillations. This variability can be seen in Figure 6.13.

6.1.9 Influence of the track width

To measure the influence of the track width on the separability of the controllers, test were made with circuits formed of two circular walls. We wanted also to analyze which circuit parts were selected by the PDM to separate the two controllers. Thus, we tested concentric circles with a variable radius for the inner circle and non concentric circles.

The experiments were made with Webots, simulating an e-puck mobile robot. The simulation parameters and the robot controllers where the same as in the previous experiment (Table 6.1). The sampling

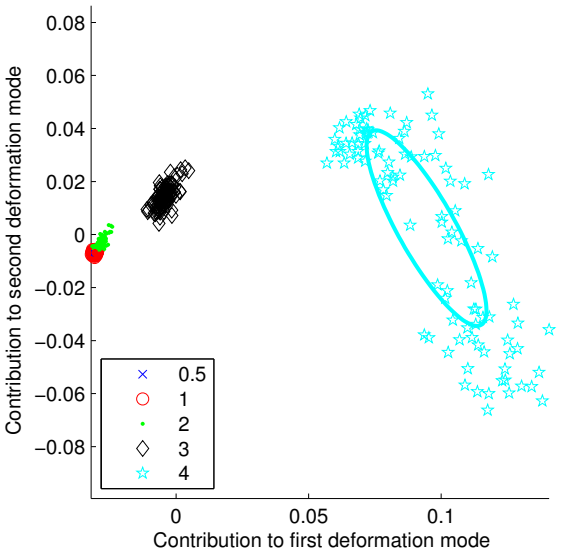


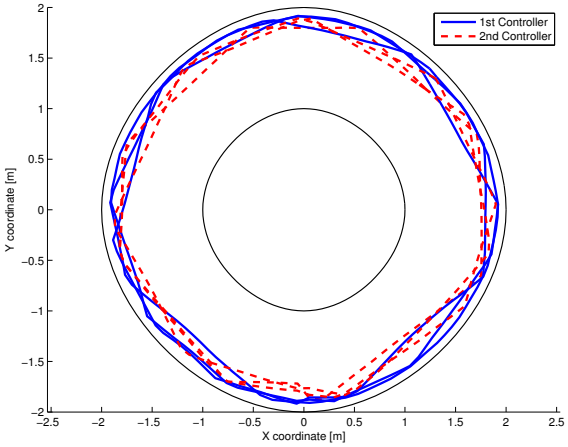
Figure 6.13: Analysis of the variation of the overall robot speed (V), for sensory range of 20 cm

was realized with 100 equidistant gates.

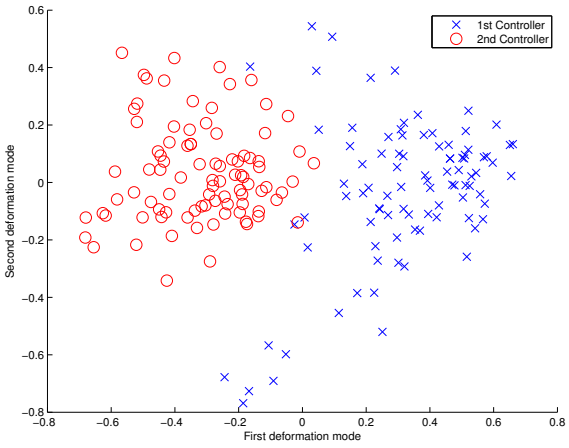
6.1.9.1 Concentric circles with variable radius

For these experiments, we used a circular outer wall with a radius of 2 meters. Three values of the inner wall radius were tested: 1, 1.5 and 1.7 meters. 90 trajectories were generated with each of the two controllers described previously (Table 6.1). Then, a PDM analysis was applied to the whole dataset. Figure 6.14 shows on the left the trajectories generated on the three circuits, and on the right, the location of the trajectories in the space of the first two modes of the PDM. Comparing Figures 6.14(a) and 6.14(c), we can observe that the inner wall has no real influence on the trajectories, as the robot is moving too far from it to be influenced. In this case, having a smaller or a bigger radius will not influence the trajectories and thus the PDM analysis, as shown in Figures 6.14(b) and 6.14(d). In the third case, with an inner radius 1.7 meter, the trajectories are clearly influenced by the inner wall, especially for the second controller (Figure 6.14(e)). For this controller, the two walls guide the mobile robot between them and thus reduce the variability of the trajectories. Figure 6.14(f) shows this reduction in the much more grouped cluster corresponding to the second controller. We can observe that this particular track width has much less influence on the first controller.

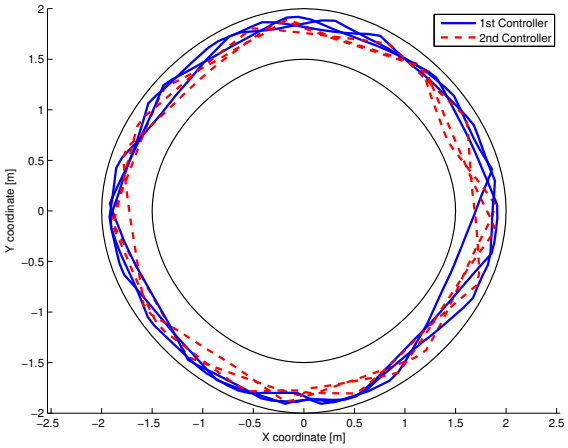
Figure 6.15 shows the three plots of the first vector of the transformation matrix (P_1) resulting from the PDM analysis of the three dataset presented before. These values were multiplied by their respective gate length. On the figure, there is a good repartition of this vector on the whole circuit. The plot is not completely flat, but there is no real trend to observe. The influence of the third circuit on the trajectories and the analysis is clearly visible: there are shorter oscillation that can be linked to the more frequent bumps of the robot with the walls.



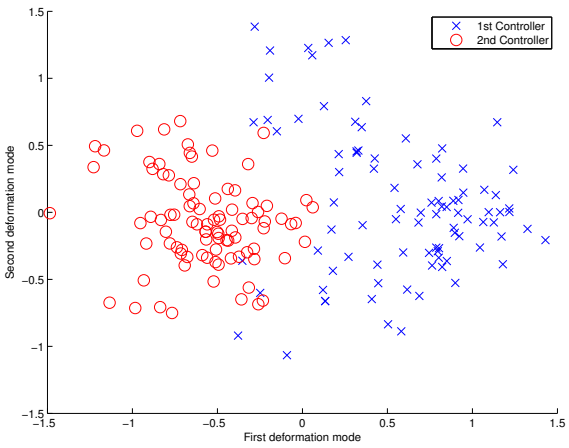
(a) Inner radius: 1 meter



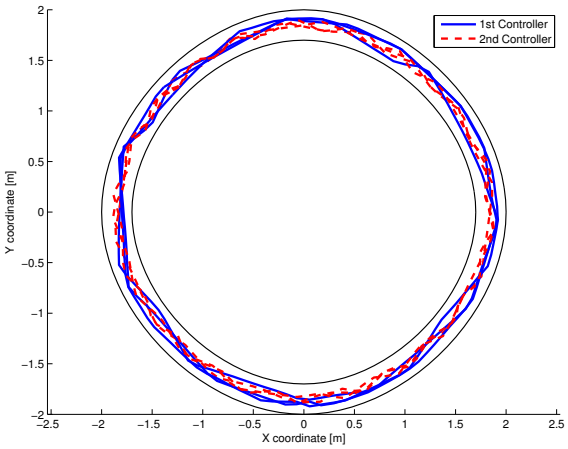
(b) Inner radius: 1 meter



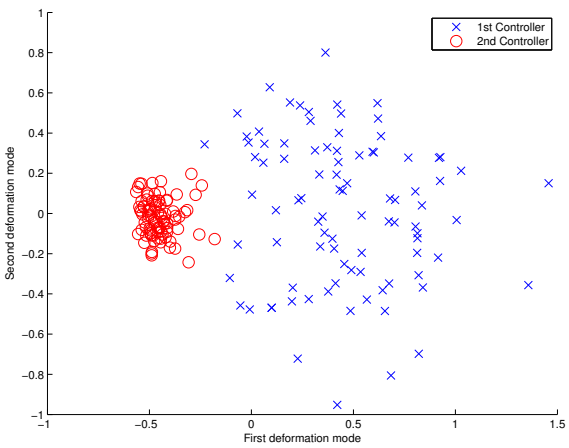
(c) Inner radius: 1.5 meter



(d) Inner radius: 1.5 meter

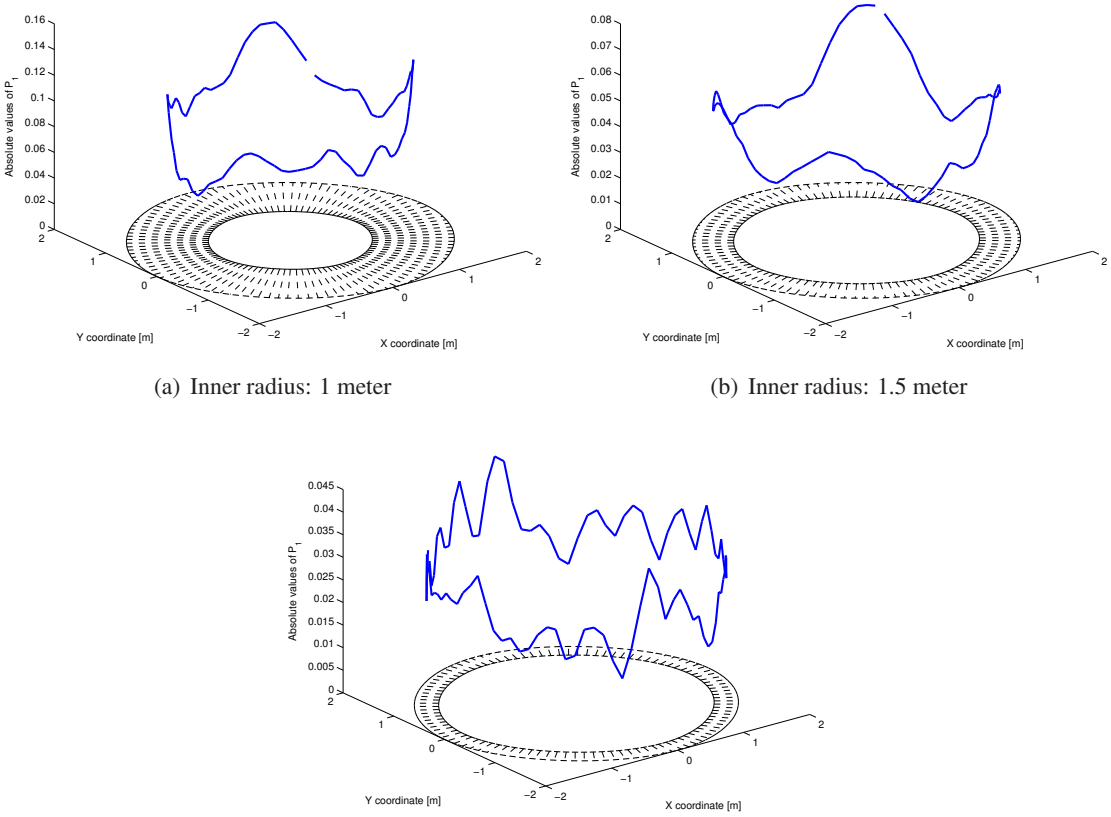


(e) Inner radius: 1.7 meter



(f) Inner radius: 1.7 meter

Figure 6.14: Three circuits made with concentric circles. The outer circle has a radius of 2 meters, when the inner circle has a variable radius. On the left, six trajectories (3 for each controller) are plotted as an example. On the right, the result of a PDM analysis made with 90 trajectories for each controller



(a) Inner radius: 1 meter

(b) Inner radius: 1.5 meter

(c) Inner radius: 1.7 meter

Figure 6.15: Plot of the first vector values of the transformation matrix (P_1) at the location of their respective sampling gates. These values were multiplied by their respective gate length. The three circuits correspond to those plotted in Figure 6.14 and are also plotted with the sampling gates

6.1.9.2 Non concentric circles

Circuits made of concentric circles are simple and help us to understand how the track width can influence the mobile robot trajectories and their PDM analysis. However, as they are perfectly symmetric, the position of the sampling gates having the main influence (biggest values of P_1) cannot be linked to characteristics of noticeable circuit parts.

Thus, four circuits were created in simulation with non-concentric circles. They were made of two circles with radius of 1.5 and 2 meters. Four distances between the circle centers were investigated: 0.1, 0.2, 0.3, and 0.35 meters. Two controllers, the same as in the previous experiments (Table 6.1), were used to generate 180 trajectories, 90 per controller. These trajectories were then sampled with 100 equidistant sampling gates and a PDM analysis was made. Figure 6.16, on the left, shows the location of these trajectories in the space of the first two modes of the PDM, for the four circuits. The first three plots are quite similar and even the fourth is not so different from the other.

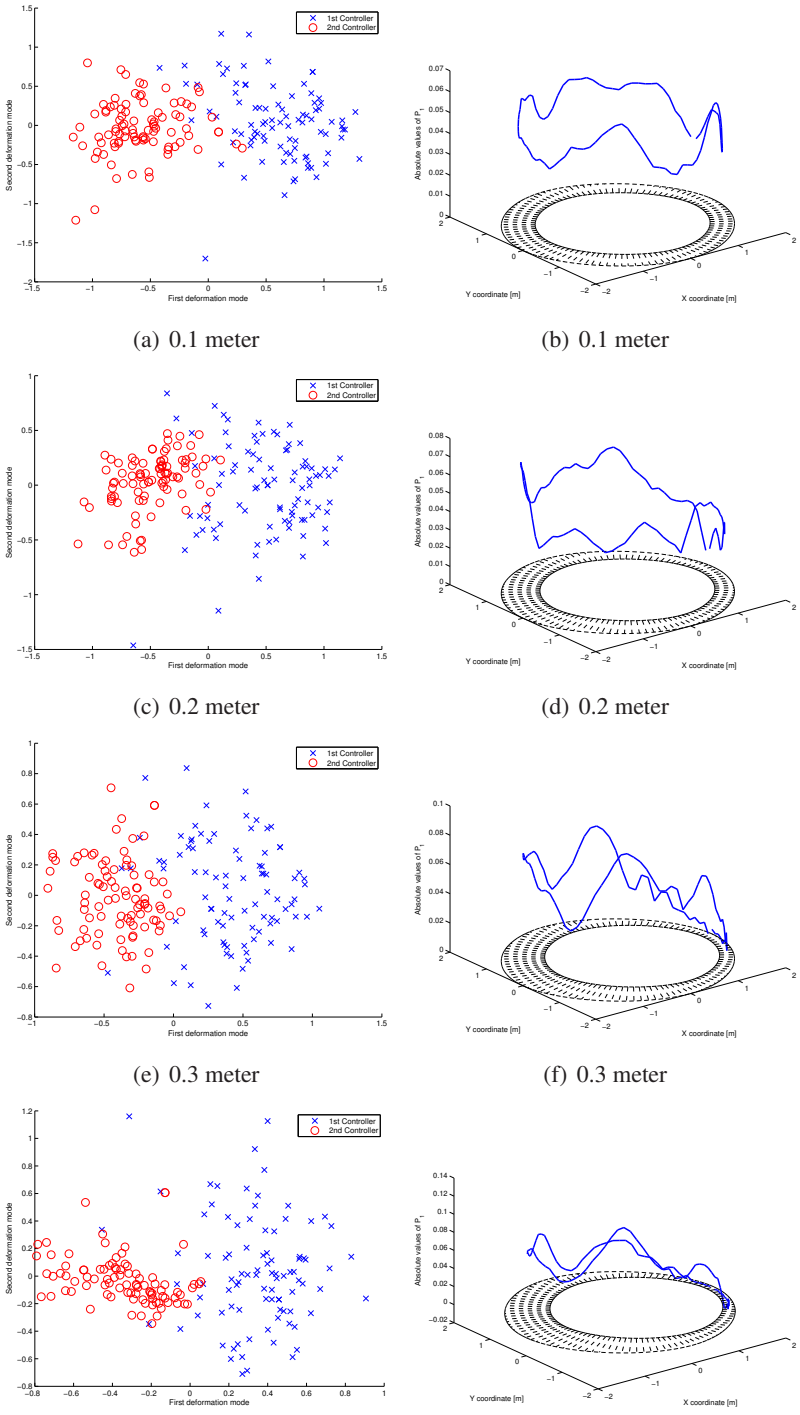
The most interesting part lies on the right plots. There, the first vector of the transformation matrix (P_1) of the PDM analysis is plotted for the four dataset. The values of this vector are multiplied by their respective gate length. For 0.1 meter, the first axis of the PDM is well spread across the whole circuit. However, as the distance between the circle centers increases, the distribution of the vector values is really influenced by the track width. As the track becomes narrower, the robot is much more guided and thus the trajectory variations are reduced. As a result, the PDM will less emphasize on the narrowest part of the circuit, and thus the corresponding values of P_1 will be smaller. For 0.3 and 0.35 meters, we can observe that the most important gates are located where the trajectories begin and finish to be guided by the walls. At this location, the Braitenberg controller will clearly guide the robot in the middle of the two walls, when the other controller is mainly influenced by the outer wall.

The track width can have an impact on the PDM analysis, but it really depends on the robot hardware and software, and on the circuit configuration. However, studying the most important gates of the PDM is still interesting as it shows which circuit parts induce most of the trajectory variations.

6.1.10 Divided track

In all the examples of PDM analysis studied previously, the trajectories generated with the same controller were always regrouped in a more or less compact cluster. To make a further study, we wanted to study an experimental setup where one controller would be represented by multiple clusters in the PDM space. Thus, we created a circuit with a separation of part of its track in two branches. Figure 6.17(a) shows the created circuit in Webots. We create also this circuit in such a way that the robot is using each branch more or less half of the time. The two controllers previously described (Table 6.1) were used to drive a simulated e-puck robot on the circuit. Two trajectories for each controller are plotted in Figure 6.17(b) as examples. 100 trajectories for each controller were generated and then sampled with 100 equidistant sampling gates. On the circuit part where it is separated in two branches, the sampling gates cover both branches at the same time. The sampling gates can be observed in Figure 6.17(b).

A PDM analysis of the trajectories was then performed. The location of the trajectories in the first three dimensions of the PCA are plotted in Figure 6.18(a). The two controllers are now represented by



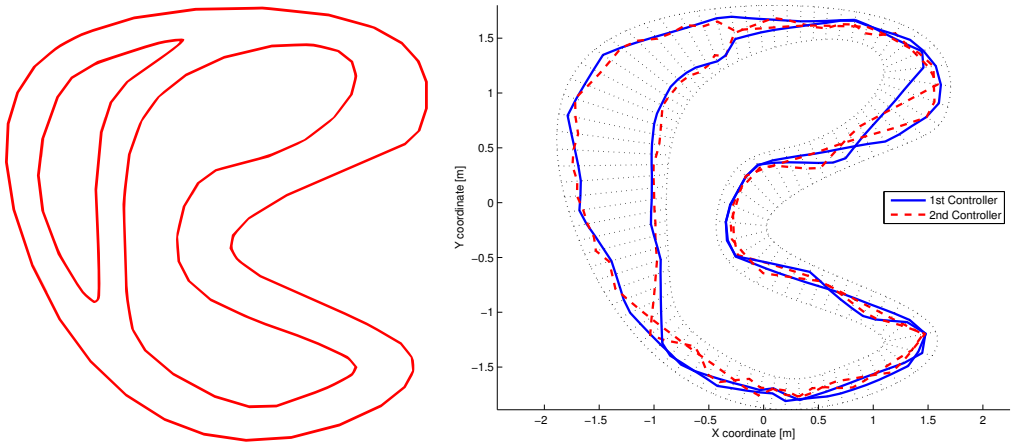
(a) 0.1 meter (b) 0.1 meter

(c) 0.2 meter (d) 0.2 meter

(e) 0.3 meter (f) 0.3 meter

(g) 0.35 meter (h) 0.35 meter

Figure 6.16: Four circuits made of two non concentric circles of radius 1.5 and 2 meters. The distance between the two circle centers goes from 0.1 to 0.35 meters. On the left, PDM analysis of 180 trajectories (90 for each controller) and on the right, plots the of first vector of the transformation matrix (P_1) of the PDM analysis



(a) Webots circuit (b) Sample Trajectories

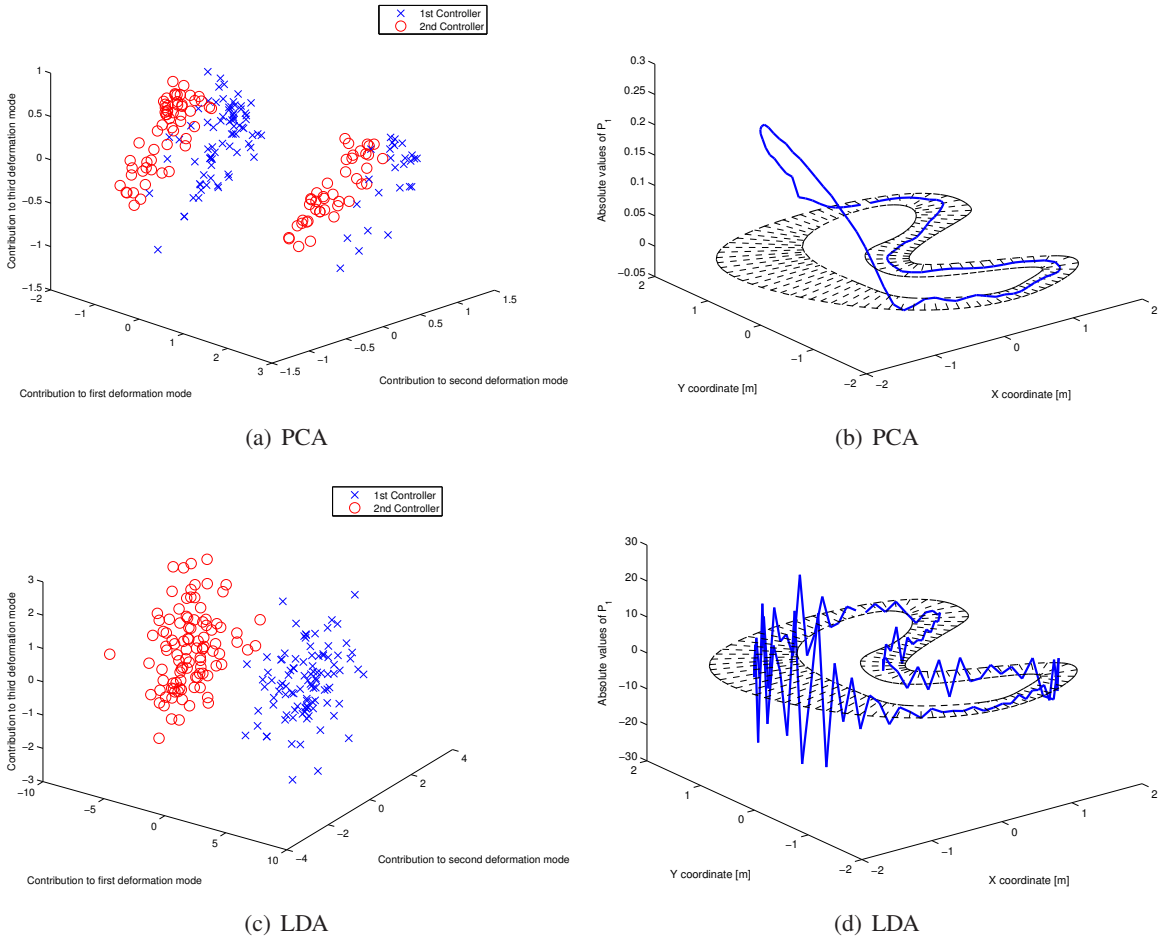
Figure 6.17: Circuit with a track splitting in two sub-track along a part of the circuit. Figure on the left shows the circuit as made in Webots. Figure on the right shows two examples for each of the two controllers used. Both controller have a trajectory going through each branch of the circuit. The plot shows also the sampling gates

two clearly separated clusters. Two groups containing one cluster of each controller can also be observed. The first axis of the PCA is clearly separating these two groups, corresponding to the trajectories going through one or the other branch of the circuit. However, it seems that the second and the third axes of the PCA would be sufficient to separate the two controllers.

Figure 6.18(b) shows the first vector values of the PCA transformation matrix (P_1) at the location of their respective sampling gates. These values are multiplied by the gate length. This plot shows that the first mode of the PCA focuses on the circuit part where the track is split in two. The average trajectory will be more or less in the middle of the two circuit branches. Thus, there is quite a big distance between the trajectories and the average trajectory. Therefore, the trajectory variance at this location is really important. As the PCA is ordering the axes based on the variance, it is thus logical that the first axis is focusing on this circuit part.

This example shows clearly that the PCA (and thus the PDM) is based on the variance. If the main part of the variance is not caused mainly by the controller differences, it is clear that this method will not be suitable to separate the controllers. In this particular case, the variance is mainly caused by the splitting of the circuit in two and therefore the first axis of the PCA separates the trajectories going through the left or the right branch of the circuit.

Another analysis was made, using the same dataset. The PCA was replaced by a LDA (Section A.6.2). The LDA is a supervised method that searches for the best axis of separation between two classes of a dataset. It is clearly different from the PCA which is unsupervised. The PCA is based on the variance and not on the knowledge of the trajectories belonging to one or the other class. Figure 6.18(c) shows the location of the trajectories in the first three modes of the LDA. In this plot, there is a clear separation of the two controllers and the first axis of the LDA is sufficient to cluster the trajectories of



(a) PCA (b) PCA

(c) LDA (d) LDA

Figure 6.18: The top left plot shows the PCA analysis of the trajectories presented in Figure 6.17. The top right plot shows the values of the first vector of the transformation matrix (P_1) of the PCA. The bottom left image shows the LDA analysis of the same dataset and at the bottom right are plotted the values of first vector of the LDA transformation matrix.

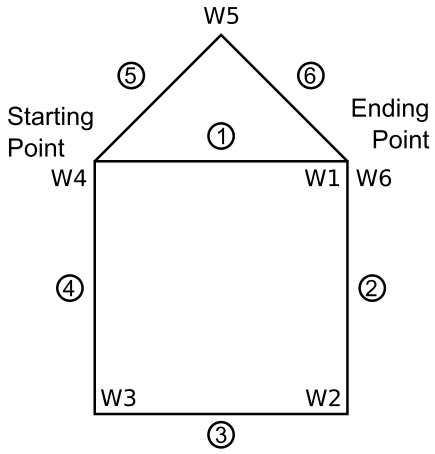


Figure 6.19: Desired trajectory drawn by the e-puck robot. The robot is going along the six edges in their numerical order. $W_1, \ldots, W_6$ are the six waypoints, and the resulting trajectory corresponds more or less to a house shape

the two controllers.

Figure 6.18(d) shows the values of the first vector of the LDA transformation matrix multiplied by the length of their respective gates. The resulting function is sawtooth: the influence of a sampling gate is nearly counter-balanced by the next sampling gate. It results in an unstable transformation matrix that is extremely efficient to cluster the trajectories corresponding to each controller.

6.2 Trajectories with a common frame and a drift

The previous experiments used trajectories generated on a circuit. Thus, the error was bound by the track width, and thus the variations were more or less constant all along the trajectories. To show an example of trajectories where the errors are cumulated along the trajectories (trajectories with a common frame and a drift), a simple case study was built in Webots. The experimental setup was made of a single e-puck robot drawing the shape of a house. Figure 6.19 shows the intended trajectory: the six straight segments and the six corresponding waypoints located at the end of each segment. The robot was always starting from the same position and with the same orientation. The robot located itself using only the odometry. The controller was simple and open-loop. The robot was moving straightforwardly until it reached the desired waypoint. Then, it was turning on the spot until the desired orientation was reached. Thereafter, the robot was alternating between straight motion and rotation, until it reached the final waypoint. The resulting movement was thus made of six straight segments separated by 5 rotations. The simulated encoders had a resolution of 10'000 steps per radian, corresponding to 20'000 steps per wheel turn and thus approximately 80'000 steps per meter. As a comparison, a Khepera III robot has a resolution of approximately 22'000 steps per meter. Two simulation parameters were then used, called encoder noise and slip noise. The first correspond to a uniform white noise added to the wheel rotation during the simulation. The second corresponds to a uniform white noise added at each step to the encoder position.

6.2.1 Encoder noise and slip noise

For the first experiment, the noise representing the wheel slip was kept to zero, but the encoder noise was modified from 5% to 50%. The resulting trajectories are shown in Figure 6.20. 100 trajectories were generated for each configuration and were then sampled with 200 points using the method based on the trajectory length (Section 4.2.2.1). The figure shows that increasing the noise increases also the spatial variations of the trajectories. However, since the noise is added all along the trajectories, the variations are more important close to the end. Moreover, as only the encoder position is noisy, the trajectories remain perfectly straight between the corners.

A PDM analysis of the trajectories was then computed using the spatial coordinates of the sampled points. Figure 6.21(a) shows the results of the analysis. As the added noise is centered, no drift is added to the trajectories, and thus all the clusters have more or less the same center. To enhance the figure visibility, the trajectory points were not plotted. Instead, the ellipse of unitary Mahalanobis distance to each cluster was plotted. The ellipse size is directly proportional to the square root of the variance (standard deviation). Thus, a more dispersed cluster will result in a bigger ellipse and a more concentrated cluster will correspond to a smaller ellipse. In the PDM space, the cluster dispersion can be directly related to the encoder noise. Increasing the noise will increase the dispersion and thus the respective ellipse size.

As it was introduced in Section 5.2, using spatial coordinates for the analysis emphasizes the last part of the trajectories, as the variations are summed all along them. To avoid this problem, the angles between the trajectory steps were used for the analysis, in a similar way to the Correlated Random Walk model (Section A.9.3). As the sampling method produces points more or less equidistant, the distance between the points (step length) was not used. The resulting PDM analysis is shown in Figure 6.21(b). As the noise is constant all along the trajectories, the result is similar to the one performed with the spatial coordinates. For this kind of experiment, both data can be used.

A similar experiment was performed. The encoder noise was kept to zero and three values of noise (5%, 10% and 20%) where used to represent the wheel slip. The same method as previously was used to sample the trajectories with 200 points. The angles between the steps were then computed and used for the PDM analysis. The resulting ellipses of unitary Mahalanobis distance are shown in Figure 6.22. In this case, the variations are no more concentrated into the corners. As the wheels slip, the segments are no more perfectly straight. However, the resulting analysis is quite similar: the noise is centered and thus the resulting ellipses have more or less the same center. Moreover, increasing the noise increases also the ellipse size.

6.2.2 Modification of the wheel radius

To induce a non-centered variation to the trajectories, the left wheel radius was modified by $\pm 1\%$ and $\pm 0.5\%$, producing a continuous drift. For this experiment, the encoder noise and the slip noise were kept at 5%. The resulting trajectories are shown in Figure 6.23. The trajectories were also sampled with 200 points using the method based on the trajectory length. It can be observed that a modification of 1% of the wheel radius results in very large spatial deformations.

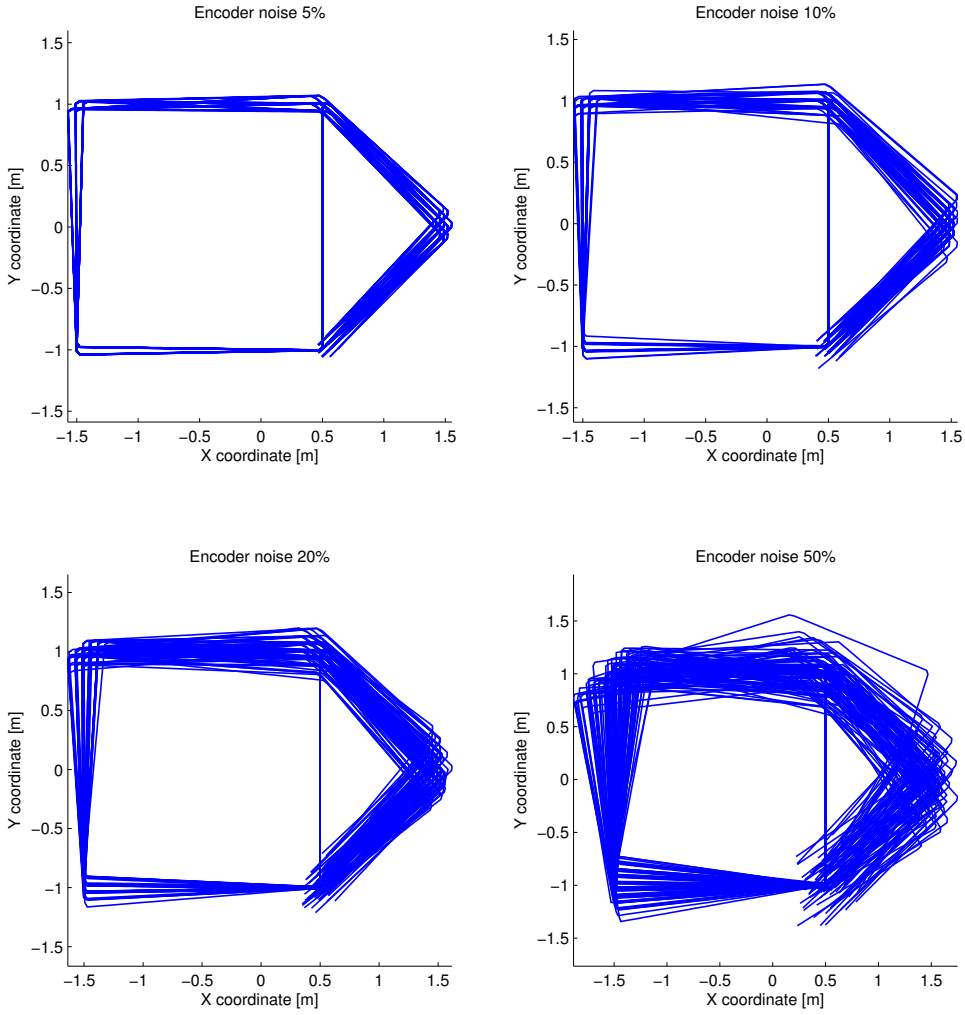
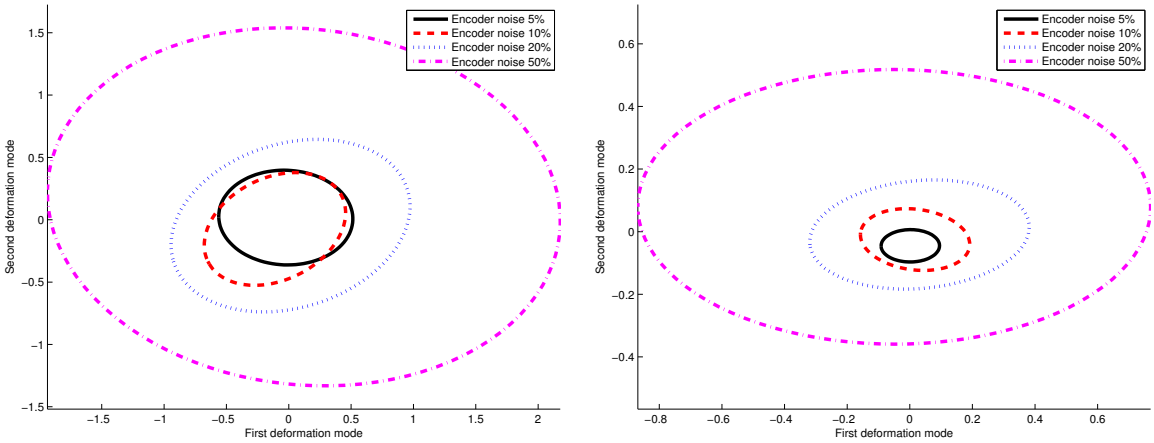


Figure 6.20: 100 trajectories generated in simulation for four different values of encoder noise (5%, 10%, 20% and 50%). As the encoder is used only for the 5 turns and for stopping the robot at the end, the straight parts remain without noise



(a) Spatial Coordinates

(b) Angles between sampled points

Figure 6.21: PDM analyses of the 4x100 trajectories presented in Figure 6.20. On the left, the spatial coordinates of the sampled points were used for the analysis. As the noise is centered, all the trajectory clusters will have approximately the same center. For a clearer visibility, the 100 trajectory points were not plotted and are represented by the ellipse of unitary Mahalanobis distance. Increasing the noise increases the variance of the trajectory points in the PDM space and thus the size of the corresponding ellipse. On the right, the angles between the steps, like in the Correlated Random Walk model, were used for the analysis. As the noise is uniform on the whole trajectory, using this data instead of the spatial coordinates does not really influence the PDM analysis

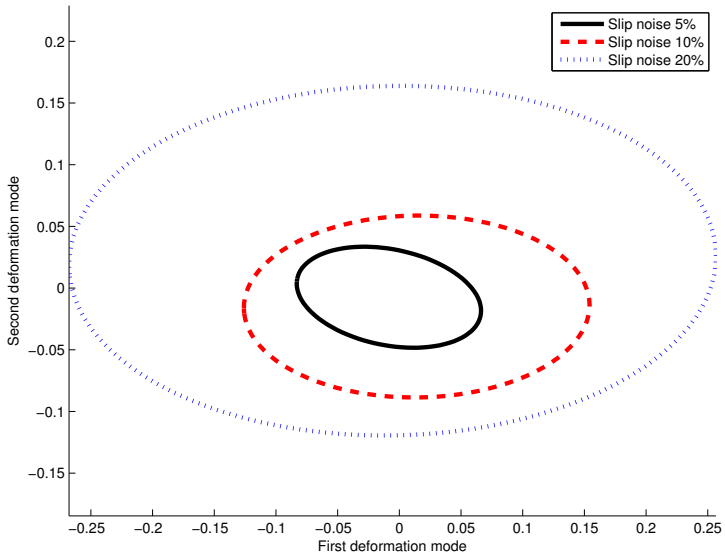


Figure 6.22: PDM analysis of 100 trajectories for three values of noise representing the wheel slip (5%, 10% and 20%). The trajectories were sampled with 200 points and the angles between the steps were used for the analysis. Like in Figure 6.21, the clusters have approximately the same center and increasing the noise will increase the size of the ellipse of unitary Mahalanobis distance

A PDM analysis of the trajectories was then performed using the spatial coordinates of the sampled points. The location of the trajectories in the PDM space is shown in Figure 6.24(a). Contrarily to the previous experiments, the trajectory clusters are clearly separated, as the variations produced by the different wheel radii are not centered.

A PDM analysis was also done with the angle between the steps. Figure 6.24(b) shows the location of the trajectories in the resulting PDM space. For this analysis, the encoder and slip noises are clearly separated from the modifications of the wheel radius. The second axis of the PDM is clearly differentiating the radii when the first axis is focusing on the simulated noises. In this case, using the second dimension is more useful as the parameter of influence (radius) is separated.

6.2.3 Error in one corner

All the previous experiments were showing parameters influencing the whole trajectory. They were global parameters of influence. To show the effect of a punctual modification, an angular error was introduced at one of the five corners. This error was 10% of the original turning angle, corresponding to 9 for the first, second third and fifth turns, and 4.5 for the fourth turn. The robot located itself only with odometry. 100 trajectories were generated for each experiment and were sampled with 200 points using the method based on the trajectory length. The resulting trajectories are shown in Figure 6.25. The angular error at the first turn is the same as the one at the fifth turn. However, the resulting spatial variations are much more important. For all experiments, the encoder noise and the noise representing

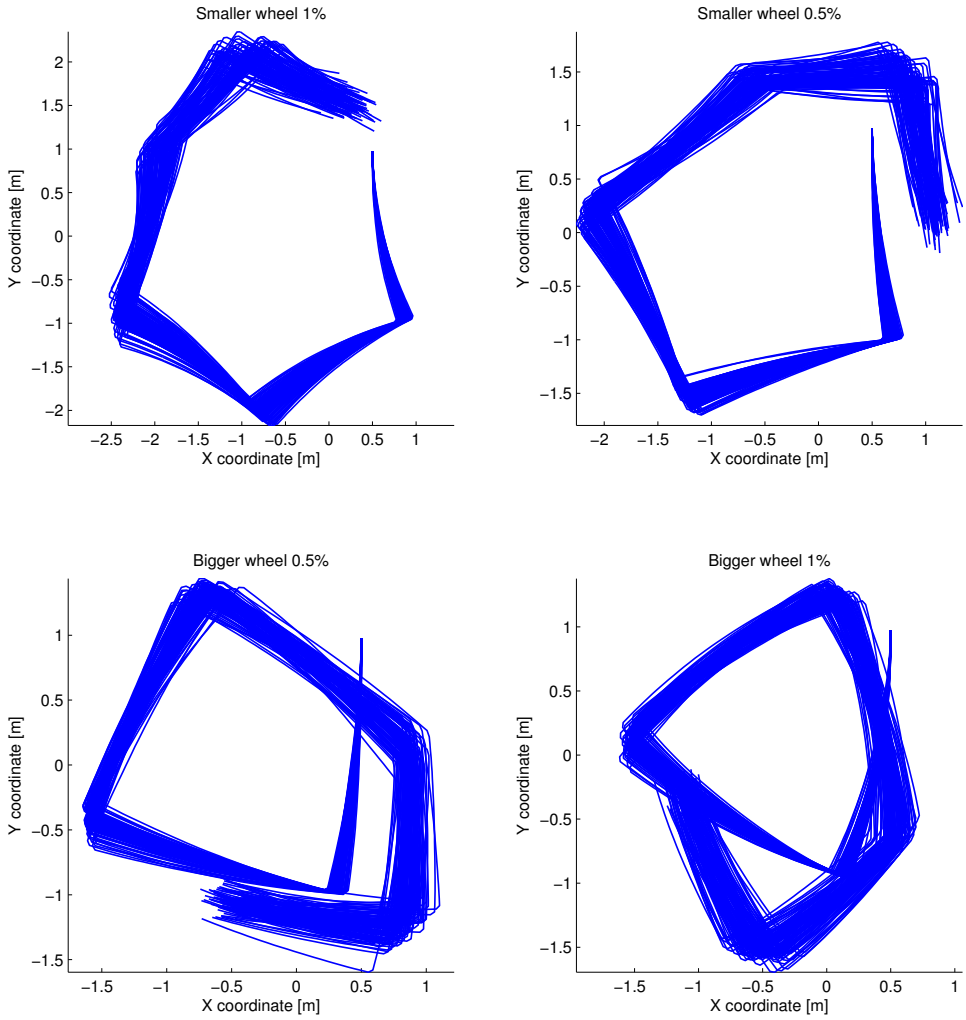
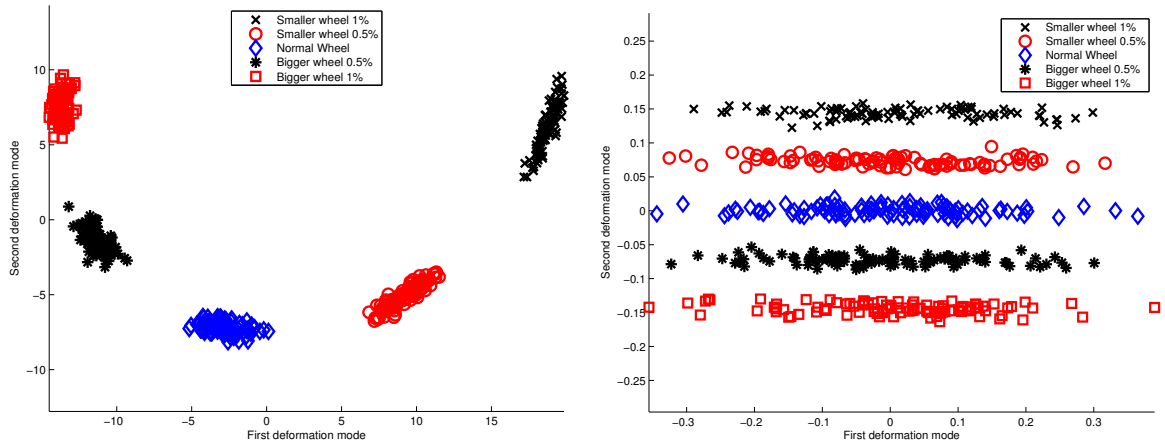


Figure 6.23: The trajectories are generated using the odometry. The diameter of the left wheel is however different from the right, producing a drift in the resulting trajectories. The trajectories were generated with an encoder noise of 5% and also with 5% of white noise representing the wheel slip. The trajectories were also sampled with 200 points, based on the trajectory length



(a) Spatial Coordinates

(b) Angles between sampled points

Figure 6.24: PDM analyses of the trajectories presented in Figure 6.23. On the left, the spatial coordinates were used for the analysis. The trajectory clusters are clearly separated in the PDM space, as the drift induced by the wheel diameter modification produces non-centered variations. On the right, the angles between the steps were used for the analysis. Thus, the variations caused by the wheel diameters are separated from those produced by the encoder and slip noise. In this particular case, the variations induced by the wheel diameters are not the most important and are thus represented by the second axis of the PDM

the wheel slip were kept at 1%.

The spatial coordinates of the trajectories were then used to perform a PDM analysis. Figure 6.26(a) shows the location of the trajectories in the PDM space. Even if the clusters corresponding to an error in the first, second and third corners are separated, the last two clusters are intermingled with the unmodified trajectories. In the spatial PDM, the modifications of the first corner have much more influence than the modification of the last corners. To focus on the modification amplitude and not on its location, it is useful to perform the analysis on the angles between the steps. Figures 6.26(b) shows the location of the trajectories in the first two modes of the resulting PDM space. In this space, all the clusters are more or less equally separated from the unmodified trajectories, exception made of those modified at the fourth corner. The fourth corner measures only 45 and compared to the other corners, the modification of 10% is thus twice smaller. As a result, the distance to the cluster of the unmodified trajectories is also about the half.

As a last example, a similar setup was built where the house is drawn using a GPS instead of odometry. The GPS had a resolution of 1cm and a white noise of 1% was added to the motor commands to represent the wheel slip. We implemented another controller to drive the mobile robot, whose commands

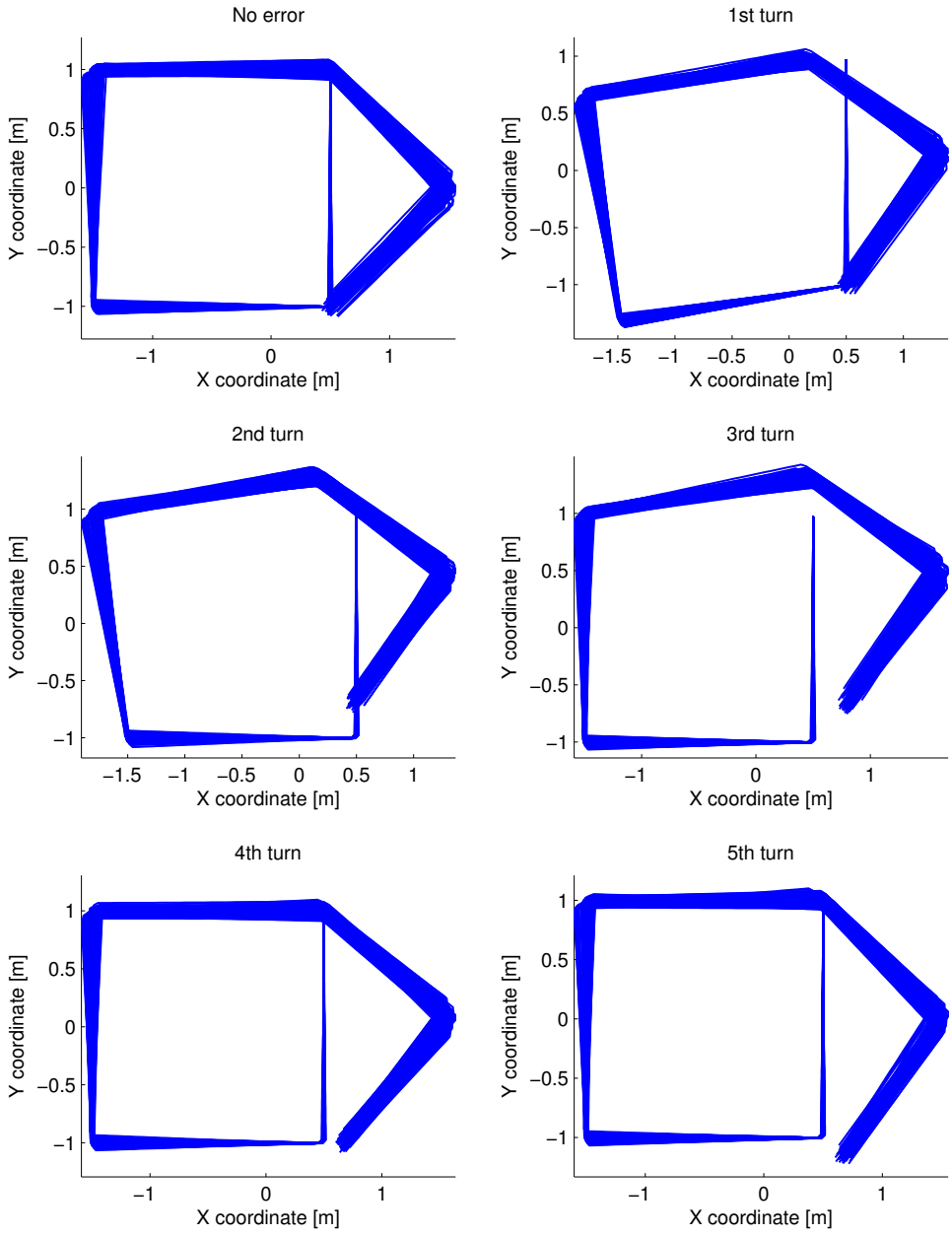
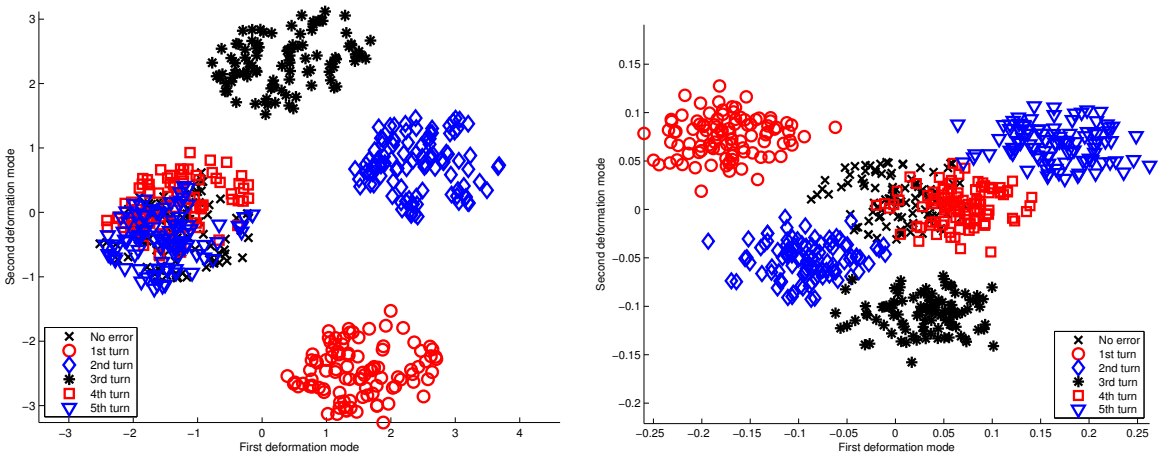


Figure 6.25: A punctual error was added in one of the five corners of the trajectories. The error was of 10% of the original angle, thus approximately 9 for the first, second, third and fifth turns and 4.5 for the fourth turn. As the trajectories were generated using only odometry, an error at the first turn will have the biggest spatial impact. 1% of encoder noise and of slip noise were used for the simulation



(a) Spatial Coordinates

(b) Angles between sampled points

Figure 6.26: PDM analyses of the trajectories of Figure 6.25. On the left, the spatial coordinates were used for the analysis. As an error in the first turn induces more spatial variations, the corresponding cluster is the farthest from the unmodified trajectories. Moreover, the error added in the last two corners are difficult to differentiate, as the variations produced are drowned out by the simulated noise (encoder and wheel slip). On the right, the angles between the steps were used for the analysis. All the clusters are clearly separated. The cluster generated with an error in the fourth turn is however closer to unmodified trajectories, as only an error of 4.5, instead 9, was introduced

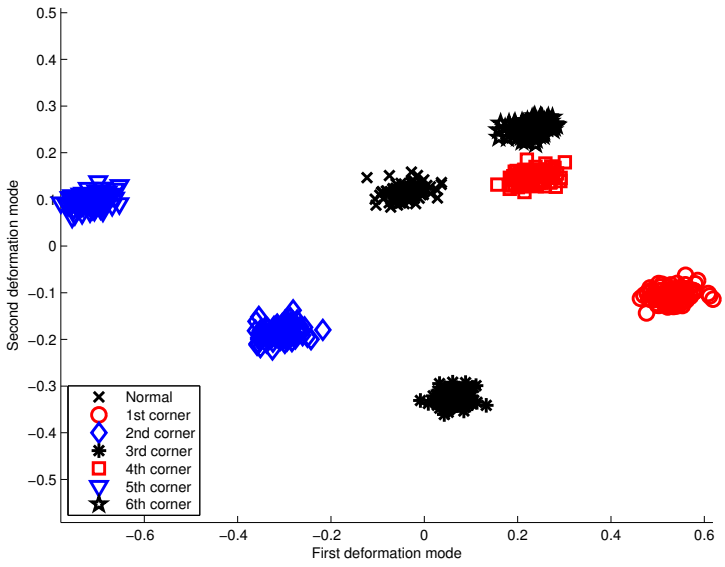


Figure 6.27: The trajectories were generated using a GPS instead of odometry. For each of the six experiments, the targeted position of one corner was moved by 0.1 meter along the X axis. A noise of 1% was used to represent the wheel slip and the GPS had a resolution of 1 cm. This graphics shows the PDM analysis using the spatial coordinates of 7x100 trajectories. As the robot positioning is achieved with a GPS, the corner displacement remains a punctual error and is not accumulated along the whole trajectory. Thus, all the clusters are clearly separated

to the left and right wheels, C_{left} and C_{right} respectively, are described in the following equations:

$$C_{left} = V + P \cdot \Delta_\theta, \tag{6.2}$$

$$C_{right} = V - P \cdot \Delta_\theta, \tag{6.3}$$

$$\tag{6.4}$$

where Δ_θ is the angular error toward the next waypoint, V is a constant value equal to the average velocity, and P is a constant value. Δ_θ, C_{left}, and C_{right} were evaluated every 32 ms. This description corresponds to a closed-loop and proportional controller. When the distance to the targeted waypoint was below 1 cm, the following waypoint was assigned as the new goal. To add some non-centered modifications, each waypoint was moved by 10 centimeters along the X axis, one at a time. As in the previous experiments, the trajectories were then sampled with 200 points, using the method based on the trajectory length. As the robot was driven by a GPS, the error was no more cumulated along the trajectories. Therefore, a PDM analysis based on the spatial coordinates is sufficient to separate them. Figure 6.27 shows the location of the trajectories in the space of the PDM. Some clusters are less separated, as the induced deformations were not all spatially equivalent.

This section intends to show that trajectories with a drift can also be analyzed with a PDM. However, it is important to use the right data to emphasize the desired differences between the trajectories. Two experiments that are quite similar need perhaps two different kinds of data to be analyzed correctly.

6.3 Conclusion

This chapter has demonstrated the power of the PDM analysis of trajectories with a common frame (with or without a drift). Even though the results were produced in simulation, the parameters of influence and the limitations were clearly studied. They cover hardware, software, and environment modifications, and include also studies of parameters more related to the analysis, such as the sampling methods, the number of selected points and the kind of data used. The link between the space of the deformation modes and the space of the trajectories was precisely described, and the techniques were presented so that the PDM results were easily mapped into trajectory differences. The inter-cluster distance was also described as a good measure of trajectory differences. This chapter showed also that the data used (spatial coordinates, derivatives, angles,...) need to be appropriately chosen, in relation to the differences to be analyzed.

As the trajectories were generated in simulation, the different parameters potentially influencing the resulting trajectories are easy to control. The same perfect decoupling of the influences is however difficult to obtain in real experiments. Some parameters are perhaps not easily controllable and thus, they produce undesired variations. Consequently, the resulting analyses will not be as clear as those presented in this chapter.

To demonstrate that the observations made in this chapter are still reproducible in reality, the next chapter will present analyses of real trajectories.

Chapter 7

Real case studies of PDM analyses

This chapter introduces four case studies using real robots. The first aims to classify the trajectories of a mobile robot driven by two different controllers. The second case study shows how the inter-cluster distance can be used to measure the difference between simulated and real trajectories. Thus, the simulation parameters can be optimized to make the simulation closer to the reality. The third case study shows that the classification can also be achieved with trajectories not generated on a circuit. Finally, the last case study shows that the analysis can be ported to a non-robotic agent: a skier and its trajectories.

7.1 PDM analysis of a mobile robot driving on a circuit

In this section we want to introduce the PDM and the inter-cluster distance as tools to classify trajectories of a mobile robot, generated with different behaviors. These results have been presented at the IEEE/RSJ 2007 International Conference on Intelligent Robots and Systems [93].

7.1.1 Case study

To demonstrate the usability of our method to separate trajectories generated with two different behaviors, we choose a simple case study. However, our method can be generalized to similar problems and any type of trajectory as long as the set of sampled points has the same size for each of them.

The arena used for our experiments was $1.4\text{m}\times1.2\text{m}$. On it, we built two closed walls in the shape of a simple track. Figure 7.1 shows the setup. A miniature differential-drive robot, the e-puck [94], was made to drive continuously around this circuit. The e-puck is endowed with eight proximity sensors (Figure 7.3), and a more thorough description of its controllers can be found in Section 7.1.1.1. To extract the robot trajectories, an overhead camera was fixed above the arena. SwisTrack [95, 96], a video-based tracking system, was used to compute the agent's position. For the tracking calibration, the optical system was represented with a second order model; the calibration matrix was computed by Least Squares on a known pattern covering the main portion of the arena. This calibration matrix was then used to transform image coordinates into real world dimensions. The average calibration error was less than 5 mm for the pattern points.

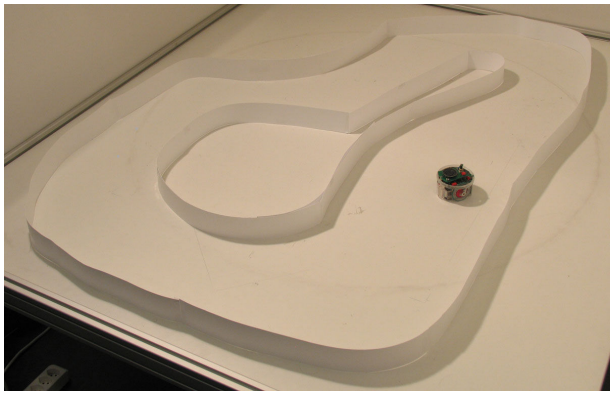


Figure 7.1: The real setup used for the experiments. We can see the e-puck robot and the circuit walls

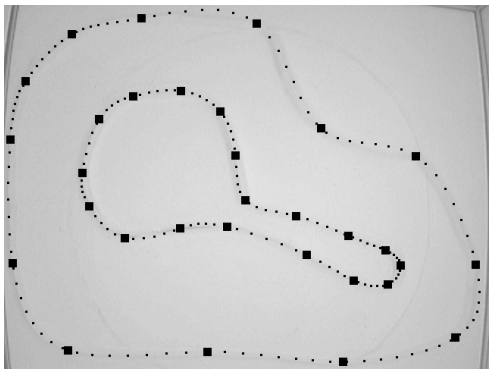


Figure 7.2: Approximation of the circuit walls by multiple b-splines. The separation between the splines are indicated with black squares

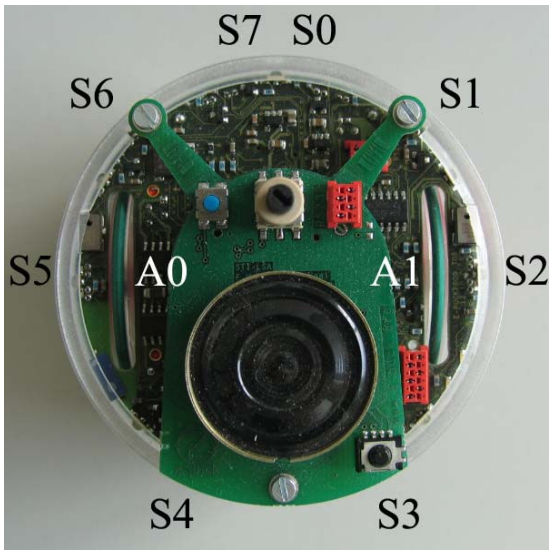


Figure 7.3: Top view of the e-puck robot with 8 sensors (S0, . . . ,S7) and two actuators (A0,A1). The front of the robot is facing towards the top of the image

7.1.1.1 Robot controllers

As possible concrete examples, two different controllers were implemented to drive the e-puck robot. The first controller was rule-based ("If sensor activation is greater than a threshold, turn in the opposite direction of the obstacle."). The second was a Braitenberg controller continuously adjusting the robot speed as a function of its proximity sensor readings. In both cases, only the six frontal sensors were used. A mathematical description of the controllers can be found in Table 7.1. The two controllers are slightly different from those used with the simulated Khepera robot (Table 6.1). Figure 7.3 shows the e-puck robot and the position of its sensors.

Both controllers were implemented identically in the simulator and on the real e-puck. In both cases, the robot moved continuously within the circuit (clockwise), and each lap was extracted as an individual trajectory. Since a lap did not begin and end at the same point, initial conditions are random, and variability was thus added to the trajectories. As the first lap was much influenced by the initial position of the robot, it was always removed from the analyzed data.

7.1.1.2 Trajectory sampling

In order to apply the PDM method, each trajectory must be sampled with the same number of points. To fulfill this requirement, we used the sampling technique presented in Section 4.1.2.2. The trajectories are sampled with gates, as orthogonal as possible to the b-spline approximations of the two circuit walls and with an equal distance between the gate centers. A sufficient number of gates (100) was used for all our experiments. Figure 7.4 shows the sampling gates, the b-spline approximations of the walls and a sampled trajectory.

Table 7.1: Description of the two controllers used for the experiments

Controller 1	Controller 2
If $\sum_0^2 S_i > T \Rightarrow \{^{A_0=-V}_{A_1=V}$ Else if $\sum_5^7 S_i < T \Rightarrow \{^{A_0=-V}_{A_1=V}$ Else $\{^{A_0=V}_{A_1=V}$	$S_r = \sum_0^2 S_i$ $S_l = \sum_5^7 S_i$ $A_0 = V \cdot (1 + K \cdot (S_r - S_l))$ $A_1 = V \cdot (1 + K \cdot (S_l - S_r))$
$S_0, \ldots, S_7$ are the robot sensors as shown in Figure 7.3 (back sensors S_3 and S_4 are not used in either of the controllers) A_0 and A_1 are the robot actuators as shown in Figure 7.3 T is a constant threshold value V is a parameter modifying the robot's overall speed K is a parameter modifying the robot's reactivity	

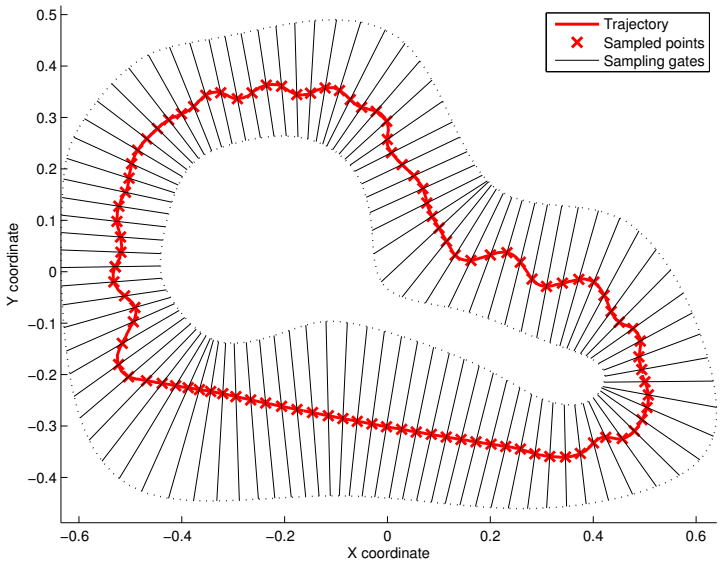


Figure 7.4: Sampling of a trajectory with gates as orthogonal to the walls as possible. The crosses indicated the intersections between the gates and the trajectory that will be used for the analysis

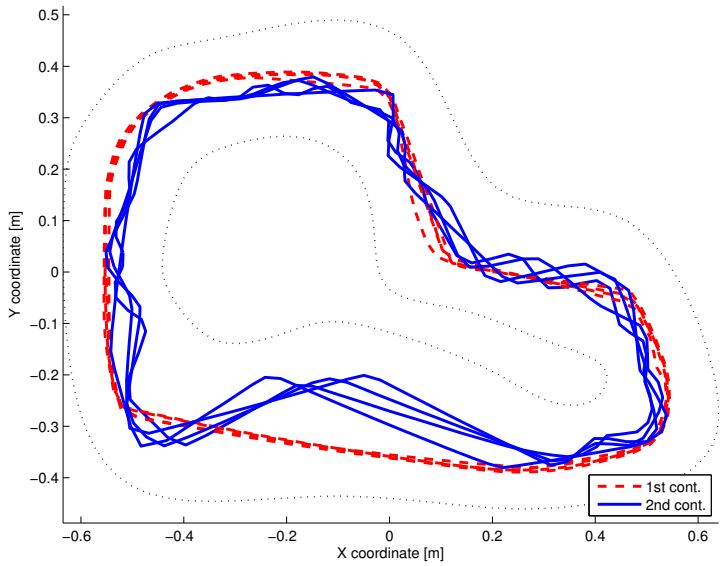


Figure 7.5: Four trajectory samples for each controller presented in Section 7.1.1.1. The main difference between the two controllers lies in the lower part of the circuit

7.1.2 Comparison of the trajectories of two different controllers

Four runs (a,b,c,d) were made on the real setup for each controller presented in Section 7.1.1.1, and for each run, twenty trajectories were extracted with SwisTrack and sampled using the method presented in Section 7.1.1.2. At the beginning of each run, the e-puck's sensors were initialized, but lighting condition changes (sunlight) happened during and between the runs. Figure 7.5 shows four trajectory samples for the two controllers. For this experiment, the two controllers are easily separable in this plot. A PDM was then applied to the resulting dataset.

Figure 7.6 shows the locations of the 160 trajectories in the space formed by the first two modes of the PDM. The separability noticeable in the trajectory plot (Figure 7.5) also exists in the PDM space. Inter-cluster distance, as defined in Section A.5.2, can be used to characterize the differences between the clusters resulting from the different runs with the two controllers. To make a perfect cluster classification, each cluster of a given run must be closer to a cluster of the same controller than to all the clusters of the other controller. The distance, as defined in Equation A.36, from each cluster of a given run to the nearest cluster of the same controller is between 0.9 and 2.0 for the first controller and between 1.8 and 2.2 for the second controller. The smallest distance between two clusters of different controllers is 5.7. Hence we can verify that each cluster of a given run is always closer to a cluster of the same controller, leading to a 100% classification using hierarchical techniques.

7.1.2.1 Classification of trajectories with bivariate Gaussian models

A bivariate Gaussian (Section A.3.1) can be used to model the trajectory clusters in the space of the PDM. As an example, we used a Gaussian model for each controller trajectories. The resulting PDF of the

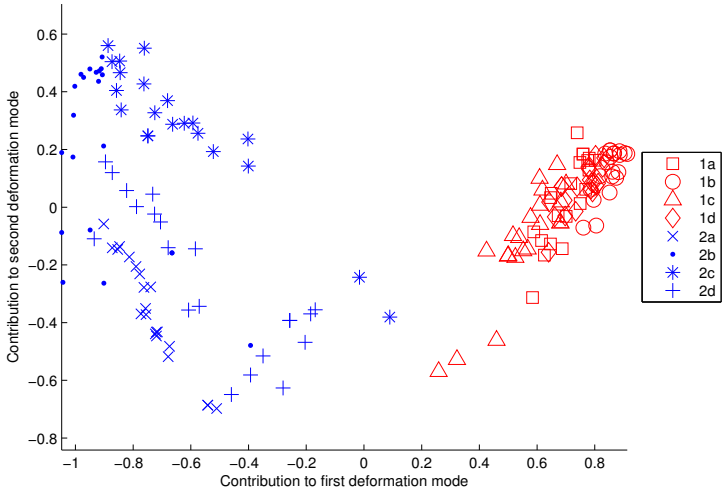
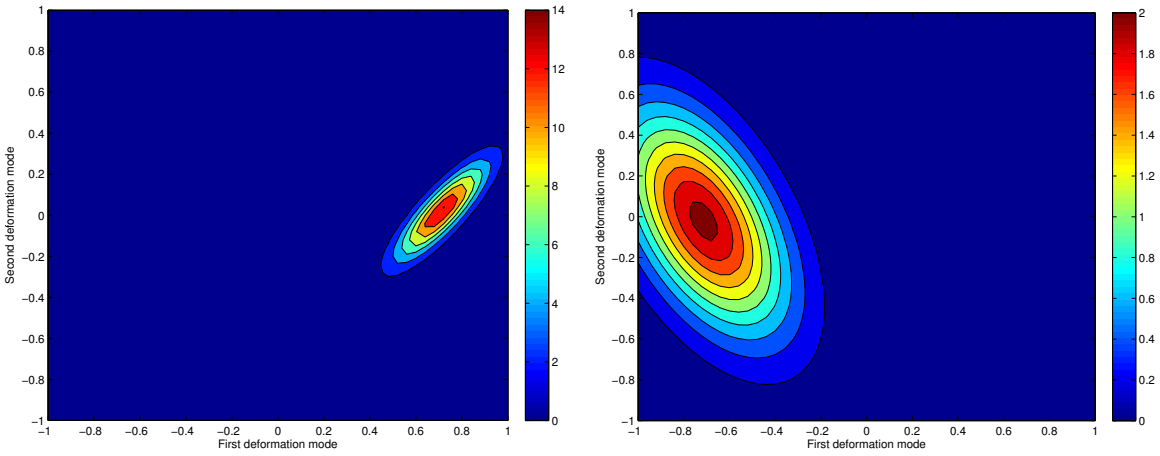


Figure 7.6: The first two modes of the PDM analysis of the two controllers (4x20 trajectories per controller) using the real setup. The different clusters of the two controllers can be easily separated



(a) Model of Controller 1

(b) Model of Controller 2

Figure 7.7: Probability Density Function (PDF) of the bivariate Gaussian Model of controllers 1 and 2, in the PDM space. These PDF can be compared with the trajectory clusters in Figure 7.6

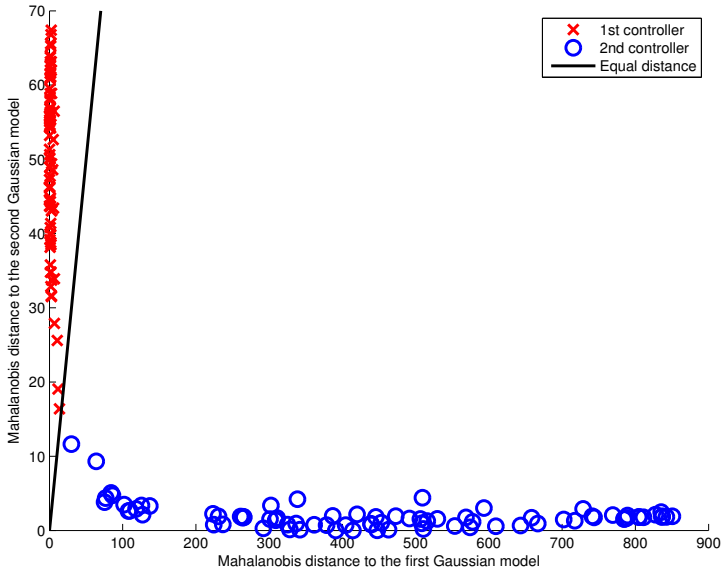


Figure 7.8: Mahalanobis distance from the trajectories of both controllers to the two Gaussian models shown in Figure 7.7. As the trajectories of both controllers are located on the right side of the equal-distance frontier, the classification using Gaussian models built on the PDM is 100% accurate

models are displayed in Figure 7.7. We can observe that the size, shape, and orientation of the Gaussian models in Figure 7.7 are similar to their corresponding clusters in Figure 7.6, with the dispersion of the second model being larger than that of the first model. The Mahalanobis distance (Section A.5.1) can be used to compute the distance from the trajectories in the first two dimensions of the PDM to the two Gaussian just created. Figure 7.8 shows these distances. As the trajectories of both controllers are located on the right side of the equal-distance frontier, the classification using Gaussian models built on the PDM is 100% accurate.

7.1.2.2 Generation of trajectories with multivariate Gaussian models

The two Gaussian models shown in Figure 7.7 were only built on the first two dimensions of the PDM. However, to have better models, we used the first 5 dimensions of the PDM space to create to other multivariate Gaussian models of the trajectories of each controller. The two models were then used to generate random vectors in the PDM space. These vectors were then transfered to the space of the trajectories, using the PDM parameters. All the dimensions after the fifth dimension were chosen as zero. Figure 7.9 shows 4 random trajectories for each multivariate Gaussian model. These trajectories are quite similar to the real ones displayed in Figure 7.5. The generated trajectories are however smoother than the original ones, as only 5 dimensions of the PDM were used.

As each of the real generation run produced trajectories slightly different, it could also be a good improvement to use a Gaussian Mixture Model (Section A.7) to represent the clusters of Figure 7.6.

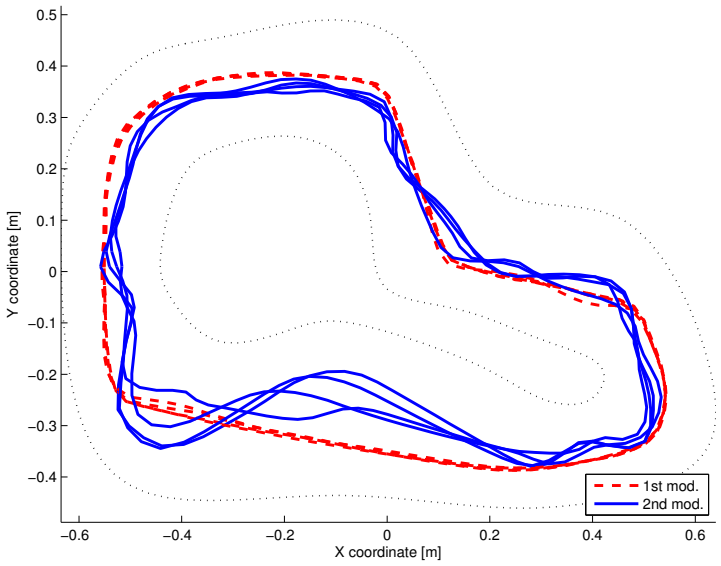


Figure 7.9: Generated trajectories using a multivariate Gaussian model of the first 5 dimensions of the trajectories in the PDM space. A different model was used for the trajectories of each controller displayed in Figure 7.6. The generated trajectories can be compared with the real ones displayed in Figure 7.5

7.2 Simulation quality evaluation

In this section we want to introduce the PDM and the inter-cluster distance as tools to evaluate how tuning simulation parameters improves the simulation quality. Quantifying the similarity of trajectories using a PDM analysis is not only useful to compare two controllers. If the similarity between simulated and real trajectories is quantified, it is possible to evaluate whether or no the modification of specific characteristics are increasing the simulation's faithfulness to reality. These results have been presented at the IEEE/RSJ 2007 International Conference on Intelligent Robots and Systems [93].

7.2.1 Experimental setups

We used the real setup described previously in Section 7.1.1. To recreate the real setup in simulation, we used again Webots [92], a realistic simulator whose faithfulness for other robotic platforms has been demonstrated in several previous papers (see for instance [97]). Only two types of obstacles can be used in Webots to represent the walls, rectangular boxes and cylinders. To reproduce them, we took a picture of the arena with the camera used for the tracking. On this image, we approximated manually each wall with multiple b-splines, trying to minimize the errors and keep a small number of splines. Fig. 7.2 shows the b-splines interpolation of the walls. The black squares represent the connection between the different b-splines, where they share a common first derivative and where their second derivative is null. These multiple b-splines were then transfered into the real world coordinates, using the calibration matrix computed for the real setup. A first order linear approximation of the b-splines was then computed, keeping the maximal error between the b-spline and the segment under 5 mm. From these connected

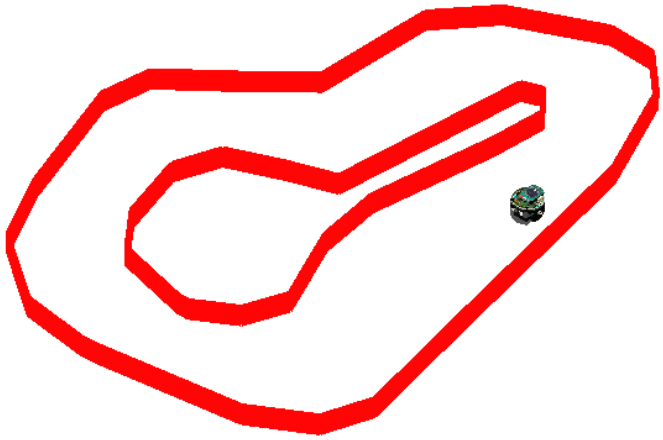


Figure 7.10: The simulated setup used for the experiments that reproduces the real experiment of Figure 7.1

segments, corresponding Webots boxes were then created in the simulated world. Fig. 7.10 shows the resulting setup with the approximated walls and the simulated e-puck.

7.2.2 Simulation faithfulness analysis

We simulated an e-puck robot in Webots and tuned three specific features of the simulation: the proximity sensor model, the wall approximation error, and the amount of wheel slip, represented as a white noise added to the motor commands. These three examples could be extended to any other feature or parameter of the simulation, representing the hardware or software of the robot, or the environment. For each experiment, only one parameter was modified and the default parameters were the improved sensor model, a wall approximation error of 5 mm, and a 10% noise on the motor commands representing the wheel slip.

7.2.2.1 Sensor model

Two different models were used to simulate the sensors of the e-puck: the first was the model delivered with the Webots package while the second was extrapolated from output measurements of the real sensors. Fig. 7.11 shows the two models and the measurements. We can see that the real sensor response is completely non-linear and diverges significantly from the original piece-wise model. It is worth noting that in these experiments we used a single-ray sensor and thus the additional computational cost of the improved version of the sensor model can be neglected. Such trade-off between computational cost and faithfulness could be key with multi-ray models reproducing the real cone of view of a proximity sensor, also implementable in Webots.

To measure the influence of the proximity sensor model on the trajectory faithfulness of the simulated e-puck within a circuit, we reproduced the whole set-up in Webots, using the two sensor models. Only

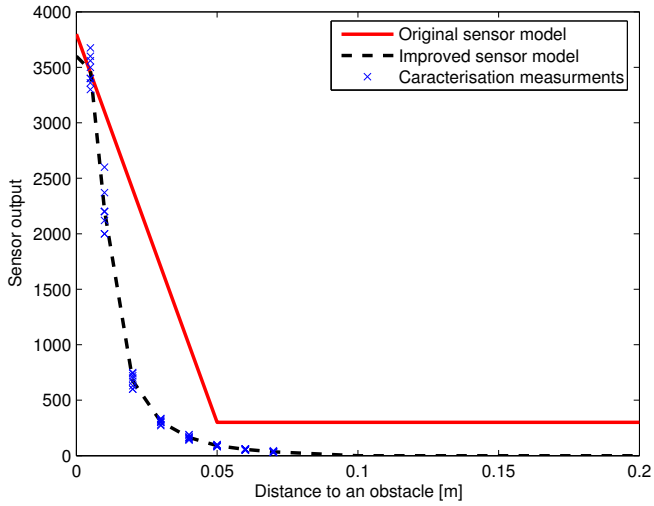


Figure 7.11: Real e-puck sensor output and two candidate sensor models. We can see that the sensor output is non-linear and that the usable range is quite short

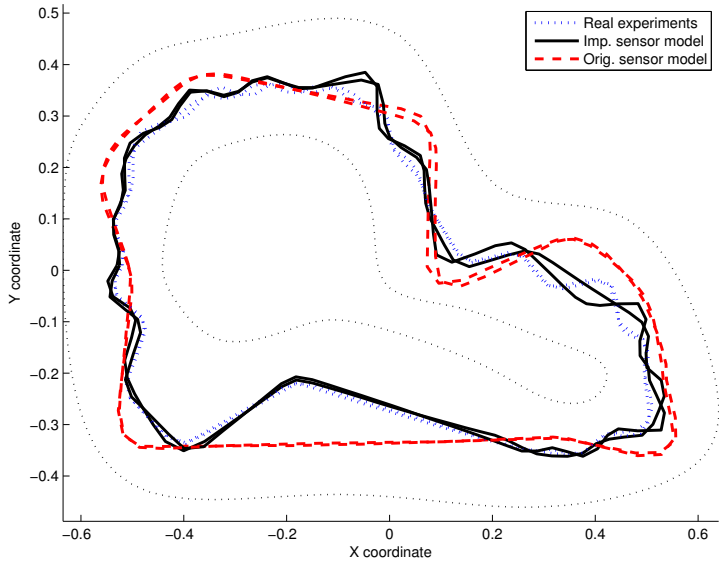


Figure 7.12: Two trajectory samples for the real experiments and simulations done with the original and improved sensor models. The trajectories of the improved sensor model are more similar to the trajectories of the real experiments than the ones of the original sensor model

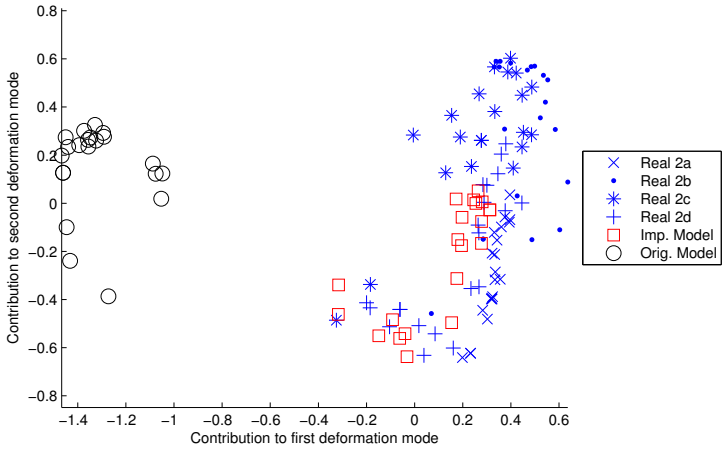


Figure 7.13: PDM analysis showing agreement between simulation (2×20 trajectories) and real experiments (4×20 trajectories), for the two proximity sensor models. The cluster of the improved sensor model is much closer to the four real experimental runs than the one of the original sensor model

Table 7.2: Inter-cluster distances between the clusters represented in Fig. 7.13, resulting from the real experimental runs and the simulations with the two different sensor models

Real experiments	Run a	Run b	Run c	Run d
Improved sensor	2.4	2.3	2.4	0.5
Original sensor	16.3	12.9	9.7	11.0

the Braitenberg controller was used for this purpose (2nd controller in Table 7.1). We extracted 20 trajectories with both sensor models and compared them with the four runs of 20 trajectories we made with this controller on the real setup. Fig. 7.12 shows trajectory samples for the different experiments. All the trajectories were extracted and then sampled as presented in Section 7.1.1.2. Afterward, we applied our PDM analysis to the trajectories. Fig. 7.13 shows the projection of the trajectories in the space formed by the first two modes of the PDM. The modified sensor model shows a marked improvement in the accuracy of the simulation. Trajectories simulated with the improved sensor model are much closer to the real trajectories in the PDM space. This observation can be related to the trajectory plot (Fig. 7.12).

Inter-cluster distance (measured using Eq. A.36) can be used to quantify the improvement in the accuracy of the simulation. Table 7.2 shows the inter-cluster distances between the clusters representing the real experimental runs, and the clusters resulting from the two simulated experiments. Even if the improved simulation is closer to reality and especially to the last run, its cluster remains different from the other three real experiments. Differences between the runs can be explained by changes in lighting conditions (sunlight influences the output of the simple IR proximity sensors). As the simulator does not model these variations, the simulation can not match all four experimental runs at the same time.

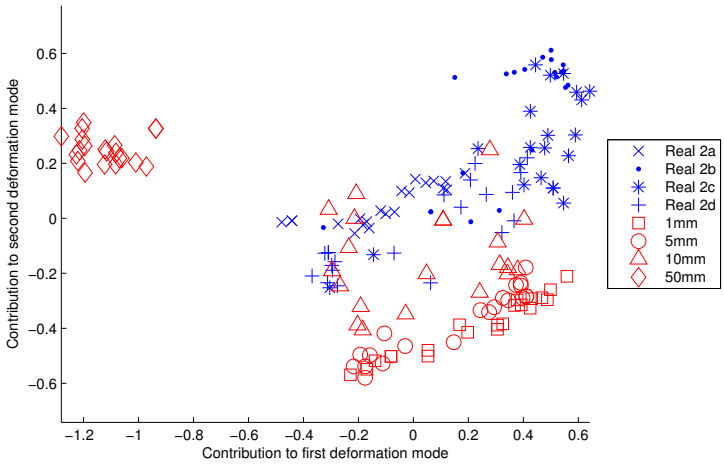


Figure 7.14: PDM analysis showing agreement between simulation (4x20 trajectories) and real experiments (4x20 trajectories) for the four values of wall approximation error. We can see that less precision on the approximation of the walls (50 mm) leads to a cluster farther away from the real experiments. The two clusters corresponding to the highest precision (1, 5 mm) are difficult to differentiate. Even if their shapes is similar to the real clusters, there is a clear offset between the cluster means. Curiously, for the intermediate precision (10 mm), the cluster is closer to the real experiments. However, its shape is less similar to them

7.2.2.2 Wall approximation error

Another parameter of the simulation is the quality of the wall representation. In Webots, the walls need to be represented with segments (rectangular boxes). To create this representation, we used the b-spline model of the wall used for the sampling (Fig. 7.2). Then we set the maximal error between the spline and the segments, and compute them automatically. Four maximal error values were used to create the simulated walls: 1, 5, 10 and 50 mm. Decreasing the maximal error, will increase the number of segments needed to approximate the b-splines. The number of segments needed to create the circuit for each maximal error value were respectively 105, 47, 33 and 15 segments. As the computation time needed to trace rays from the sensors to all the obstacle is a linear function of the number of segments, it is important to keep it as small as possible without decreasing the simulation faithfulness.

To evaluate the influence of the wall approximation on the simulation faithfulness, experiments similar to the analysis of the influence of the sensor models were performed. Twenty trajectories were extracted for each of the four approximation errors, using the Braitenberg controller (controller 2 in Table 7.1). The trajectories were then compared with the four runs of 20 trajectories we made with this controller on the real setup. A single PDM was computed and Figure 7.14 shows the projection of the trajectories in the space of the first two deformation modes. We can see that an approximation error of 50 mm decreases the simulation faithfulness significantly, and that it is nearly impossible to differentiate the clusters resulting from an approximation error of 1 and 5 mm. These observations can be directly linked to the respective inter-cluster distances in Table 7.3.

Table 7.3: Inter-cluster distances between the clusters represented in Fig. 7.14, resulting from the real experimental runs and the simulations with the four wall approximation errors

Real experiments	Run a	Run b	Run c	Run d
Error 1 mm	16.5	6.1	5.2	6.8
Error 5 mm	12.8	5.6	4.6	5.8
Error 10 mm	1.8	2.7	2.2	0.8
Error 50 mm	11.2	10.9	11.2	10.7

From the PDM, it can be easily pointed out that the imperfect modeling of the walls with the b-splines led to an offset in the average trajectory of the simulation compared to the real average trajectory. Curiously, a medium quality of the segment approximation (10 mm) can even lead to a better average trajectory than more precise ones (1 and 5 mm). This offset can also be seen in the trajectory space (Figure 7.15). However, it is important to notice that the cluster shapes for 1 and 5 mm are closer to the fourth run of the real experiment. This means that the trajectory variations are more faithful than for an approximation error of 10 mm. Thus, a measure purely based on the distance between the clusters is not sufficient and a comparison of the cluster shapes (covariance matrix) is also needed when the clusters become too close.

The optimal solution is perhaps more difficult to choose in this case. As the b-spline representation of the walls is not perfect, reducing the error that the next approximation phase induces (polyline approximation) will just emphasize this imperfection. The clear offsets produced by the best approximations (1 and 5 mm) demonstrates the statement. As a result, the optimal solution would be to make a better approximation of the walls with b-splines.

7.2.2.3 Wheel slip representation

A last simulation parameter will be investigated: the noise representing the wheel slip. Two values of noise were simulated: 10% and 100%. Figure 7.16 shows the projection in the PDM first two modes of the 2x20 trajectories of the two simulations and of 2x20 trajectories of the runs c and d of the real experiments. In all cases, the Braitenberg controller (controller 2 in Table 7.1) was used. It can be extrapolated that this noise has hardly any influence on the average trajectory shape. The offset seen before is still there, and only a difference in the cluster shape can be observed for the two noise values. This relative low weight of the noise representing wheel slip on the trajectory faithfulness is an artifact of the circuit scenario: without a sensor-based guidance between the walls, this noise would have had a major impact on the simulation accuracy.

7.2.2.4 Combined analysis of the three simulation features

To quantitatively analyze the relative influence of the different features on the simulation faithfulness, a joint PDM analysis was realized using the trajectory data collected for the previous experiments. We selected a standard simulation using the default parameters for the proximity sensor model, the wall

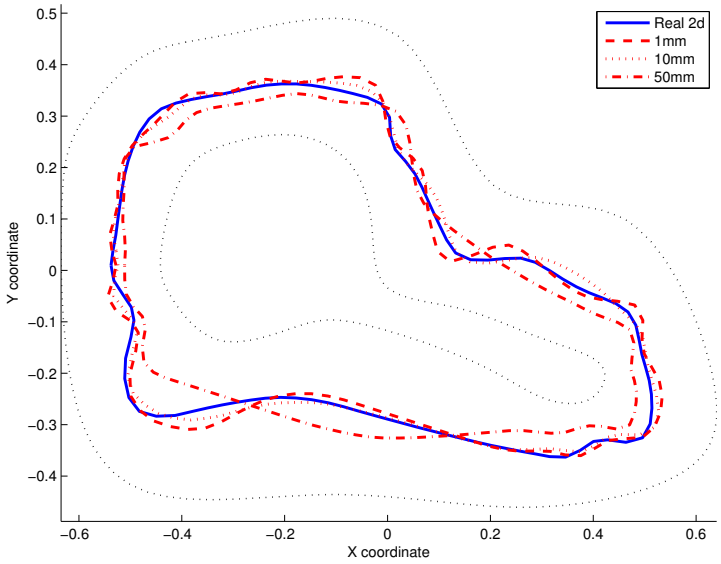


Figure 7.15: Average trajectory for the the real experiment (Real 2d) and for the three simulated runs with a wall approximation error of 1, 10 and 50 mm respectively. We can see that for a reduced precision (50 mm), the average trajectory is really different than the real average trajectory (Real 2d). Moreover, for the highest precision (1 mm), the oscillations are really similar to the reality. However, the middle precision (10 mm) is closer to the reality, even though its oscillations are less faithful

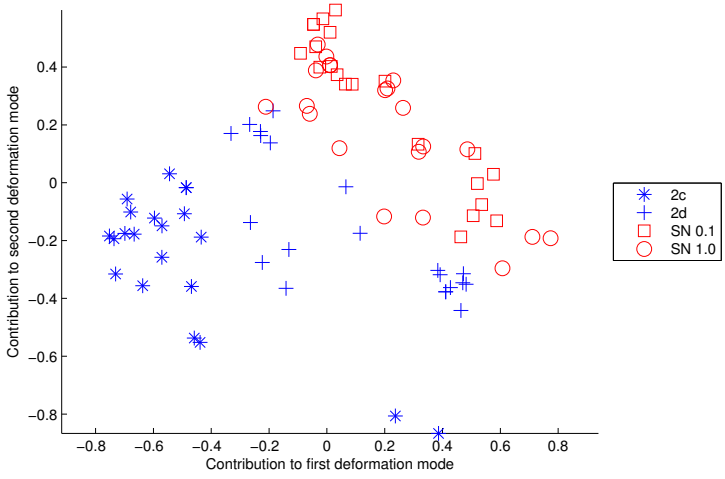


Figure 7.16: PDM analysis showing agreement between simulation (2x20 trajectories) and runs 2c and 2d of the real experiments (4x20 trajectories) for the two values of noise (10% and 100%) representing the wheel slip. We can see that the value of the noise has not a big influence on the trajectories

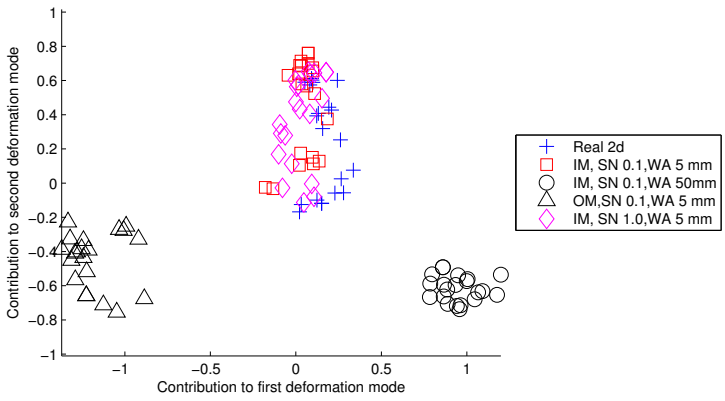


Figure 7.17: PDM analysis of one real experimental run (Real 2d) together with the three simulation parameters: the sensor model (original and improved, respectively OM and IM), the reproduction of the wheel slip (SN) and the wall approximation error (WA)

approximation error, and the wheel slip noise. Then, we chose another value for each simulation characteristic (original sensor model, slipping noise of 100%, and wall approximation error of 50 mm) and selected the corresponding experiment for each of these values. The PDM was then built using 20 trajectories for each simulation and for run 2d of the real experiments. Figure 7.17 shows the projection of the trajectories in the first two dimensions of the PDM. We can see that modifying the wall approximation error or the sensor model considerably influences the quality of the simulation. However, modifying the noise reproducing the wheel slip has a much smaller impact. Therefore, it would be better to improve the previous simulation features instead of matching wheel slip between the real and simulated systems for these particular experimental conditions.

7.2.2.5 Discussion

In the current analysis of the fidelity of the simulation, a number of other features and parameters were not taken into account: the e-puck sensors we used are represented by single rays when in reality they have some finite cone of view. Likewise, actuators in Webots are a simplified representation of the real stepper motors (white noise instead of real non parametric slip/friction effects). Moreover, the box approximation of the walls does not perfectly recreate the complex shapes of the real walls, the complex infra-red reflections are not taken into account, and neither tracking noise nor variable lighting conditions are reproduced. However, even though our method does not facilitate the creation of a more faithful simulation, it allows us to quantify the influences of various simulation design choices that may have a potential impact on the resulting trajectory of a mobile robot. Moreover, our modeling method helps us to evaluate the relative value of these choices in terms of computational requirements versus simulation

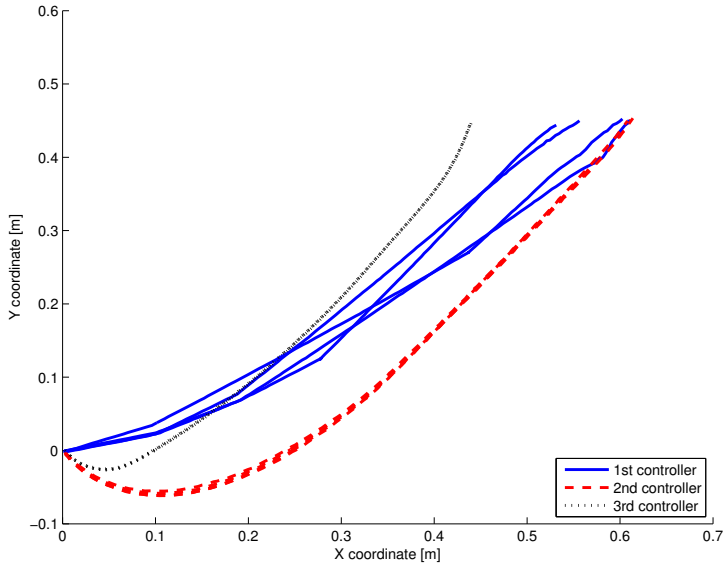


Figure 7.18: Trajectories generated on the setup created during the work of Aïsha Hitz, presented in Section 4.2. Four trajectories for each of the three controllers implemented are plotted. The difference between the resulting trajectory sets is clearly observable. The four trajectories of the second controller are overlapped, as those of the third controller

faithfulness.

The inter-cluster distance, as presented in Section A.5.2, has its highest value when evaluating the relatively big differences between trajectory sets. However, when the clusters are too close, a multivariate extension of the Kolmogorov-Smirnov test (Section A.3.3) would be more suitable than the inter-cluster distance or Hotelling's T^2 statistics, as it takes into account covariance matrix differences.

Finally, the PDM analysis presented here is purely spatial. The temporal aspect of the trajectories was not considered, making the analysis easier to understand. However, the temporal value of the sampled points can be easily added to a PDM, leading to a spatio-temporal analysis.

7.3 Light tracking

Aïsha Hitz, during her semester project [90], worked on the trajectories of an e-puck mobile-robot searching for a light. The setup which she developed, and the four sampling techniques used hereafter have been presented in Section 4.2. Three controllers were implemented to drive the robot on the setup, resulting in three different behaviors. Figure 7.18 shows four sample trajectories for each controller. The trajectories of the second and third controllers are quite repeatable, when those of the first one are variating a lot. The robot was always starting from the same position and with the same orientation. The acquisition of

the trajectories was manually started and automatically stopped when the measured distance between the robot and the box containing the light was below a given threshold. Thus, all the acquired trajectories start at the lower left part of the plot and finish at the upper part of the plot.

To sample the trajectories, four techniques were used:

- Time-based: all the sampled points are separated with the same time interval,
- Trajectory length: all the sampled points are separated with steps of same trajectory length,
- Circle: all the trajectories are sampled at their intersections with concentric circles,
- Parallel lines: all the trajectories are sampled at their intersections with parallel lines.

All the trajectories were reduced to 20 sampled points. The sampled points of fours trajectory examples are shown in Figure 7.19 for each controller. Differences between the sampling techniques are clearly observable. For the same controller, the time-based method produces sampled points which are not always close to each other. Ignoring the first controller trajectories, the method based on parallel lines looses a great part of the information contained in the first curvature of the trajectories. However the four techniques produce the same information.

7.3.1 PDM analyses

The four datasets (one per sampling technique) are made of 60 trajectories (20 per controller) represented by 20 sampled points. A PDM was then built on each of these datasets. For each dataset, Figure 7.20 shows the location of the trajectories in the space of the PDM. The separation between the three controllers is really clear. The variability of the first controller is also observable in this plot, contrasting the repeatability of the two other controllers. In this particular case, the four sampling techniques give more or less the same results. The methods based on circles or parallel lines produce more compact and thus more separated clusters. However, the difference with the two other methods is not so important. Even if it was not used here, the inter-cluster distance could be used in this case to measure the difference between the controllers.

Figure 7.21 shows the cumulative energy of the deformation modes of the PDM. For all sampling techniques, the first deformation mode contains the main part of the energy (74% to 90%) and the first two deformation modes contain more than 97% of the energy. Thus, reducing the problem to two dimensions will hardly reduce the information hold in the dataset. This plot shows also that the sampling methods based on circle and parallel lines are better than the other two, as they contain more variance in their first deformation modes.

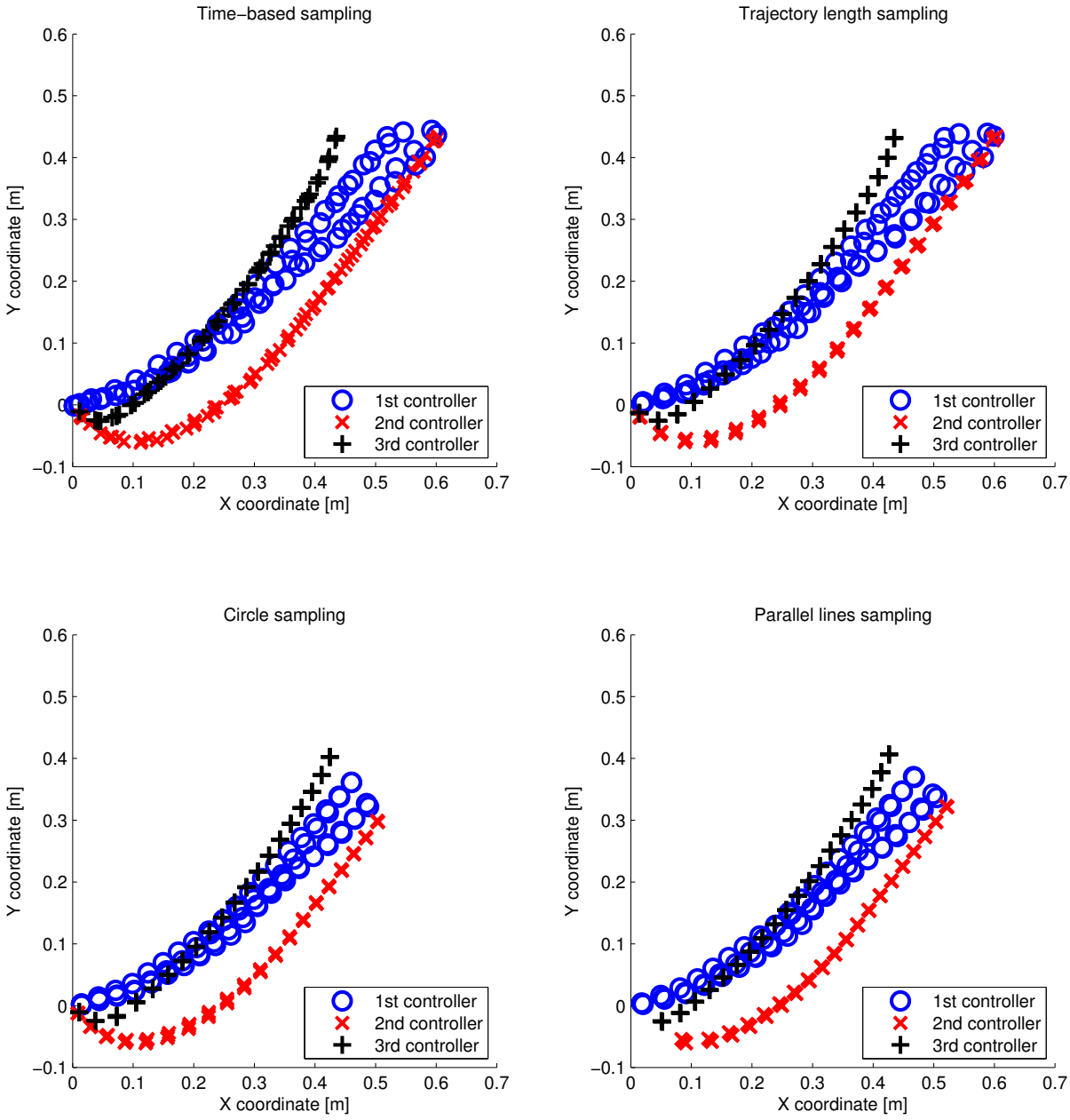


Figure 7.19: For the four sampling techniques used, the 20 sampled points of the 12 trajectory examples (four per controller) are shown. The four sampling methods produce datasets containing more or less the same information. However, the time-based technique produces sampled points which are not close to each other for the same controller. When using parallel lines, the first curve of the trajectories is not well represented by the sampled points, especially for the second controller

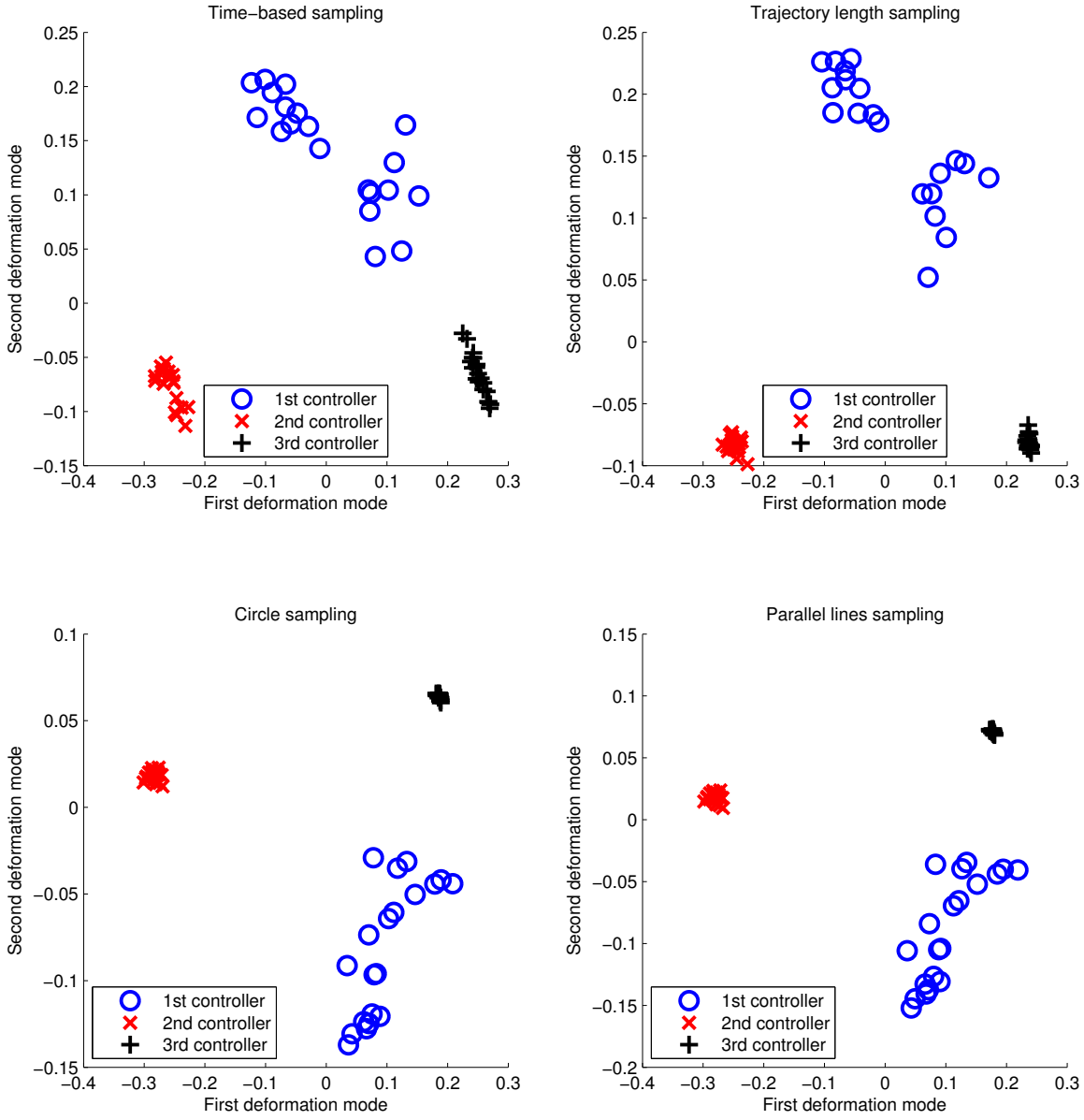


Figure 7.20: PDM analyses of the four trajectory datasets (one for each sampling technique). The separation between the three controllers is clearly observable. There is however not a lot of differences between the results obtained from the four different datasets. In this case, no sampling technique is clearly better than the others

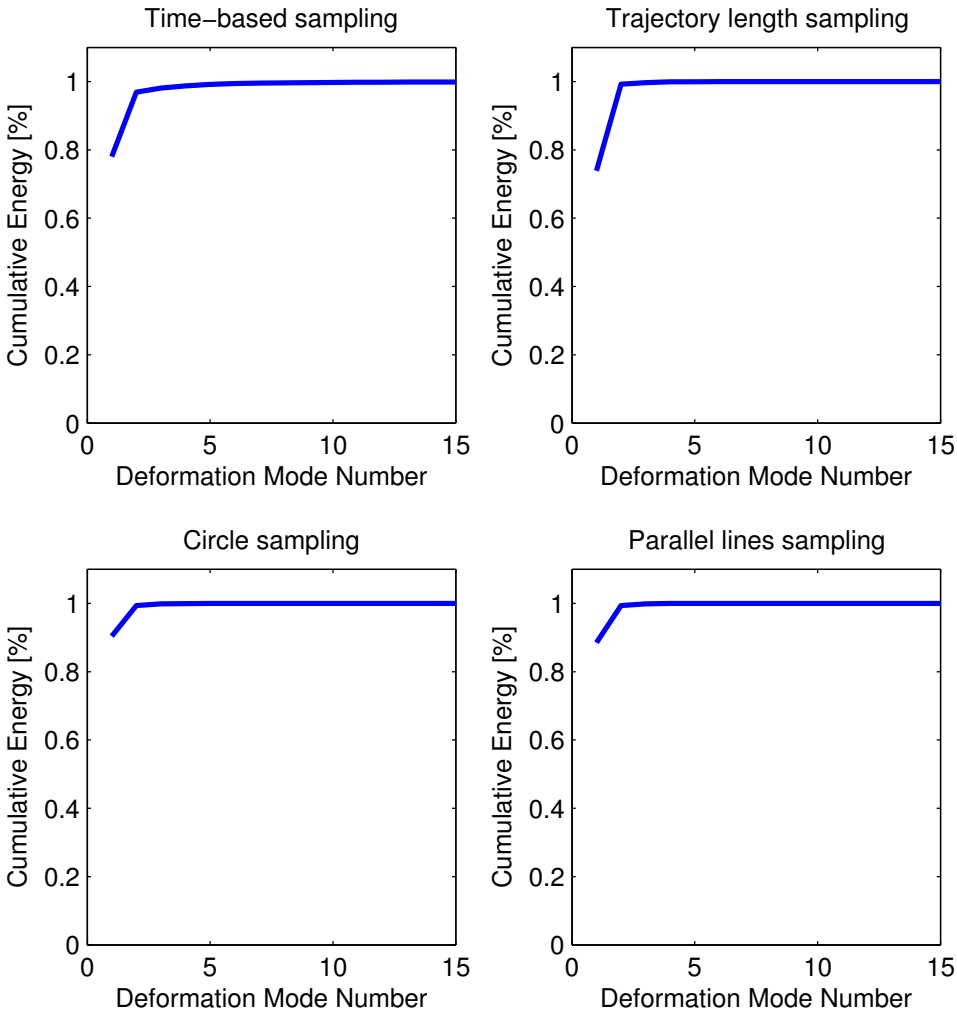


Figure 7.21: Cumulative energy contained in the first deformation modes of the PDM analyses of the four datasets (one per sampling technique). For each dataset, more than 97% of the variance is explained by the first two deformation modes

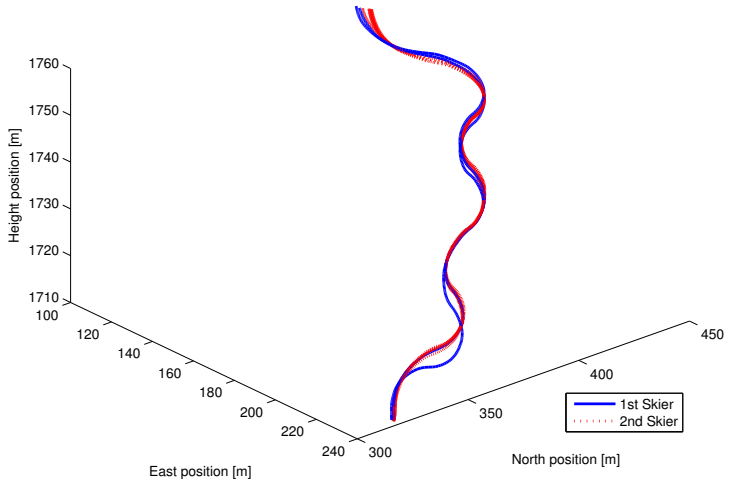


Figure 7.22: 10 skier trajectories (2 for the first skier and 8 for the second). The first and second skiers correspond respectively to Jan Skaloud and a professional skier. The first skier made a deviation from the "standard" trajectories at the lowest part of one of his trajectories

7.3.2 Conclusion

This example presented a new type of setup. However, four new sampling techniques allowed the use of a PDM to analyze the trajectories. In this case, the difference between the controllers is so large, that differences among the sampling techniques are difficult to assess: all are able to separate clearly the trajectories of the three controllers. However, the two methods based on concentric circles and on parallel lines give better results. This example shows how sampling methods can be developed to measure controller differences with a PDM.

7.4 PDM analysis of skier trajectories

Adrian Wägli and Jan Skaloud, worked on the extraction of skier trajectories, using a combination of a single frequency GPS receiver and an inertial measurement unit, both chips being low cost MEMS. In their contribution to the *Inside GNSS Magazine* [1], they compared also the trajectories, using a similar sampling technique as the one presented in Section 4.1.2.1 to compute comparable trajectory points.

Adrian Wägli provided us with 10 trajectories recorded during their experiments. These trajectories were performed by two different skiers: Jan Skaloud and a professional skier. Figure 7.22 shows the 10 trajectories. Two were performed by Jan Skaloud and eight by the professional skier. In this plot, One of Jan Skaloud trajectory clearly shows a major spatial deviation from the "standard" trajectory at the lower

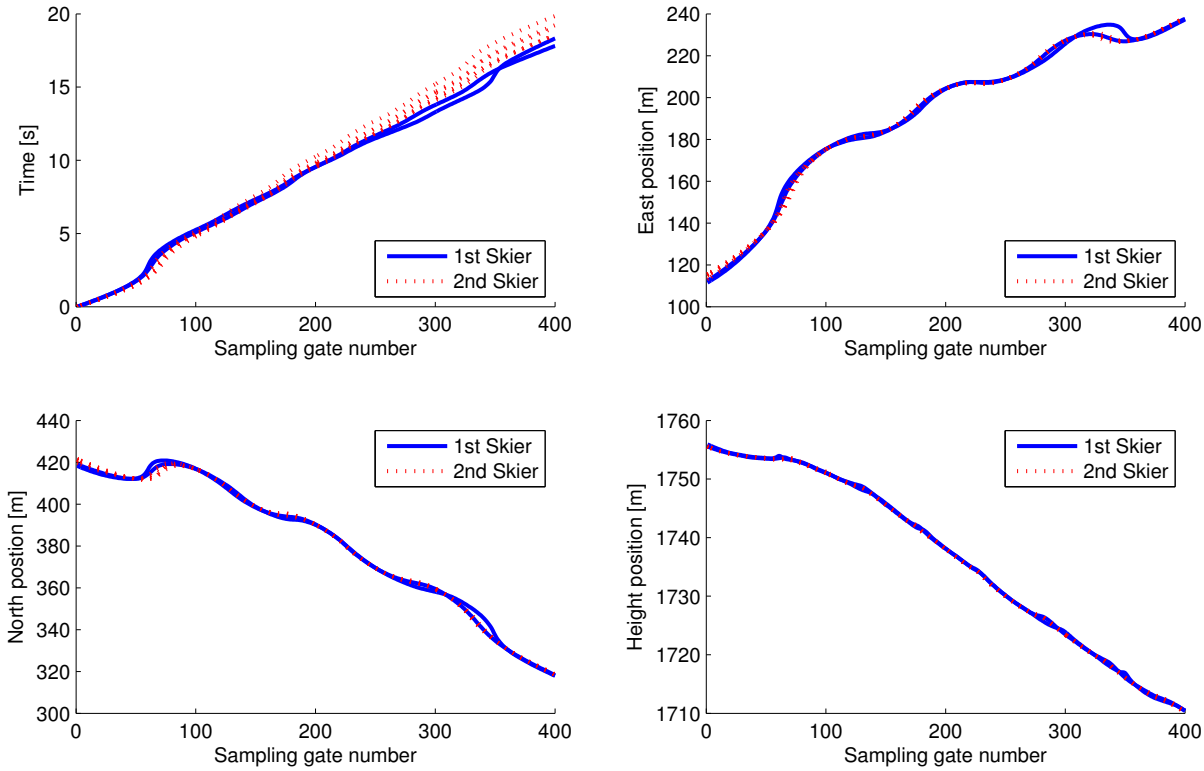


Figure 7.23: Time, north, east and height profiles of the 400 sampled points. The deviation of the first skier is also visible in the eastern and northern profiles. The height profiles are nearly all the same, as the height is constrained by the trail morphology. Finally, the first skier (Jan Skaloud) is faster than the second skier (professional skier)

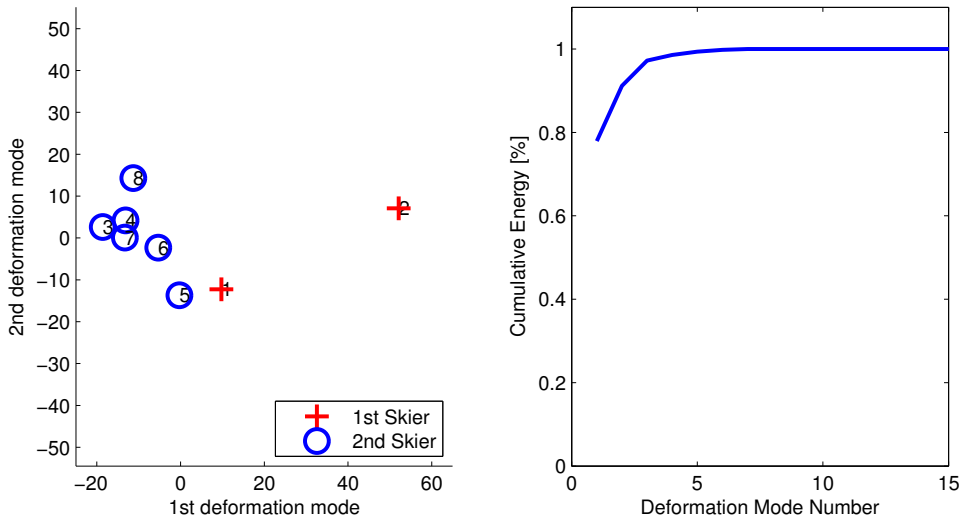


Figure 7.24: On the left, positions of the trajectories in the space of the PDM modes. The trajectory during which the first skier made a deviation (Trajectory 2) is clearly an outlier. The trajectories are also clearly separated with the first and second skier trajectories lying on the right and on the left respectively

part.

The trajectories were then sampled using 400 parallel planes, using the method described in Section 4.2.2.3. The resulting temporal and spatial profiles are shown in Figure 7.23. The deviation of Jan Skaloud is visible in the eastern and northern profiles. As the height of the trajectory profile is highly constrained by the trail morphology, no deviation is visible. Looking at the temporal profile, it can be assessed that the first skier (Jan Skaloud) is faster than the second skier (professional skier). After the first part (sampling gates 1 to 100), his two trajectories are always lower on the temporal profile plot.

7.4.1 PDM spatial analysis

To analyze the trajectories, we made a PDM of the spatial profiles of the trajectories (north, east and height). Figure 7.24 shows the positions of the ten trajectories in the space of the PDM modes. The deviation made by Jan Skaloud during one of his trajectory (2) is clearly visible. Moreover, the trajectories of both skiers are clearly separated: on the left for the professional skier and on the right for Jan Skaloud. The trajectory cluster of the professional skier is quite compact and can be related to its skiing abilities. On the right of the figure, the cumulative energy is plotted. More than 90% of the energy is contained in the first two deformation modes. However, as the dataset contains only ten trajectories, there are only 9 degrees of freedom and thus 100% of the energy is contained in the first 9 deformation modes.

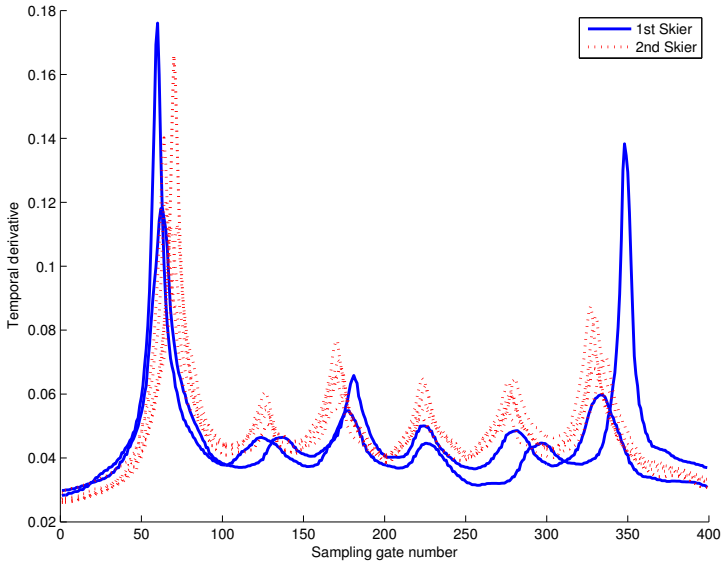


Figure 7.25: Derivative of the temporal profiles (speed).The first skier is slower than the second for the first 60 sampling gates, but then he is always faster, exception made of the deviation during one of his trajectory

7.4.2 PDM temporal analysis

The temporal profiles can be considered as trajectories with a common frame and with a drift. Thus, it is not optimal to compare them directly, as the drift will result in much more variance contained in the last part of the trajectories. Thus an approximation of the profile derivative was computed. It corresponds to the temporal difference between two successive sampling points and can be linked to the skier speed. As there is no sampling gate after the last one, the temporal difference cannot be evaluated there and thus the profiles are reduced to 399 points. Figure 7.25 shows them. The difference of speed between the two skiers is visible. Moreover, the deviation made by Jan Skaloud during one if his trajectory can be observed near the 350^{th} sampling gate.

The results of the PDM analysis of these temporal profile derivatives are shown in Figure 7.26 In this analysis also, the deviation made by Jan Skaloud is clearly visible ($2nd$ trajectory). The trajectory cluster of the professional skier is also here very compact. In this case, more than 80% of the energy is contained in the first two deformation modes.

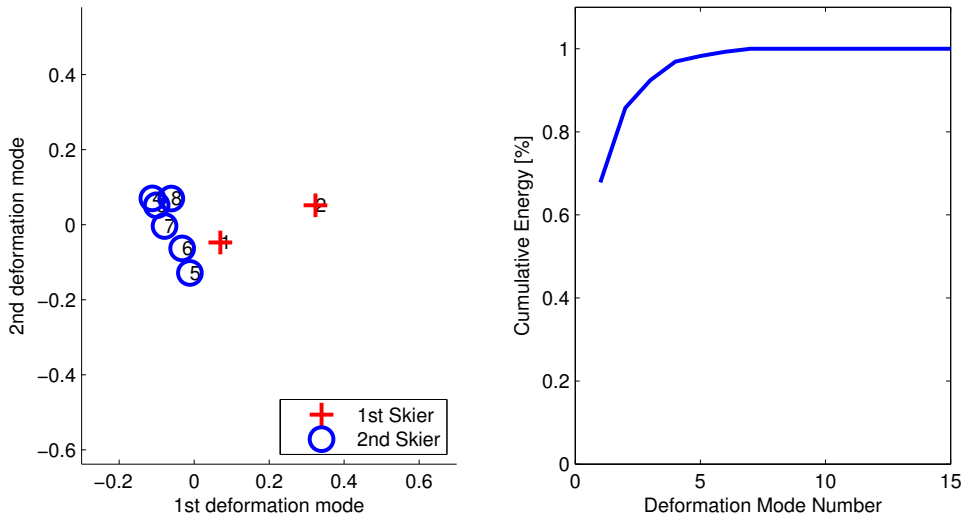


Figure 7.26: PDM analysis of the temporal profile derivatives. On the left, the trajectory positions in the space of the deformation mode. On the right, the cumulative energy of the deformation modes. The two skiers are clearly separable in the PDM space

7.5 Conclusion

This chapter has shown the portability of the PDM method to analyze and compare trajectories generated by agents in the real world. It can be used to differentiate two behaviors, to measure simulation quality, or to compare skier trajectories. The sampling techniques and the parameters must, however, be chosen as a function of the application, but simple solutions can be used and even non-robotic trajectories can be analyzed in this way.

The PDM method is however limited to trajectories with a common frame. Without a common frame, the sequence of sampled points cannot be directly compared. Alternative techniques will thus be presented in the next chapter to analyze this kind of trajectory.

Chapter 8

Comparison of the PDM analysis with other techniques

This chapter introduces two different techniques, the GMM/HMM hybrid model and the Correlated Random Walk model, and compare them with the PDM method described in the previous chapters. Both methods are used to model the trajectories of a real mobile robot, and both their advantages and their limitations are presented.

8.1 GMM/HMM hybrid model

The GMM/HMM hybrid model est is model where each output state of the HMM corresponds to a GMM. It is more thoroughly presented in Section A.8.1. During his semester project [98], Sathyanarayan Anand used GMM/HMM hybrid models to compare trajectories. The results presented in this section are the follow-up of his work.

The GMM/HMM hybrid model was tested with the trajectory data generated for the IROS conference article, presented in Section 7.1. Each model was built with the X and Y coordinates of the sampled points or with the positions on the sampling gates. The dataset contains 201 trajectories generated with the first controllers in four runs $\{34, 41, 63, 63\}$ and 124 with the second controller also generated in four runs $\{34, 33, 33, 24\}$. Each trajectory was sampled with 100 equidistant gates. For the modeling and the learning, we used the Bayes Net Toolbox for Matlab (BNT)[1], implemented by Kevin Murphy. Two models were trained, one for all the trajectories of each controller. The training was performed using the expectation-maximization algorithm implemented in BNT. The maximum number of training iterations was fixed to 80. For all the results presented hereafter, the classification was performed on the same dataset used for learning. No cross-validation technique was used, such as the *leave one out* method.

[1] http://www.cs.ubc.ca/ murphyk/Software/BNT/bnt.html

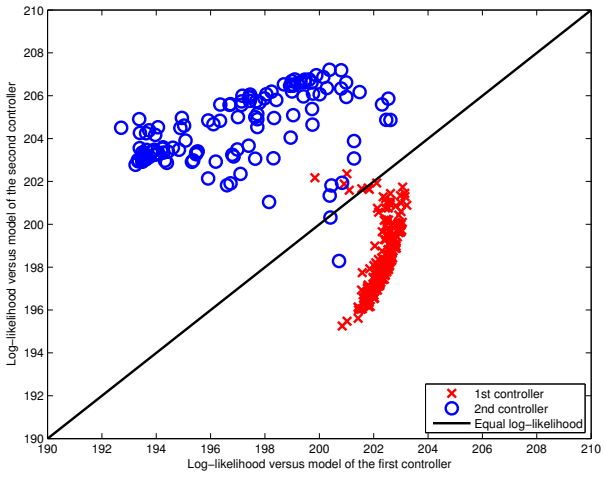


Figure 8.1: Log-likelihood of the trajectories generated with the first and the second controllers versus the two GMM/HMM hybrid models created with the trajectories of only one controller

8.1.1 Model of spatial positions

X and Y coordinates of the sampled points were used to build two GMM/HMM hybrid models of the trajectories, one per controller. The HMM models were full (i.e. from each state there is a non null transition probability to all the other states). The models were made of 10 states and a mixture of 4 Gaussian distributions for each state.

The standard method used to classify elements using GMM/HMM hybrid models is based on the maximum likelihood (i.e. an element is attached to the class from whom model the trajectory is most likely drawn). Figure 8.1 shows the log-likelihood of each trajectory versus both GMM/HMM hybrid models. On this plot, the clusters belonging to each controller are quite well separated. However, the maximal likelihood classification is not perfect: 5 trajectories of the first controller (2.5%) and 2 of the second (1.6%) are badly classified. The error rate of the classification is a measure of the quality of the modeling, but it does not give a good idea of the cluster separation. Thus, the inter-cluster distance defined in Section A.5.2 can be used to evaluate this separation. In this specific case, this distance is 5.9.

To study the influence of the number of states and the number of Gaussian distributions on the modeling quality, a parameter space was spanned from 1 to 15 states and from 1 to 6 Gaussian distributions per state, for a total of 90 experiments per run. To have a good statistical measure, 93 runs were made.

Figure 8.2 shows the inter-cluster distance separating the two controllers as a function of the number of states and Gaussian distributions per states. In this particular case, the distance between the two clusters in the log-likelihood space is not really influenced by the number of hidden states of the HMM or the number of Gaussian distributions per mixture.

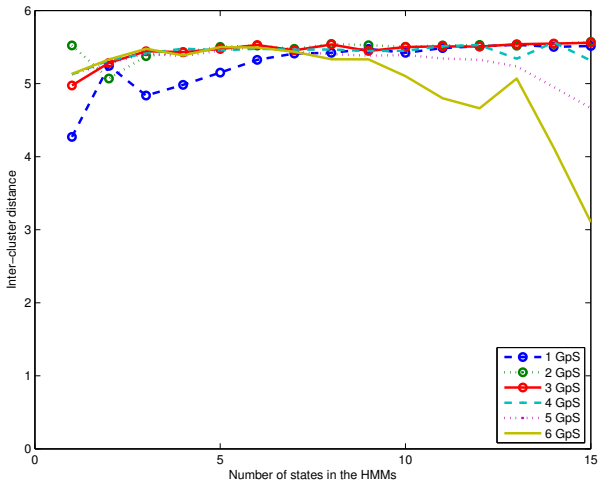
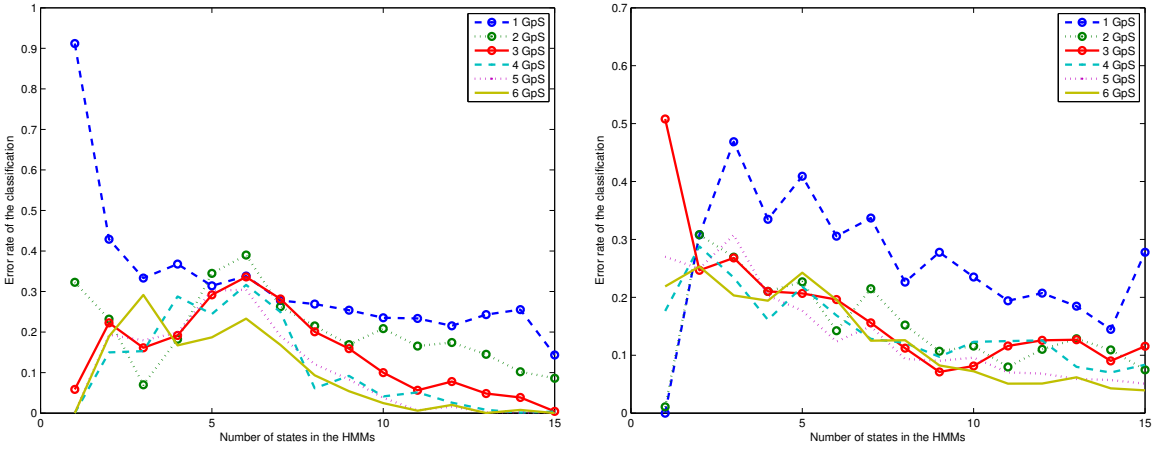
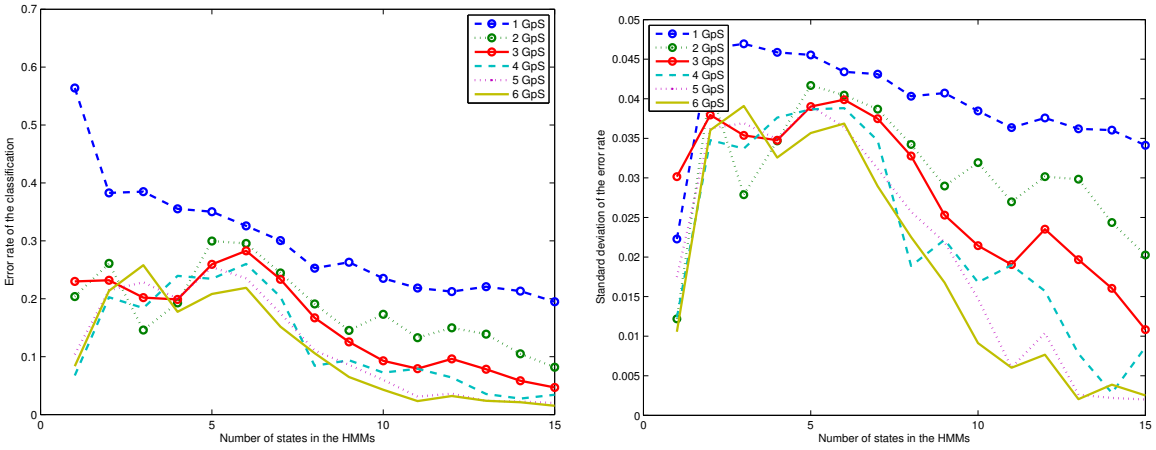


Figure 8.2: Average inter-cluster distance between the trajectory clusters of the first and second controllers, measured in the log-likelihood space. The acronym *GpS* stands for the number of Gaussian distributions per output state



(a) First controller (b) Second controller

Figure 8.3: Average error rate of the classification of the trajectories as a function of the number of hidden states per HMM and number of Gaussian distributions per mixture. The classification was performed using the maximum likelihood method. The error rate for the trajectories of the first and second controllers can be found on the left and on the right, respectively



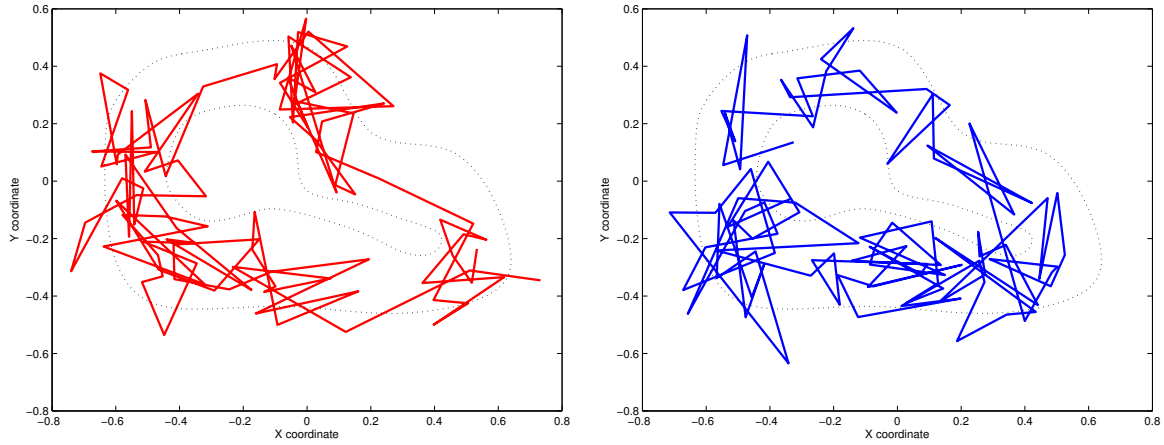
(a) Average Error Rate for both Controllers

(b) Average Std of Error Rate for both Controllers

Figure 8.4: On the left, average classification error rate using the two HMM/GMM models for all the trajectories of both controllers together. A clear influence of the number of states is observable. On the right, the average standard deviation of the classification error rate. The classification variance is also clearly decreasing with the number of states

The classification performance can be used to measure the modeling quality, instead of the inter-cluster distance measured in the likelihood space. Figure 8.3 shows the error rate of the classification using the maximum likelihood method. The average value was computed using the same 93 runs than for the inter-cluster distance. It can be observed that increasing the complexity of the model leads to a better classification. However, even with the most complex model (15 hidden states and 6 Gaussian distributions per mixture), a small error rate ($\sim$4%) remains when classifying the trajectories of the second controller.

Figure 8.4(a) is a summary of the two previous graphics, as it shows the average error rate for the trajectories of both controllers, instead of just one. There is a clear decrease of the error rate with the increase of the number of hidden states in the HMM model. Increasing the number of Gaussian distributions per states reduce also the rate of the classification errors. However, for a HMM/GMM model made of 1 state and 6 Gaussian distributions (that is equivalent to a pure GMM model), the classification error rate is not so bad with less than 10% of errors, but remains higher than that achievied with the most complex model. It is quite interesting to notice that a fairly simple model can achieve better results than a more complex HMM/GMM model (e.g. 6 hidden states and 6 Gaussian distributions). Excepting the pure GMM model (i.e. only one hidden state), the classification rate variance is also clearly decreasing when the number of hidden states is increasing. The near-optimal solution appears to be the most complex model (i.e. 15 hidden states and 6 Gaussian distributions).



(a) GMM/HMM model 1

(b) GMM/HMM model 2

Figure 8.5: One trajectory generated with both HMM/GMM models representing the first and second controllers respectively. Advance on the track is visible, but the generated trajectories are not similar at all to the original ones

8.1.1.1 Trajectory generation

Another way to assess the quality of the modeling of the trajectories with HMM/GMM is to use the models to generate trajectories. Figure 8.5 shows one trajectory generated with both models (10 states and 4 Gaussian distributions per state). These plots clearly show that the models do not extract all the subtle elements of the trajectories used for learning. The intrinsic nature of the model is not fitting to data it is representing. However, the advancement on the track is clearly visible.

8.1.1.2 Model parameters

Figure 8.6 shows the positions of the centers of the four Gaussian distributions for each of the 10 hidden states. In this case, each state represents a different circuit part and the succession of the states is clearly visible. As the hidden Markov Chain of the HMM is fully linked (i.e. the transition from one state to all the other states is always possible), there is no reason that the state numbers are ordered along the track.

To understand better how the modeling is done, a thorough analysis of the model parameters can be made. The HMM/GMM example presented at the beginning of the section will be used. It is made of 10 hidden states and 4 Gaussian distributions per state. It was evolved using the 201 trajectories generated with the first controller. At the learning start, all the parameters (Transition and emission probabilities, mean and variance values of the Gaussian distributions) were initialize randomly. The HMM model was full (i.e. from each state, there is a non-null probability to go to all the other states).

Figure 8.6 shows the positions of the 4 Gaussian distributions for each hidden state, at the end of

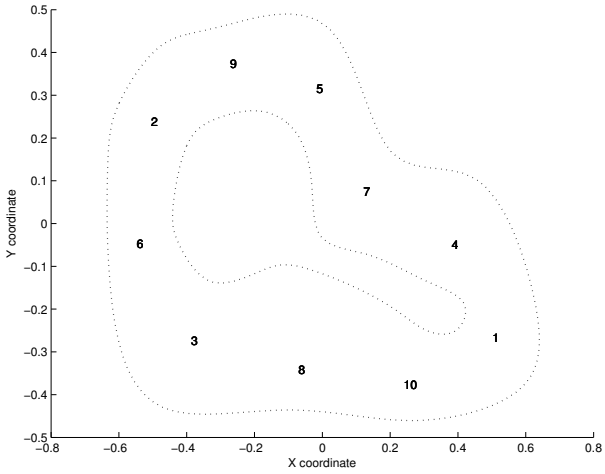


Figure 8.6: Position of the four Gaussian distributions for the 10 states of the HMM/GMM model of the first controller. As for each state, the four Gaussian distributions possess nearly the same center, the four plotted numbers are so close that they cannot be differentiate

Table 8.1: Matrix of the transition probability (in percents) of the HMM/GMM model of the first controller. The value at the position $[I_k, O_l]$ corresponds to the probability (in percents), given the current state k, that the next will be state l

	O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8	O_9	O_{10}
I_1	90	0	0	0	0	0	0	0	0	10
I_2	0	89	0	0	0	0	0	0	11	0
I_3	0	0	90	0	0	10	0	0	0	0
I_4	10	0	0	90	0	0	0	0	0	0
I_5	0	0	0	0	89	0	11	0	0	0
I_6	0	12	0	0	0	88	0	0	0	0
I_7	0	0	0	10	0	0	90	0	0	0
I_8	0	0	9	0	0	0	0	91	0	0
I_9	0	0	0	0	11	0	0	0	89	0
I_{10}	0	0	0	0	0	0	0	10	0	90

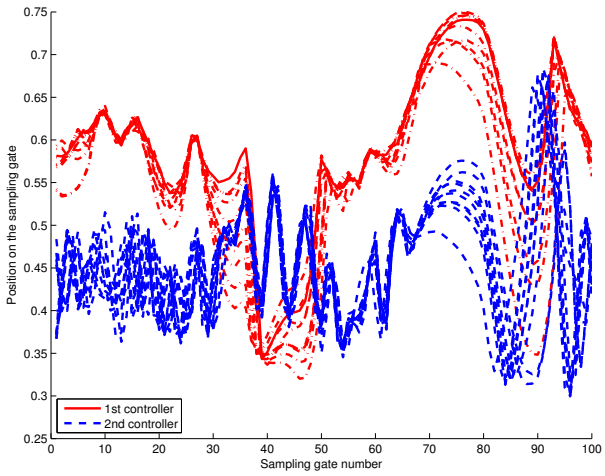


Figure 8.7: 10 example trajectories for both controllers displayed as a function of their positions on the sampling gates

the optimization process. For each state, the four distribution centers are so close, that the four values cannot be differentiated on this graph. The different states are also equally spread along the track of the circuit. For each state, the variance of the four Gaussian distributions are nearly the same. As a result, having four distributions per state does not seem to add something to the final model and the emission probabilities are uninteresting, as choosing a different Gaussian distribution will result in nearly the same distribution.

The last model parameters are the transition probabilities, displayed as a matrix in Table 8.1. This matrix is quite simple to apprehend: for each state, there is approximately a probability of 0.9 to stay in the same state and a probability of 0.1 to go the state that is located after it on the track. There is a circular order of the state: $[1 \rightarrow 10 \rightarrow 8 \rightarrow 3 \rightarrow 6 \rightarrow 2 \rightarrow 9 \rightarrow 5 \rightarrow 7 \rightarrow 4 \rightarrow 1 \rightarrow \ldots]$ and this order is clearly visible in Figure 8.6. The probability values can also be linked to the trajectory data. As each trajectory is reduced to 100 sampled points, there is approximately 10 points per state ($\frac{100}{10}$) and it explains the transition probabilities (0.9 and 0.1).

The resulting model is thus really simple and even though there are a lot of parameters, they could be reduced to a small number of values. It is however surprising that the classification error rate (Figure 8.3(a)) is clearly not the same for a model with one to six Gaussian distributions. Perhaps, having a larger number of Gaussian distributions improves the learning phase.

8.1.2 Model of the positions on the gates

As Figure 8.5 shows that the previous GMM/HMM models poorly represent the trajectories, other GMM/HMM models were built feeding them with the trajectory positions on the gate instead of the

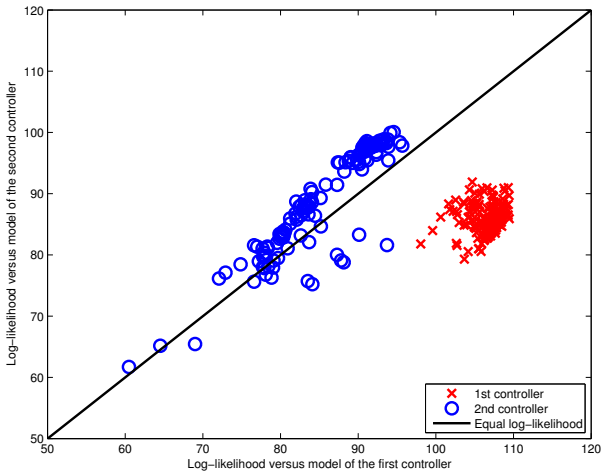


Figure 8.8: Log-likelihood of the trajectories generated with the first and second controllers versus the two GMM/HMM hybrid models created with the trajectories of only one controller

spatial coordinates of the sampled points. Figure 8.7 shows trajectory examples from both controllers displayed as a function of their positions on the sampling gates. The trajectories in this form correspond much more to the kind of process that the HMM/GMM is supposed to represent. For the trajectories of the first controller, a minimum of 3 states can be distinguished:

- A first state on the left where the values are pretty constant (Gates 0 to 35),
- A sudden change of state between gates 35 and 50 where the position on the gate are clearly lower,
- A final state (gates 50 to 100), where there is much more variance.

Further subdivision could be made, but for each of these states, the representation by a GMM is not so bad, even if non-random patterns are clearly visible.

As an example, two HMM/GMM models were learned using the previously described dataset. They were made of 5 hidden states and 3 Gaussian distributions per mixture and each model was evolved using the trajectories of a unique controller. Figure 8.8 shows the location of the trajectories in the space formed of the log-likelihood of the trajectories to the two models. The separation between the two clusters is clearly enhanced in comparison to the previous experiments using the spatial positions of the sampled points. However, the maximum likelihood classification provides even worse results, as there is a classification error rate of 14.5% for the trajectories of the second controller, even if those of the first controller are perfectly classified. This result can be related to the larger variance contained in the second controllers, as it was visible in Figure 7.7. Moreover, a line could be drawn between the two clusters and thus a linear classifier built on top of the HMM/GMM would result in a perfect classification. One

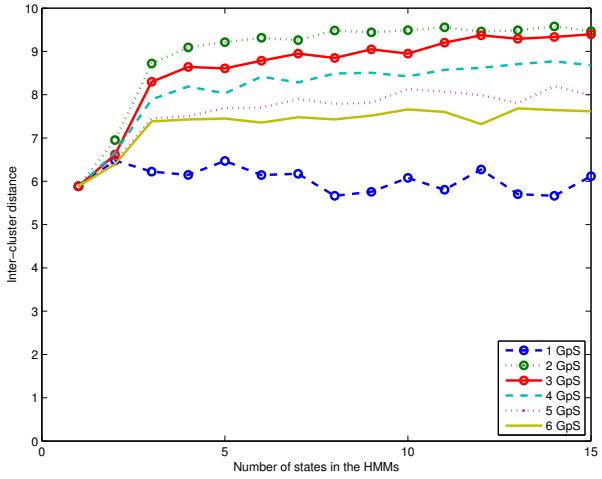


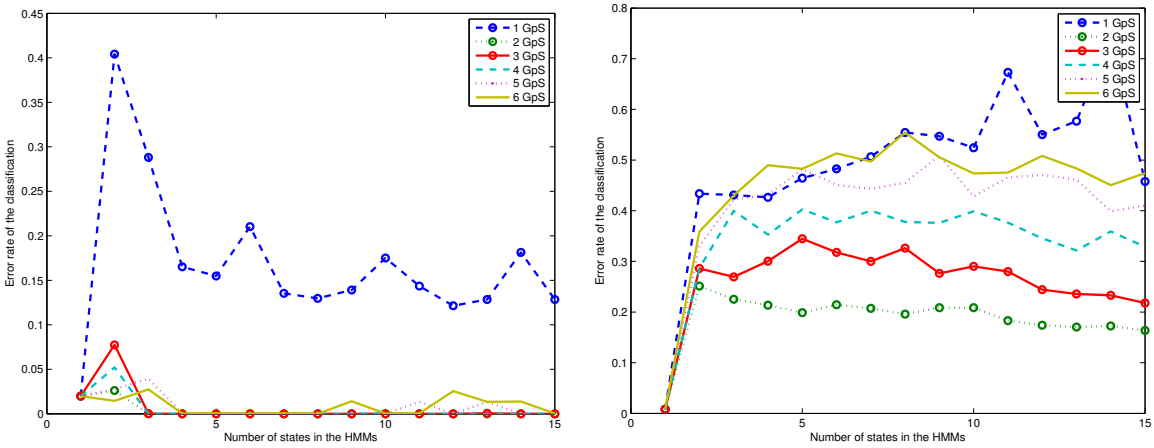
Figure 8.9: Average inter-cluster distance between the trajectory clusters of the first and second controllers, measured in the log-likelihood space

point is that the inter-cluster distance between the two clusters is 9.4, clearly bigger than in the previous example.

The influence of the number of hidden states and Gaussian distributions per mixture was then studied in a similar way as before. The research space spanned also from 1 to 15 hidden states and from 1 to 6 Gaussian distributions per mixture. 73 runs of the 90 experiments were done to acquire valid statistical data. Figure 8.9 shows the inter-cluster distance as a function of the number of hidden states and Gaussian distributions per mixture. Increasing the number of hidden states clearly increases the distance between the two clusters. A maximal value is obtained with about 5 hidden states. Moreover, using only two Gaussian distributions per mixture gives the best separation.

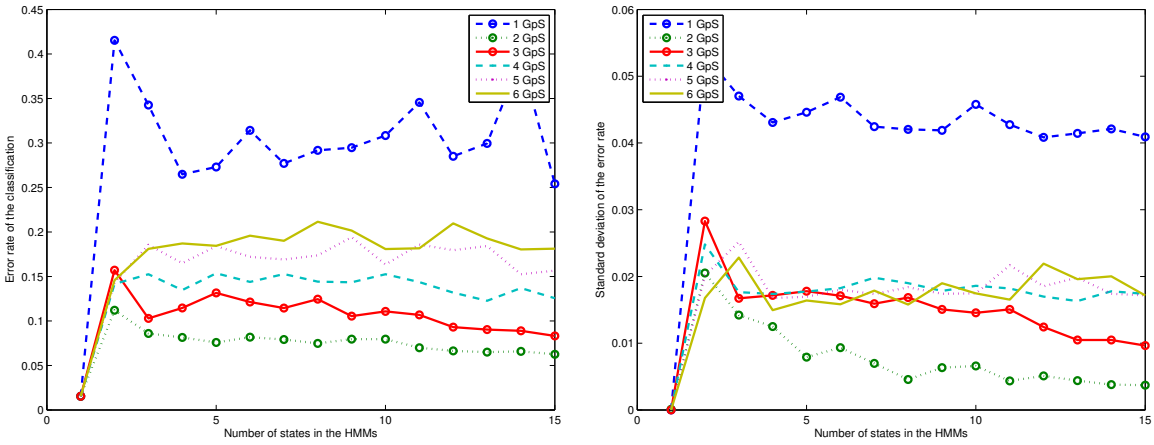
Figure 8.10 shows the classification error rates for the trajectories of each controller. With more than 2 Gaussian distributions per mixture and more than 4 hidden states, a nearly perfect classification is achieved with the trajectories of the first controller. Excepting the pure GMM model (i.e. 1 hidden state), the classification is quite bad for the trajectories of the second controller. Even for the best solution (i.e. 2 Gaussian distributions per mixture), an error rate of approximately 20% remains.

Figure 8.11(a) shows the average classification error rate for the trajectories of both controllers. Increasing the number of states above three does not influence the classification error rate. The second most simple model (i.e. only two Gaussian distributions per mixture) gives clearly the best results. Figure 8.11(b) shows the average standard deviation of the classification error. Above 3 hidden states, this variance is quite constant, excepting the models with 2 or 3 Gaussian distributions per mixture, whose variance is decreasing with the number of hidden states.



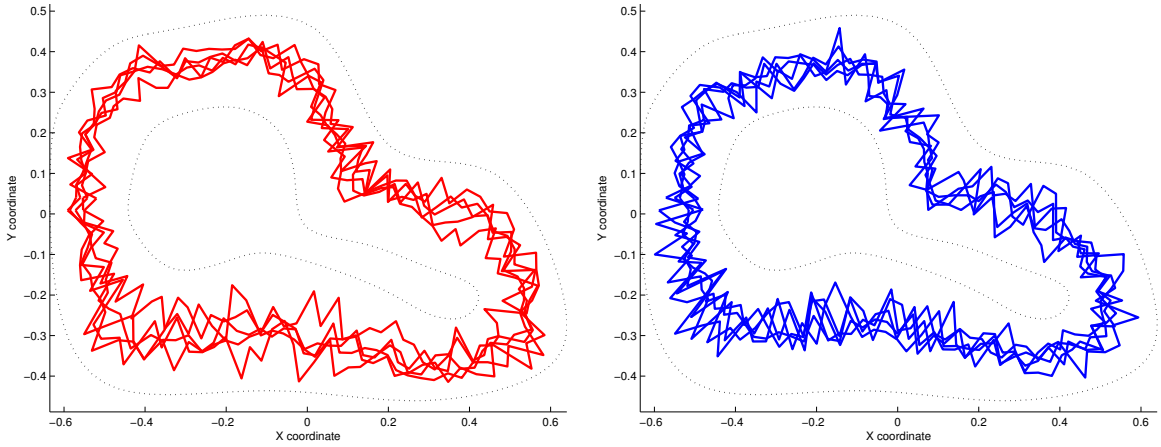
(a) First controller (b) Second controller

Figure 8.10: Average error rate of the classification of the trajectories as a function of the number of hidden states per HMM and of the number of Gaussian distributions per mixture. Classification was achieved with the maximum likelihood method. The error rate for the trajectories of the first and second controllers can be found on the left and on the right, respectively



(a) Average Error Rate for both Controllers (b) Average Std of Error Rate for both Controllers

Figure 8.11: On the left, average classification error rate using the two HMM/GMM models for all the trajectories of both controllers together. With more than 3 hidden states, the results are pretty constant. On the right, the average standard deviation of the classification error rate. The classification variance is also slightly decreasing with the number of states, especially with 2 Gaussian distributions per mixture.



(a) Trajectories of the first GMM/HMM model

(b) Trajectories of the second GMM/HMM model

Figure 8.12: One trajectory generated with both HMM/GMM models representing the first and second controllers respectively. The generated trajectories are much more similar to the original trajectories than in the previous experiment (Figure 8.5). However, the difference with the original trajectories is also clearly visible

Finally, it is important to note than a simple GMM with even a unique Gaussian distribution can achieve a fairly good classification with a really small variance. This classification is even better than all the more complex HMM/GMM hybrid models. However, leaving out the maximum likelihood classification method and building a linear classifier on top of the HMM/GMM model would give us better results, as shown previously. However, the modeling remains complex. As the PDM shows better classification results, it seems useless to use such a complicated method.

As shown before, the HMM/GMM hybrid models can be used to generate trajectories. The models previously presented, made of 5 hidden and 3 Gaussian distributions per mixture, were thus used to generate a trajectory. Figure 8.12 shows the two resulting trajectories. Even if they are staying on the track and that they are much closer to the original ones, a difference between the patterns of the original and generated trajectories is clearly visible.

8.1.2.1 Model parameters

The model parameters can be dissected. As previously, the parameters of the Gaussian distributions are nearly the same for each mixture. The values of the mean and of the standard deviation are shown in Table 8.2. S_1-S_2 and S_4-S_5 are really similar, and thus only three different Gaussian distributions can be observed. It could even be further reduced to 2 distributions (S_3 and $S_{(1,2,4,5)}$). Table 8.3 shows the matrix of the transition probabilities between the hidden states of the HMM. Two blocks are clearly

Table 8.2: Parameters of the Gaussian distributions (mean and standard deviation) for each of the 5 hidden states ($S_1 \dots S_5$)

	S_1	S_2	S_3	S_4	S_5
Gaussiam mean	0.4592	0.4594	0.6059	0.4543	0.4542
Gaussian STD	0.0167	0.0167	0.0159	0.0166	0.0167

Table 8.3: Matrix of the transition probabilities (in percent) of the HMM/GMM model of the first controller. The value at the position $[I_k, O_l]$ corresponds to the probability, given the current state k, that the next will be state l

	O_1	O_2	O_3	O_4	O_5
I_1	37	23	3	34	4
I_2	24	47	4	19	6
I_3	7	3	86	2	2
I_4	35	16	0	27	22
I_5	13	8	2	37	39

visible: the third state and the other states. There is a high probability to stay inside these blocks and a small probability to leave them. As in the previous experiments, the model seems far too complex for what it really contains and it could be reduced to a model made of 2 hidden states and 1 Gaussian distribution per state.

8.1.3 Discussion

Even though the classification done by HMM/GMM models is not perfect, they keep their advantage. The HMM/GMM model can handle all kinds of trajectories. They can compare trajectories of different lengths and shapes, and there are not a lot of parameters to choose (number of hidden states and number of Gaussian distributions per state). As a result, this kind of model is quite efficient to compare sets of trajectories that are really different as Porikli demonstrated in [47]. However, when comparing sets of trajectories slightly different, the PDM analysis presented previously seems much more efficient. Moreover, HMM/GMM models need much more computational resources. For a model made of 15 hidden states and 6 Gaussian distributions per state built on the spatial position of the trajectories, the learning process takes 3 to 15 minutes on a Intel Dual Core 2 processor running at 2.4 GHz, when the PDM analysis took only 14 milliseconds on the same computer. Finally, the HMM/GMM model is really complex and a number of customization techniques can be adopted in order to better match the model to the trajectory data sets, as it has been done previously in the literature [47, 46, 35, 99].

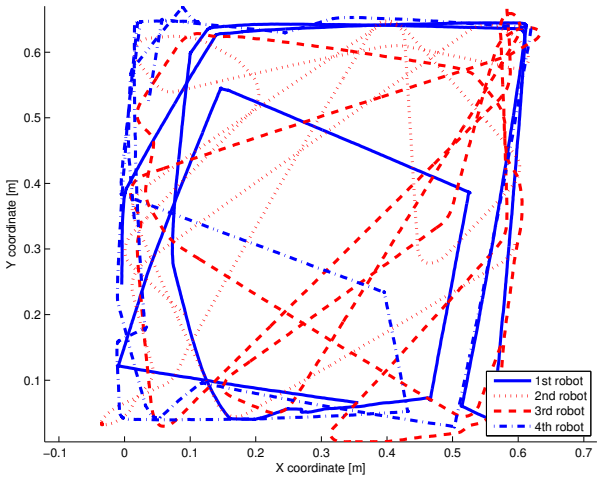


Figure 8.13: Four trajectories without common frame. Each trajectory is made of 2000 points. The trajectories of the first and fourth robots possess sharp angles and correspond to a rule-based controller. The two other trajectories (second and third robots) were generated with a Braitenberg controller and display more smooth turns.

8.2 Correlated random walk

This section presents the Correlated Random Walk model (CRW) as an alternative to the PDM for the analysis of trajectories without common frame. The inefficiency of the PDM will first be proved with a simple case study. Then the CRW will be thoroughly studied and its advantages and drawbacks will be pointed out. Finally, modification of the CRW will be proposed to improve its performance.

8.2.1 Comparison of controllers

A simple case study was built to generate trajectories without common frame. It was made of a square arena of about 0.7x0.7 meters and into it, four robots were avoiding themselves and the walls. The trajectories were extracted with an overhead camera, using the open source software SwisTrack [95, 96]. The robot positions were acquired at 15 Hz. The average calibration error was around 2.5 mm. If an error of 1 pixel is considered for the robot tracking in the image, an average error of 1.5 mm must also be added. The overall positioning error could thus be evaluated to 4 mm. The four robots were e-puck differential-drive robots [94] and they were driven by two different controllers: a Braitenberg controller and a rule-based controller. Both controllers were used for previous experiments and were presented more thoroughly in Section 7.1.1.1. Two robots were embedded with each controller. The resulting trajectories are shown in Figure 8.13. The differences between the two behaviors is visible in the curves.

To show how the PDM does not suit to this kind of trajectories, a simple analysis was performed. The

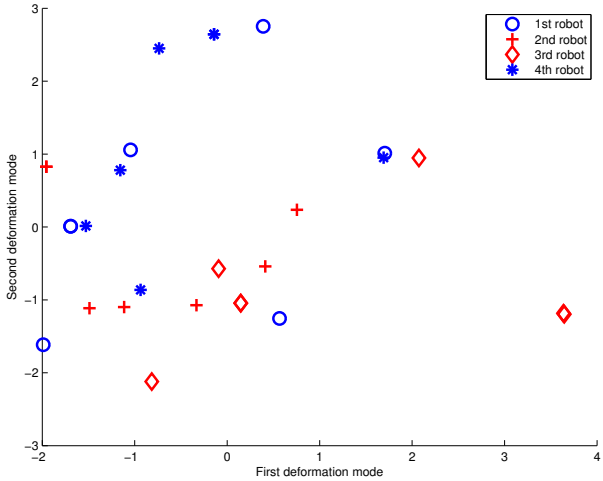


Figure 8.14: PDM analysis of six parts of 1000 points for each trajectory shown in Figure 8.13. There is no clustering of the different robots or behaviors

trajectories were first cut in parts of 1000 points and then resampled with only 100 points, keeping only one point for ten successive points, emulating an acquisition frequency of 1.5 Hz. Six parts were kept for each trajectory and a PDM was built with their spatial coordinates. Figure 8.14 shows the location of the trajectory parts in the PDM space. The behaviors are clearly not clustered in this space and no pattern is visible. Another indicator is the cumulative energy of the PDM modes. The first modes contain only 30% of the total energy and seven modes are needed to obtain more than 90%. The fact that the PDM method is not working is not surprising as the trajectory parts possess no spatial pattern to compare. The starting and ending points are not the same, and the spatial coverage of the robot motion is also completely different.

As the trajectories cannot be compared as a whole, their difference can be measured between the distributions of small patterns: the Correlated Random Walk model presented in Section 5.3. The first method to acquire the distributions of step lengths and angles was used, it takes steps of same duration. It stores the segment lengths between two successively acquired points and the angles between this segments. Only 20% of the points were taken, emulating an acquisition rate of 3 Hz. To reduce the tracking noise influence, each step whose dimension was smaller than 2.5 mm was considered as a step of null length and the corresponding angle was also put to zero. The resulting histograms are shown in Figure 8.15. The histograms are quite similar: most of the steps are located in two specific regions. The first region corresponds to the moments when the robot is turning on place or is stopped (step length close to zero and step angle also close to zero). The second region corresponds to the straight movement of the robot at full speed (step length close to 15 mm and step angle close to zero). The dispersion of the steps at this location is mainly caused by tracking errors. The difference between the two controllers

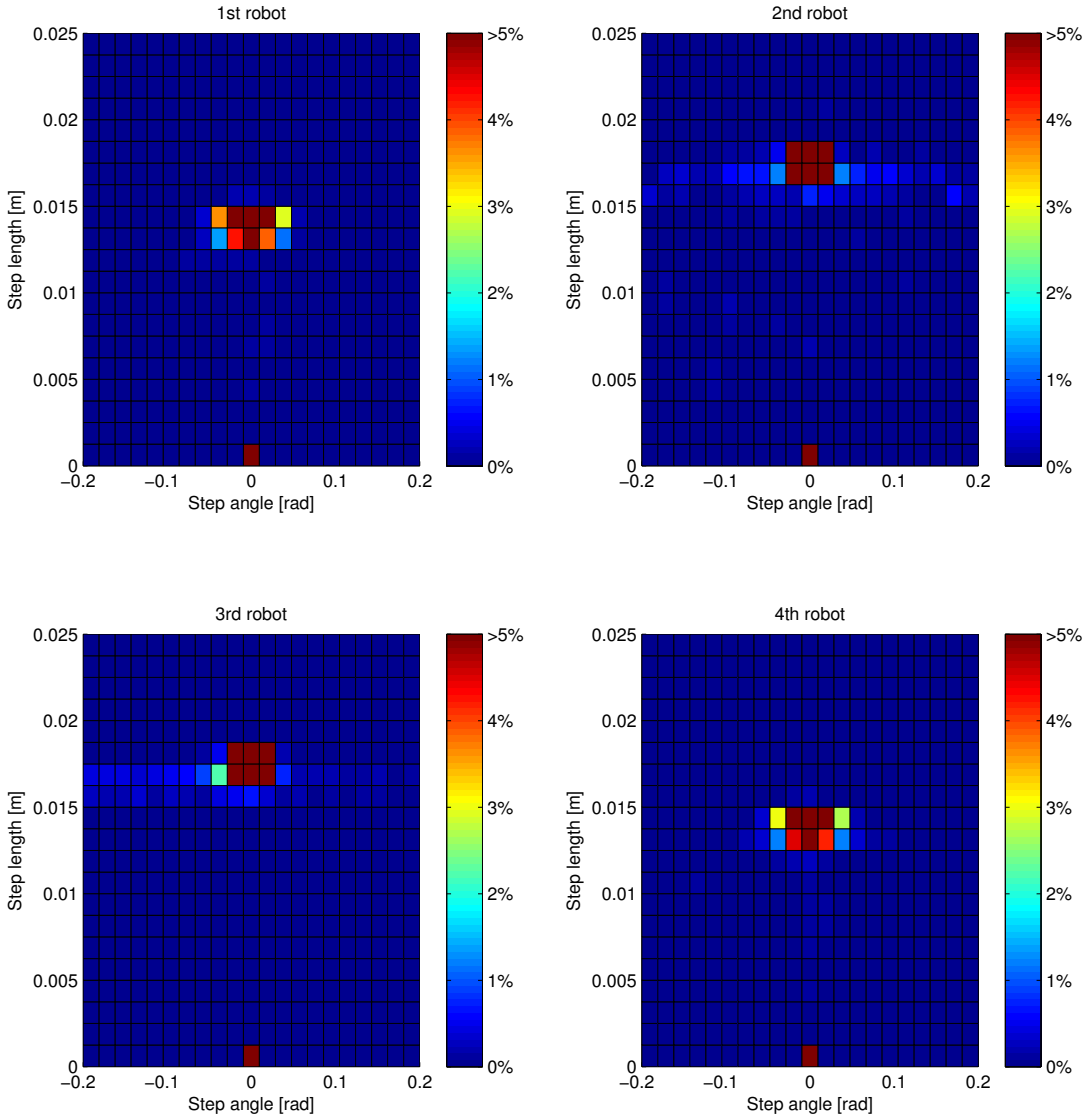


Figure 8.15: Histograms of the CRW distributions of Figure 8.13 trajectories. Even if the histograms are quite similar, the difference between the two behaviors remains visible

Table 8.4: Comparison of the CRW distributions of Figure 8.13 trajectories. The values measured correspond to the result of the F-F test and the difference of CDF in the histograms of Figure 8.15

	F-F test				F-F CDF diff. (%)				Hist. CDF diff. (%)			
	1^{st}	2^{nd}	3^{rd}	4th	1^{st}	2^{nd}	3^{rd}	4th	1^{st}	2^{nd}	3^{rd}	4th
1^{st} Rob.	0	30.9	58.6	5.2	0	89.7	90.0	9.5	0	90.5	91.2	8.7
2^{nd} Rob.	30.9	0	3.4	28.6	89.7	0	9.9	87.8	90.5	0	7.3	90.8
3^{rd} Rob.	58.6	3.4	0	48.5	90.0	9.9	0	89.1	91.2	7.3	0	91.5
4th Rob.	5.2	28.6	48.5	0	9.5	87.8	89.1	0	8.7	90.8	91.5	0

is however visible. The rule-based controller is a bit slower (shorter step lengths) and the Braitenberg controller is turning more often close to its full speed (there are more non empty bins for step angles different from zero and step lengths around 15 mm).

The difference between the distributions can be computed using the Fasano-Franceschini extension of the K-S test (F-F test, Section A.3.3). The results are shown in Table 8.4. The Z_n values of the F-F test must be compared to the critical value of the test for the number of elements and the correlation. This value is around 1.7 for all the comparisons. Thus, the test concludes that all the distributions are different with a confidence interval of more than 95%. However, the difference is sufficiently important to create two groups of robots (1-4 and 2-3), corresponding to the two controllers.

The maximal CDF difference measured with the F-F algorithm can be compared with the maximal difference of CDF within the histograms. This approximation is quite good. For large datasets, using CDF difference of the histograms can be useful, as computing a histogram has a complexity of n when the F-F test is in n^2. As most of the time, the number of histogram bins is much smaller than the number of elements, computing the CDF difference in the histogram takes a time negligible compared to the histogram computation.

The influence of the acquisition period on the controller separation was studied. A small period (0.06 s, 15 Hz) will emphasize the tracking noise, as the robot is not moving a lot between two acquired points. A this rate, the maximal length of a robot step is around 3 mm, a value pretty close to the tracking error. For a period of 10 s, the maximal length is around 450 mm: more than half of the arena side. In this case, all the information contained in the curves is lost and thus the difference between the two controllers disappears. To evaluate this influence, the CRW distributions of the second, third and fourth robots were compared with the CRW distributions of the first robot for different acquisition periods. The comparison was performed using the F-F test. For a good clustering, the value resulting of the test, Z_n, for the fourth robot must be as small as possible (similar to the CRW distributions of the first robot). On the contrary, the Z_n values for the second and third robot must be as big as possible. Figure 8.16 displays the ratios of these Z_n values. A big ratio corresponds to a good separation. In this case, the optimal acquisition period

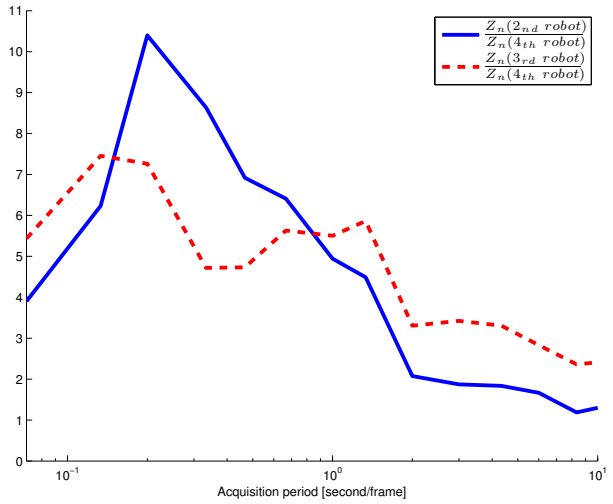


Figure 8.16: Influence of the acquisition period on the difference between the two controllers using the F-F test. For each acquisition rate, the CRW distributions of the second, third, and fourth robots are compared with the CRW distribution of the first robot. Thus, the ratio, Z_n value of the second or third distributions divided by Z_n value of the fourth robot, must be as big as possible to have the best separation

is around 0.2 s (5 Hz). This plot shows that the acquisition period has a major impact on the histogram differences: it must be sufficiently large to low-pass filter acquisition noise but as the same time not too large to filter out turning information.

The second method proposed in Section 5.3 was also used to extract CRW distribution. It approximates the trajectory with a succession of steps of variable length and duration, keeping the error between the trajectory and its approximation smaller than a defined value. Figure 8.17 shows the resulting histograms. The difference between the histograms corresponding to the two behaviors is noticeable. The rule-based controller produces more long steps (around 0.4 m), when the Braitenberg controller produces more shorter steps, corresponding to its smooth curves. For both controllers, the maximal step length (around 0.6 m) is clearly related to the arena size.

The F-F test can also be used to measure the difference between the CRW distributions. Table 8.5 shows the resulting Z_n values. As the CRW distributions have less elements (longer steps), the critical values of the F-F test for a 95% confidence interval are also a bit smaller: around 1.6. For each comparison, the test concludes that the distributions are different. However, it is still possible to group the CRW distributions corresponding to the two controllers.

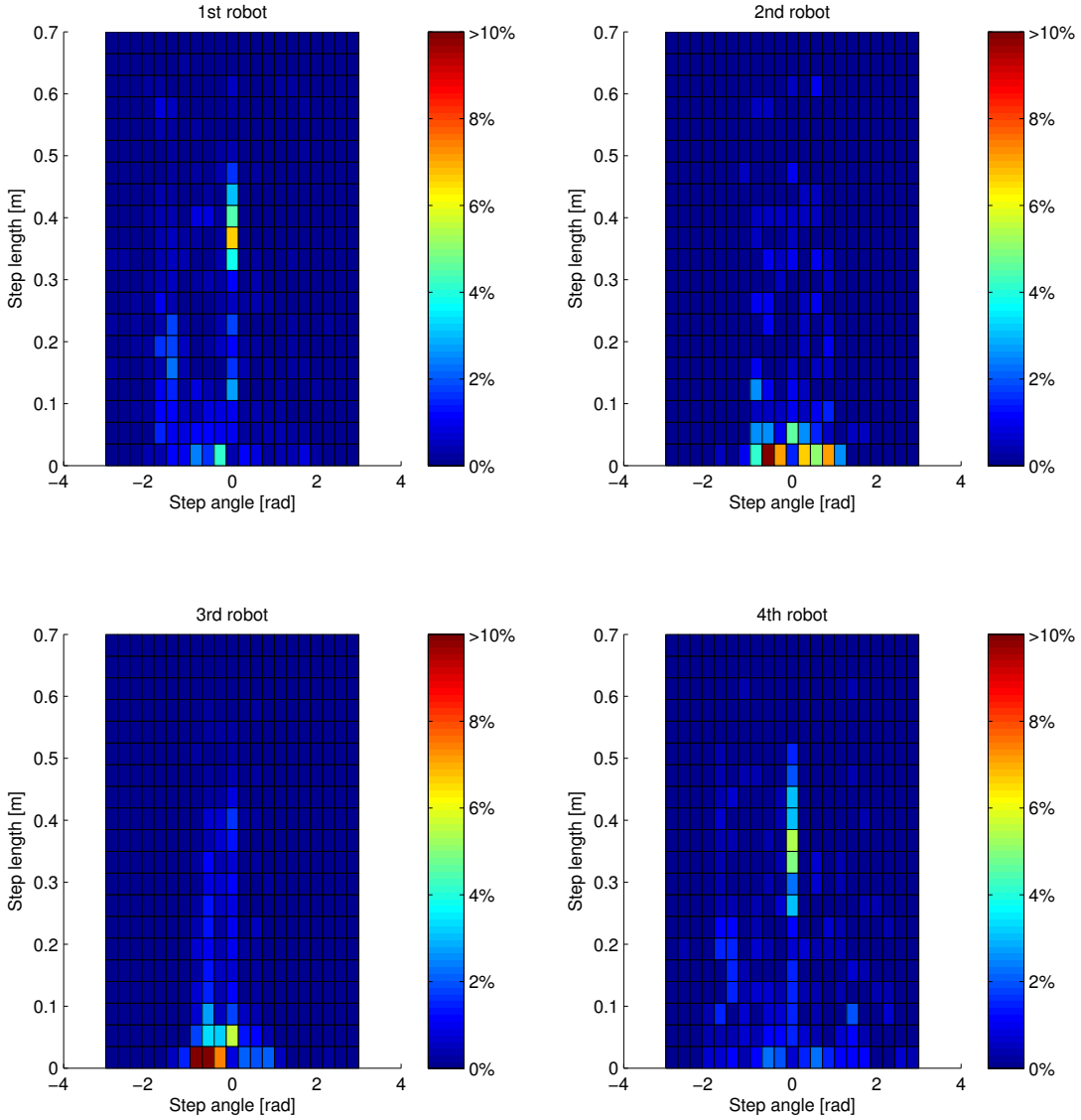


Figure 8.17: Histograms of the CRW distributions of Figure 8.13 trajectories. The histogram differences are less visible in this plot

Table 8.5: F-F test applied to the CRW distributions using the second extraction method. The displayed values corresponds to the Z_n values

	1^{st} Robot	2^{nd} Robot	3^{rd} Robot	4th Robot
1^{st} Robot	0	5.6	8.0	3.2
2^{nd} Robot	5.6	0	3.7	4.3
3^{rd} Robot	8.0	3.7	0	5.4
4th Robot	3.2	4.3	5.4	0

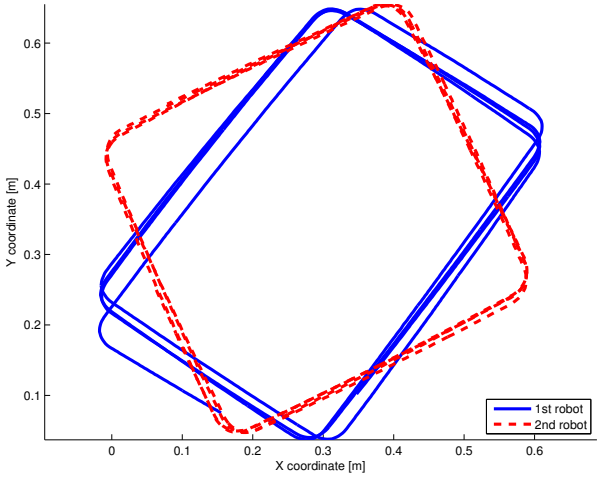


Figure 8.18: Trajectories of a robot driven by the Braitenberg controller when only two robots are at the same time in the arena. Each trajectory corresponds to a different experiment. In both cases, the movement becomes quite stable, as the robot is driving in squares

8.2.2 Dependence to experimental conditions

As previously demonstrated, the CRW method is quite efficient to analyze trajectories without common frame. However, it has limitations, as it will be shown here. The goal of this subsection is to show that there is no direct link between the controller implementation and the CRW distributions. As the movement behavior will change as a function of the experimental conditions, the resulting CRW distributions will also variate.

8.2.2.1 Direction of rotation

Two experiments were performed in the same arena as previously described. Only two robots were used for these two experiments and they were endowed only with the Braitenberg controller. In this particular

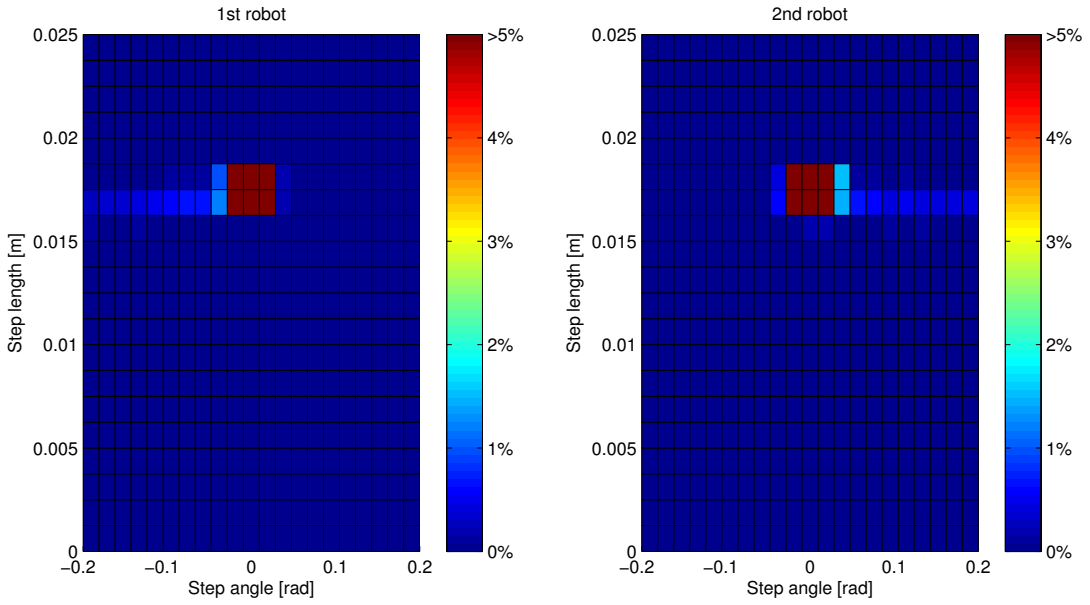


Figure 8.19: Histograms of the CRW distribution of Figure 8.18 trajectories, using the first method of distribution extraction. From these histograms, it can be observed that the first robot is mainly turning on the right, when the second is mainly turning on the left

case, when the system is stabilized, the two robots produce quite constant trajectories: they are driving in squares. The direction of rotation depends however on the initialization. Figure 8.18 shows two trajectories, one for each experiment. It is not visible in this plot, but for the first experiment, the robot was turning clockwise and for the second, it was turning counterclockwise.

The extraction of the trajectories was performed as before, with an acquisition rate of 3 Hz. The first method was used to extract the CRW distributions. The distribution histograms are shown in Figure 8.19. It is visible that the robots do not rotate in the same direction, the first robot is mainly turning on the right (more negative angles), when the second is mainly turning on the left. Thus, even if the robots and the controllers are exactly the same, the resulting behaviors are different and the CRW distributions are also different. There is no direct link between the CRW distribution and the controller.

The two CRW distributions remain however mainly similar, as the forward speed is the same and the main part of the two behaviors corresponds to driving straightforwardly. The F-F measure of the difference is $Z_n = 22.1$ for a maximal CDF difference of 30.1%.

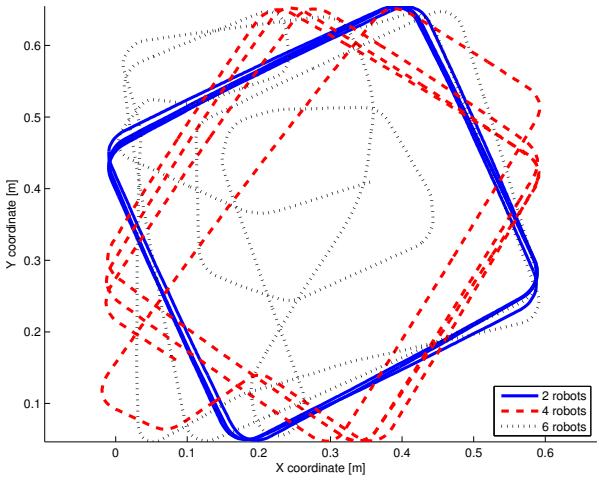


Figure 8.20: Trajectories generated with two, four or six robots in the arena. Increasing the number of robots increases the number of collisions and thus, influences on the trajectories

Table 8.6: Measure of the difference between the CRW distributions shown in Figure 8.21, using the F-F test. The table below shows the Z_n values resulting of the test and the maximal difference of CDF

	Z_n values			CDF diff. (%)		
	2 r.	4 r.	6 r.	2 r.	4 r.	6 r.
2 robots	0	12.4	11.4	0	27.7	25.5
4 robots	12.4	0	6.1	27.7	0	13.5
6 robots	11.4	6.1	0	25.5	13.5	0

8.2.2.2 Number of robots in the arena

To show another example, three experiments were performed in the same conditions as before, with the exception that the number of robots in the arena varied between two, four, and six. One trajectory for each experiment is shown in Figure 8.20. The trajectories are clearly influenced by the number of robots. More robots induce more collisions and, thus, shorter straight segments are resulting.

The CRW distributions were extracted using the first method. The resulting histograms are shown in Figure 8.21. The increase of collisions is clearly visible in this plot, as the proportion of non-zero angles is increasing with the number of robots in the arena. Table 8.6 shows the measure of the distribution differences using the F-F test. In these cases, the critical values for a confidence interval of 95% are around 1.7.

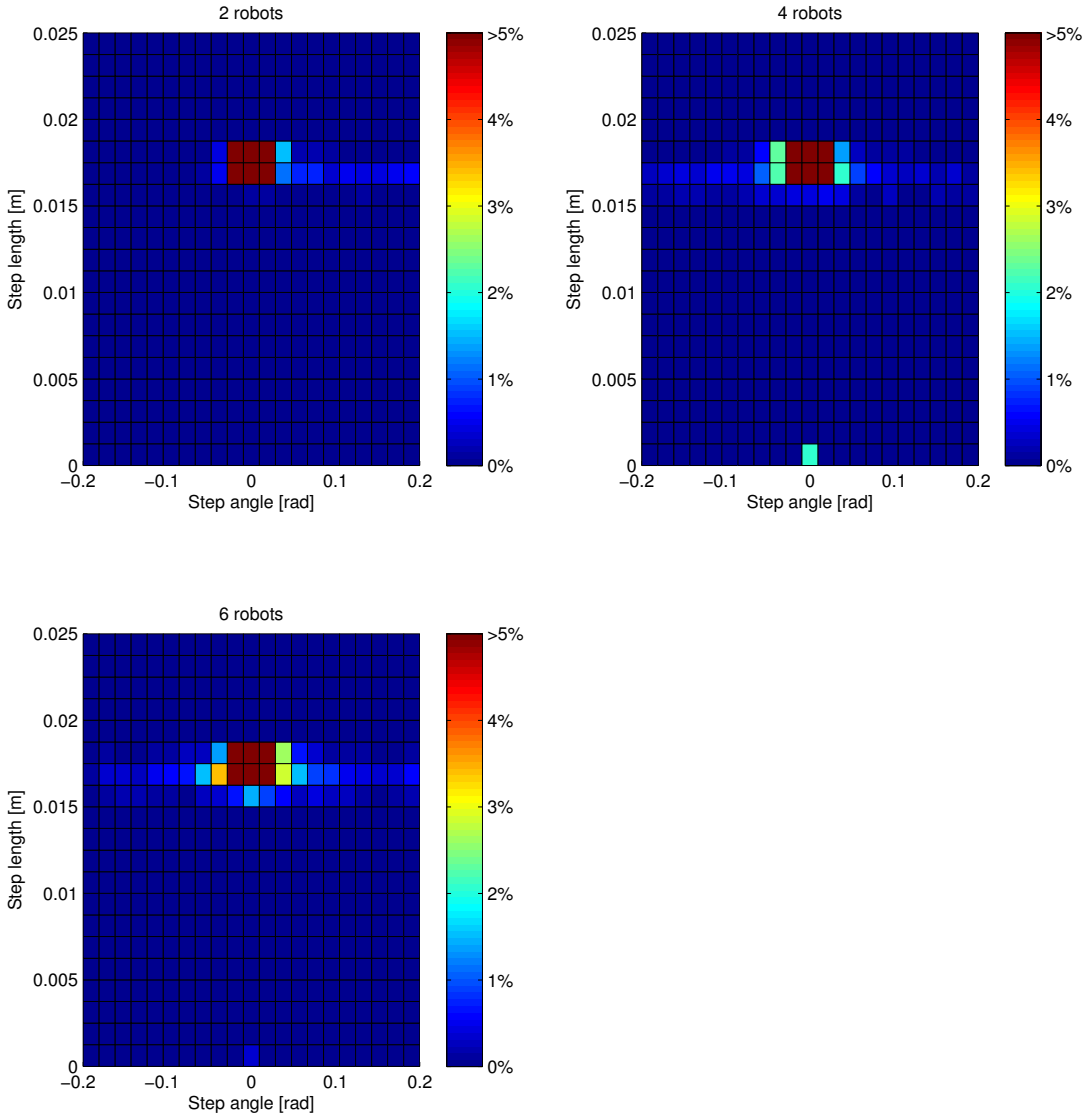


Figure 8.21: Histograms of the CRW distributions extracted from the trajectories of Figure 8.20. Increasing the number of robots clearly influences the CRW distributions, as the proportion of non-zero angles is increased

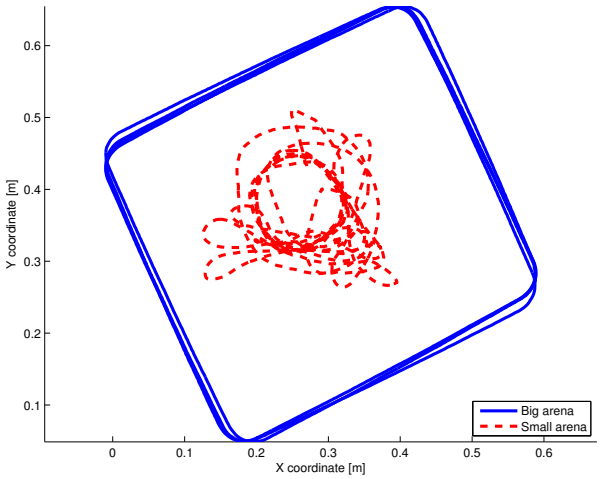


Figure 8.22: Two trajectories generated in two arenas of different size. The trajectory generated in the smaller arena possess obviously shorter straight segments

8.2.2.3 Arena size

The last parameter of influence studied was the size of the arena. A smaller arena was built with a side length of around 0.4 meter. In both experiments, two robots interact in the same arena. Both were driven by the Braitenberg controller. Figure 8.22 shows one trajectory for each experiment, acquired at 3 Hz. The trajectories are obviously influenced by the arena size, even if the robot hardware and software is kept identical. In the smallest arena, the two robots do not achieve a stable state, as they are continuously colliding. No repeatable pattern is thus produced.

The first extraction method was used to get the CRW distributions, and the resulting histograms are shown in Figure 8.23. The two behaviors are clearly different: reducing the arena size clearly increases the proportion of non-zero angles. Using the F-F test to measure the histogram difference, $Z_n = 21.4$ and the maximal CDF difference is 68.6%.

The second extraction method was also used to acquire the CRW distributions. Figure 8.24 shows the resulting histograms. The arena size has a clear influence on theses histograms: reducing the size clearly reduces the length of the steps. The comparison made with the F-F test gives a value of $Z_n = 4.6$ for a maximal CDF difference of 33.9%.

8.2.3 HMM/CRW hybrid model

In Section 5.3, three trajectories made from the same CRW distribution were introduced and shown in Figure 5.24. They can also be found on the top left plot of Figure 8.29. The histogram of their CRW distributions is shown in Figure 8.25. As it was already introduced in Section 5.3, the main problem of

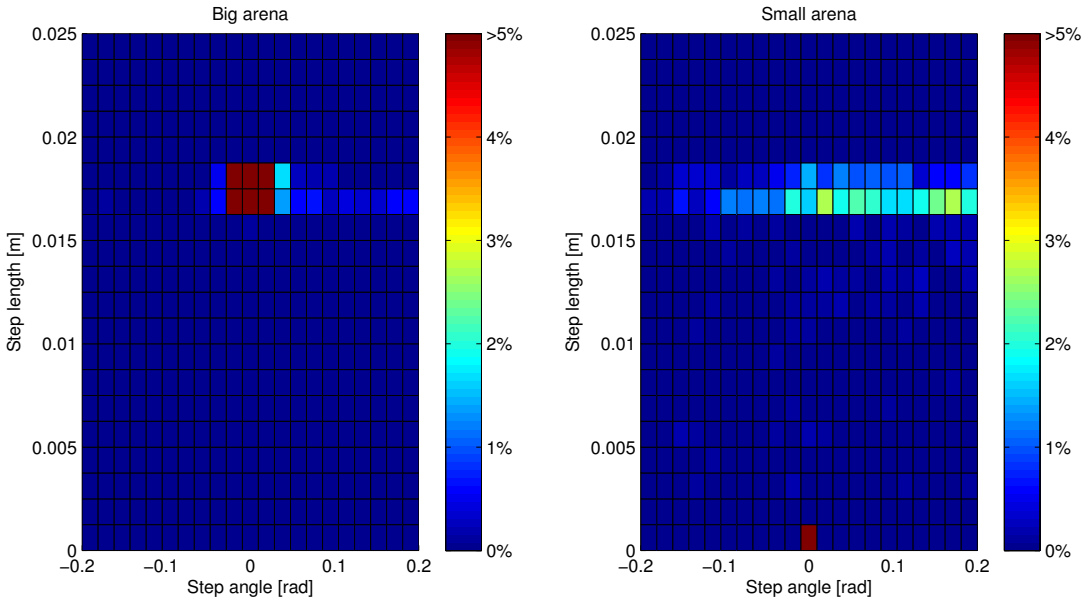


Figure 8.23: Histograms of the CRW distributions extracted from Figure 8.22 trajectories, using the first method. Reducing the arena size clearly increases the proportion of non-zero angles

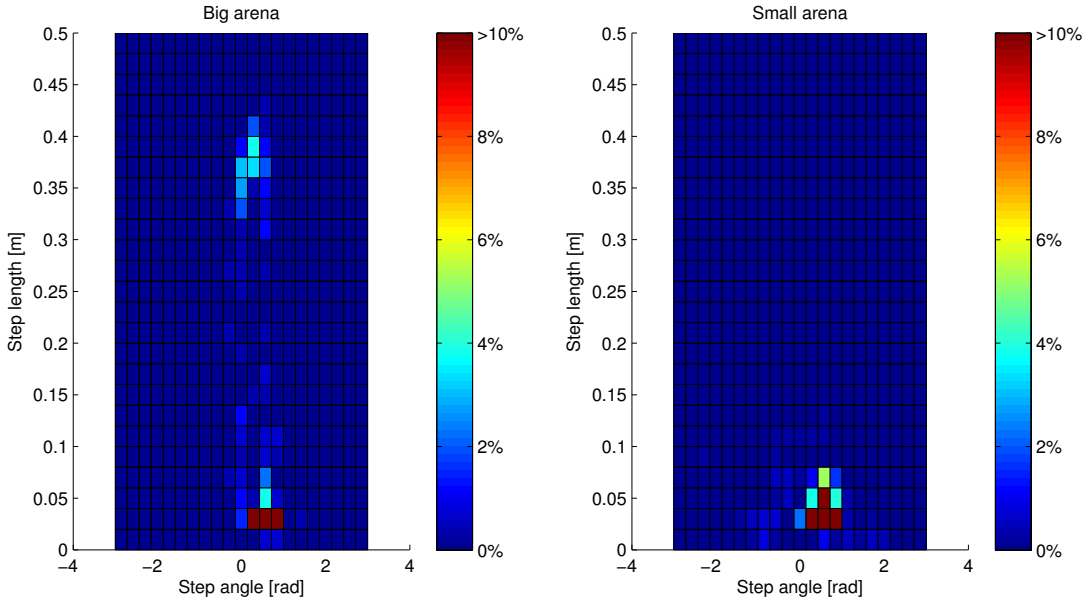


Figure 8.24: Histograms of the CRW distributions extracted from Figure 8.22 trajectories, using the second method. Reducing the arena size reduces also the step lengths

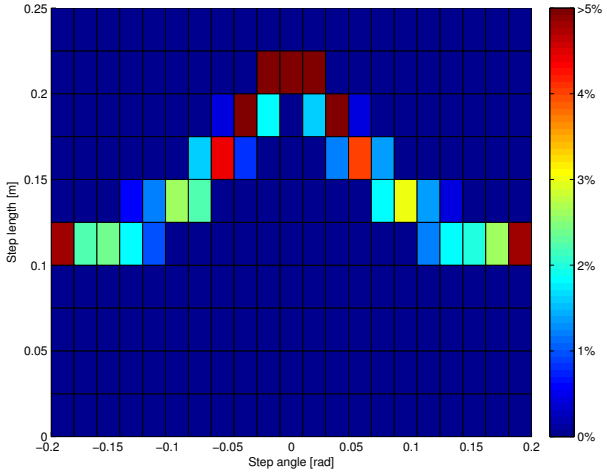


Figure 8.25: CRW distribution histogram of the three trajectories of Figure 5.24. They can also be found on the top left plot of Figure 8.29. As the histogram is exactly the same for the three trajectories, it is only displayed once

Table 8.7: Log-likelihood of each of the three trajectories versus the hybrid HMM/CRW models trained with each trajectory. The biggest log-likelihood (less negative) stay on the diagonal, as each trajectory is more similar to its own model

	1^{st} model	2^{nd} model	3^{rd} model
1^{st} trajectory	-1159	-2112	-3174
2^{nd} trajectory	$-\infty$	-1472	-17567
3^{rd} trajectory	-2851	-2062	-1290

the CRW model is the loss of the temporal link between the steps.

To provide a solution to this problem, a HMM can be built on top of the CRW, to model the transitions between the steps. The modeling is quite simple. Each bin of the histogram is given a state number. In the particular case, the histogram has $21 \cdot 10 = 210$ bins and thus 210 output states. Each trajectory can thus be transformed as a sequence of states and a HMM can be trained on this sequence. One HMM made of four hidden state was evolved for each of the three trajectories. The likelihood of each trajectory versus the three HMMs trained is shown in Table 8.7. Contrarily to the comparison of the CRW distributions, where no difference exists between the three trajectories, this hybrid model gives clear differences of likelihood. For classification purpose, this kind of model can thus be used to take into account the temporal link between the steps.

HMM/CRW models can also be used to generate trajectories. From the HMM, a sequence of states

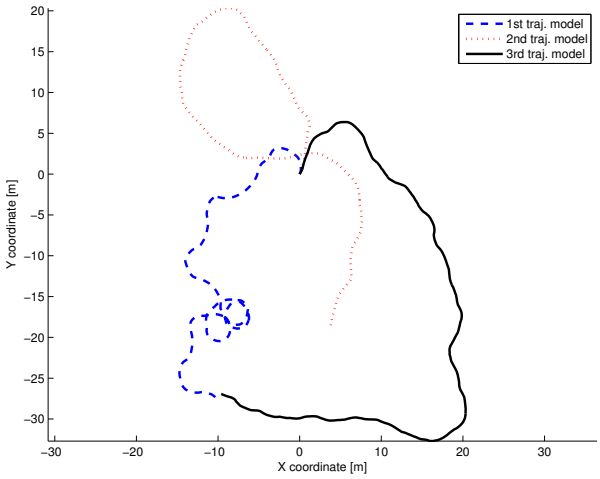


Figure 8.26: One trajectory generated with each of the three hybrid HMM/CRW models trained previously. The generated trajectories are clearly different from the original ones

can be created and then transformed into a list of spatial coordinates, using the edges of the histogram bins. The position inside the bin is chosen randomly using a uniform distribution. A trajectory has been generated for each of the three hybrid models trained. They are displayed in Figure 8.26 and are clearly different from the original trajectories. As it was already stated in the first section of this chapter, the HMM is efficient to differentiate or classify trajectories. However, it does not encompass all the trajectory aspects, and thus, it is not really helpful to generate trajectories similar to those used for the training.

8.2.4 CRW with history

Another solution taking the temporal link between the trajectory steps into account, is to create multidimensional distributions made of sequences of steps instead of just one step. Suppose that the length of the i^{th} step is l_i and the respective angle is a_i. If n steps of history are stored, the first sequence will be $[s_1 \dots s_n, a_1 \dots a_n]$ and the i^{th} sequence will be $[s_i \dots s_{i+n-1}, a_i \dots a_{i+n-1}]$. Thus, the distribution of sequences has $2n$ dimensions instead of two. The problem becomes quite complicated and completely impossible to visualize and for our experiments, PCA did not help us to reduce the dimensionality of the problem. For the previous experiments using the PDM, each trajectory corresponded to a vector and the standard vector of the different behaviors were spatially different. However, in this case, we are comparing distributions of sequences (vectors) and the difference does not lie in the vectors composing these distributions, but in the spatial repartition of these vectors. Thus, measuring the CDF difference between the different sequence distributions is the adequate tool.

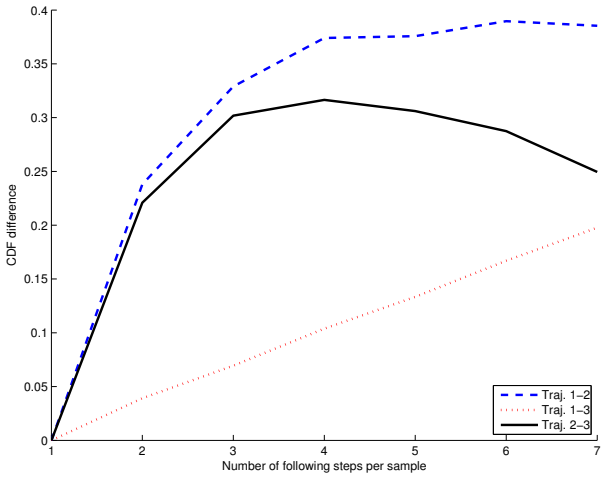


Figure 8.27: Maximal difference of CDF between the CRW distributions of the three trajectories presented previously, as a function of the number of steps of history used

The algorithm proposed by Fasano and Francescini (F-F test) can easily be extended to more than three dimensions. We implemented this extension to an unlimited number of dimensions to evaluate the maximal difference of CDF between the different trajectories. Even though, we were not able to compute the critical values of the test, the CDF difference is sufficient to measure the difference between two trajectories or the classify trajectories. Fasano and Franceschini algorithm is less exact than Peacock version, but with more than two dimensions, it keeps a complexity of n^2 when Peacock version has a complexity of $n^{(nb.\ dimensions)}$ which is impossible to handle.

Figure 8.27 shows this maximal difference of CDF for the three trajectories presented previously, as a function of the number of steps of history (length of the stored sequence). A history of one step corresponds to the original CRW model and as it was stated previously, it is not sufficient to measure difference between the three trajectories. However, with two and more steps of history, the difference between the trajectories is directly visible. Theoretically, increasing the history will only increases the difference of CDF, but Fasano-Franceschini algorithm is an approximation and is responsible of the decrease of the maximal difference of CDF when comparing the second and the third trajectories.

History can also be taken into account to generate trajectories more similar to the original ones. The computation is quite similar to the original CRW model. However, histograms with more than two dimensions are quite difficult to handle: the number of bins in increasing as a power of the number of dimensions. Thus, even with a small numbers of histogram bins like the 210 bins used for the previous CRW distribution histogram (Figure 8.27) and an history of five steps, the resulting $210^5 \approx 4 \cdot 10^{11}$ bins are difficult to handle. For most of the experiments, this number of bins will even be greater than the number of samples contained in a trajectory. Thus, instead of storing the information into a multi-

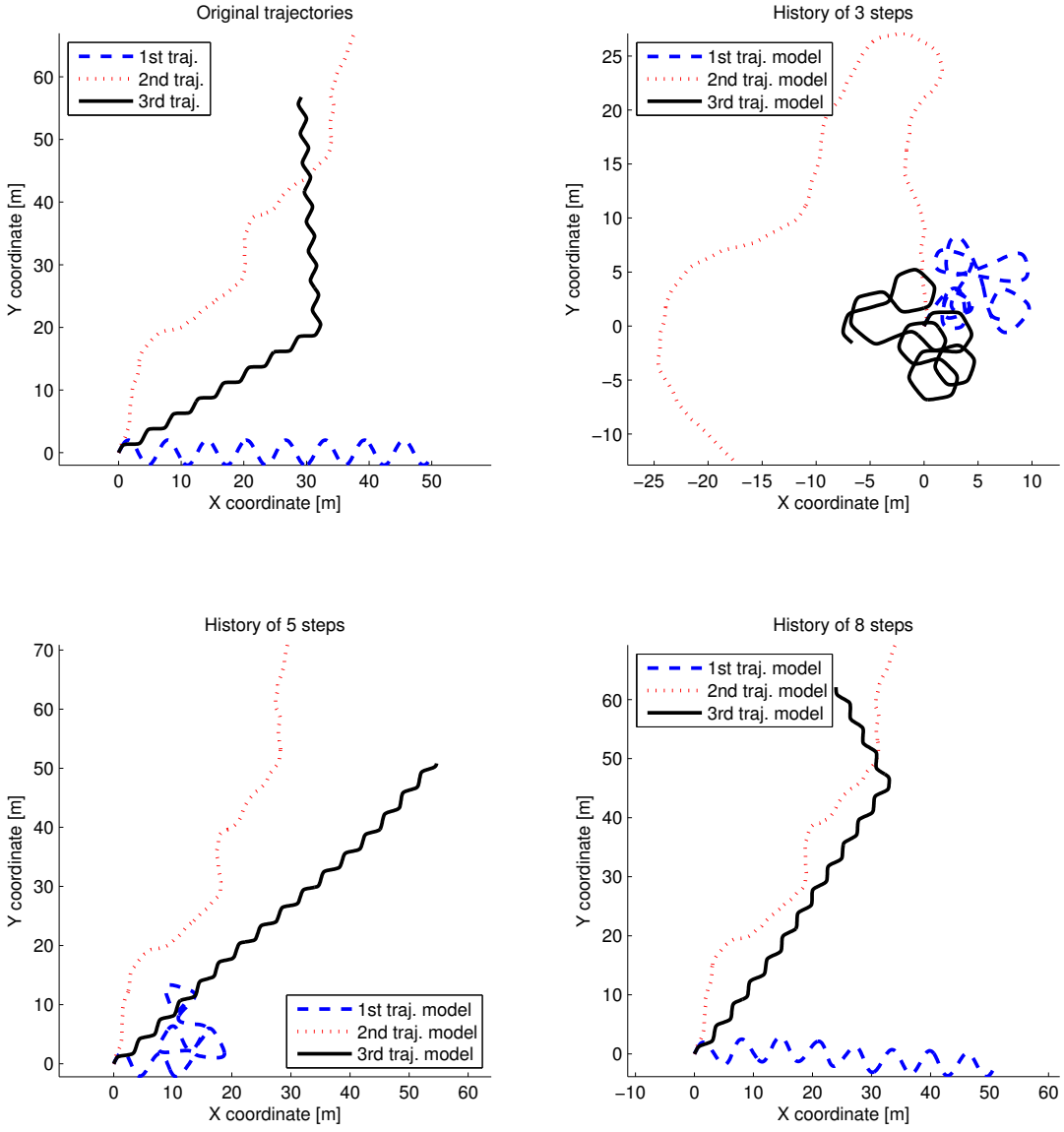


Figure 8.28: Trajectories generated with more than one step of history for the CRW model. Increasing the number of steps of history makes the generated trajectories more similar to the original ones

dimensional matrix containing mainly empty bins, it is easier to store the position in this matrix of the non-empty bins and their content.

The histogram will have $2^{(number\ of\ steps\ of\ history)}$ dimensions as it stores the step length and angle for each step of history. However, the grid of the bin edges can be kept identical for each step of history, as the same kind of data is stored. For our experiment, the same bin edges were used as in Figure 8.27. Suppose that each bin of this histogram is labeled with a letter, each step can be represented by the letter of the bin in which it is falling into. Thus a trajectory can be transformed into a list of bin positions. For example, if the first point of the trajectory falls into bin a and the second point falls into bin b, the trajectory becomes $[ab\ldots]$. This trajectory can then be cut into a list of sequences of two letters, e.g. trajectory $[abcadg]$ is decomposed into $\{ab, bc, ca, ad, dg\}$. The number of occurrences of a particular sequence can be summed and the result corresponds to the number of elements in the bin whose position in the multivariate histogram is this particular sequence. This way of coding can easily be extended to more than two dimensions.

To generate a trajectory, a sequence has first to be randomly chosen, for example the sequence da. Then all the sequences starting with the last letter, a, are summed and one of this sequence is chosen randomly. In the previous example, there was two sequences starting with a: ab and ad. So one of the sequence is chosen, for example ad and the trajectory becomes $[dad]$. The next sequence has to start with d and there is only one choice, dg, for a resulting trajectory $[dadg]$. A problem appears now, as no sequence starts with the letter g. In our experiments, it was considered as a dead end and the generation process was stopped. A solution to this problem would be to select a sequence starting with a letter close to g, but it was not used.

This method can easily be extended to more than two steps of history: in this case, all the letters of the sequence less the last one must be similar, instead of just the first. Increasing the number of steps of history does however reduce the number of elements per histogram bin and thus the capacity to generate different sequences.

Finally, when a series of letters corresponding to a trajectory has been generated, it can be translated back to spatial coordinates, using the edges of the histogram bins. A random coordinates is chosen inside the bin limits using a uniform distribution, like it was performed with the HMM/CRW hybrid model.

Figure 8.28 shows random trajectories generated with this extension of the CRW model with an history of multiple steps. There is clearly no structure visible. With a history of three steps, the generated trajectories are different from the original ones, but they display clear patterns for the first and third trajectory models. The curves are already quite close to the original ones, but are not structured in the same way. With five steps of history, the trajectory generated with the third model becomes quite well structured. It is however more important to notice that the second model generates a trajectory close to an exact copy of the original one. Thus, this model does not model sub-patterns arranged in a correct way, but does contain the whole trajectory information. As a result, it is only able to generate the same trajectory. With 8 steps of history, all the three trajectories generated are the quasi exact copies

of the original ones. Only the trajectory generated with the third model shows that there is still some randomness: the location of the inflection.

It is clear that with a history length corresponding to the number of steps of the trajectory, there will be only one sequence that can be generated: the original ones. It is however more interesting to notice that eight steps of history produce a clear over-fitting of the model and that there is hardly any randomness kept during the generation process.

8.2.4.1 Real robot trajectories

History CRW models can also be applied to trajectories of mobile robots. Figure 8.29 shows the influence of the number of steps of history on the aspect of a randomly generated trajectory. The original trajectory was generated with six robots in the same arena and the acquisition rate was 3 Hz. 4000 trajectory points were stored, even if only the first 500 are displayed in the top left figure. The histogram of the CRW distributions was built with 41 bins for the angle axis, between ± 1 radians, and with 15 bins for the step length axis, between 0 and 25 mm. Each trajectory generated with the history CRW model contains 500 points.

With a history of one step (original CRW model), the generated trajectory has nearly no pattern similar to the original one. On the other hand, with ten steps of history, the trajectory curves share a lot of features with the original ones. Finally, with 100 steps of history, the resulting trajectory is nearly identical to the original one. Most of the main features appears in the same order, even if some angular difference exists, caused by the way the random trajectories are generated.

The over-fitting of the model created with 100 steps of history is evaluated in Figure 8.30. The ratio of the number of non-empty bins divided by the number of sequences is a good measure of the diversity contained in the histogram. With 99 steps of history, this ratio is one. It means that maximum one occurrence exist for each sequence of 99 letters (bin positions). Thus, with a history of 100 steps, there will be at maximum one possible sequence that can be used to pursue the generation. Only the same sequence of bin location can be generated: the sequence of the original trajectory. It is a clear indicator that the model fits so much to the original trajectory, that it is only able to regenerate it.

8.2.5 Discussion

This section showed that the Correlated Random Walk model is a great tool to measure the difference between trajectories without a common frame. For most of the applications, the simple model considering just one step at a time (original CRW model) is sufficient to observe and measure differences between trajectories. In some cases, it is however unable to see the temporal organization of features, and taking into account more than one step of history would be a great asset. However, it is important to keep the number of steps of history into a usable range. A value too small will fail to see differences between the patterns existing in the trajectories. But a too large value, will overfit the model which will focus on the

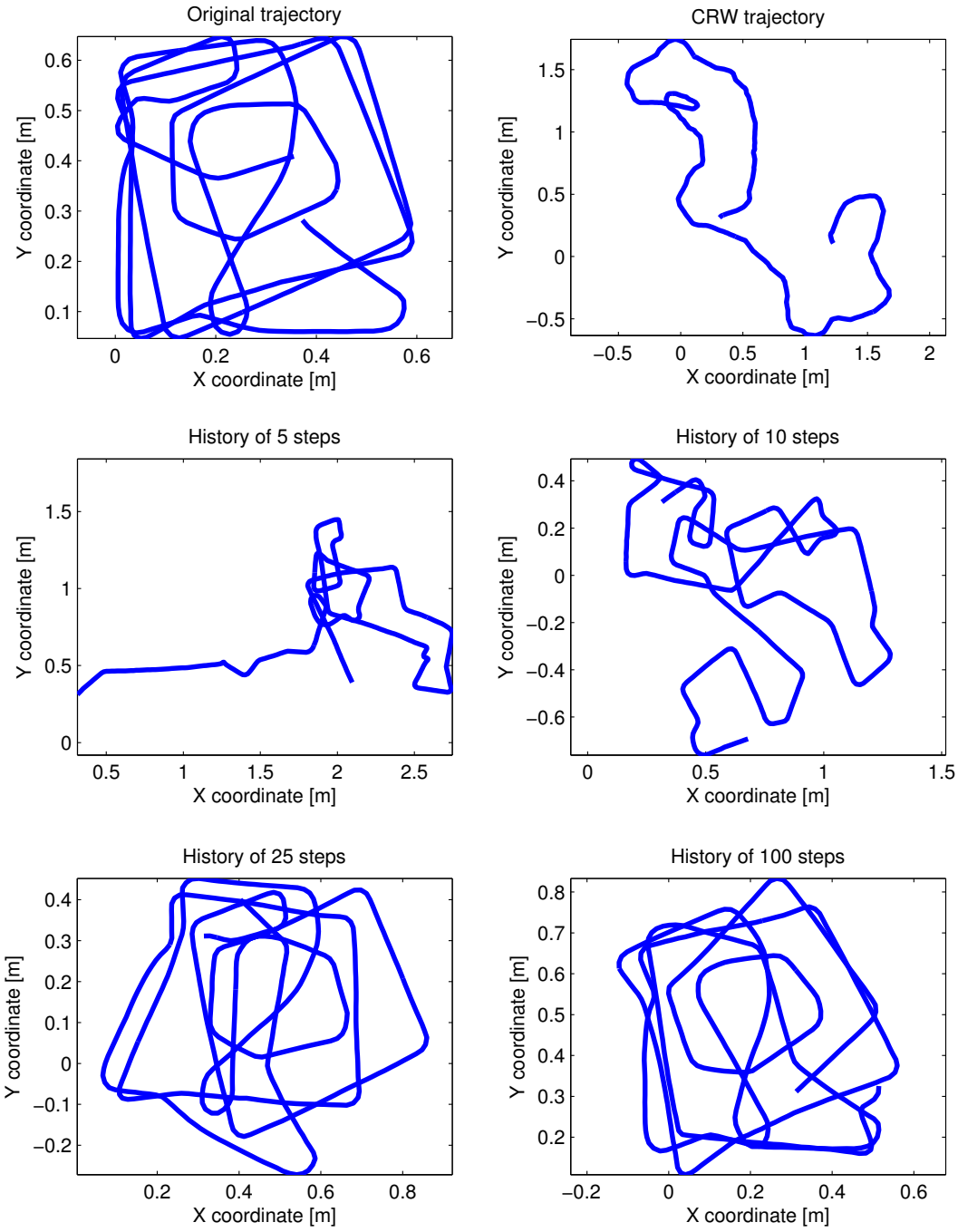


Figure 8.29: Influence of the number of steps of history on the aspect of a randomly generated trajectory. The original CRW model corresponds to a history of one step. It is quite clear that having a lot of steps of history makes the generated trajectory more similar to the original one

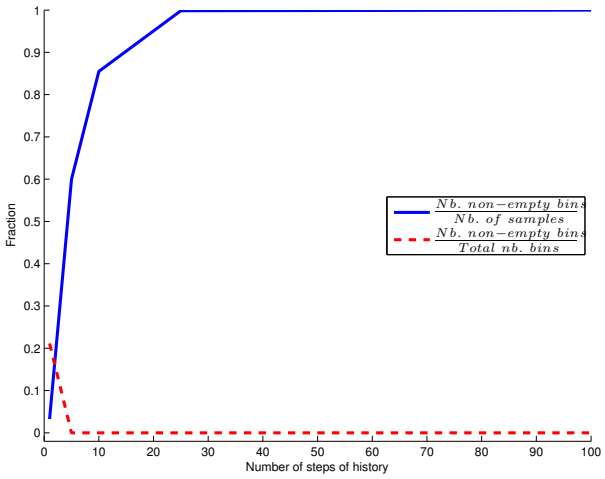


Figure 8.30: The first ratio is an indicator of the diversity contained in the multidimensional histogram. The second ratio indicates how much increasing the number of steps of history increases drastically the proportion of empty bins

exact succession of the features, and thus trajectories displaying similar features in a different order will be considered as different.

8.3 Conclusion

This chapter described two alternative methods to the Point Distribution Model. The first one is the GMM/HMM hybrid model. It is well known and well spread in the research community. A lot of toolbox already exist, and it is thus quite easy to use this solution to measure the difference between two trajectory datasets. Moreover, the trajectories do not need to be characterized by the same number of points. Thus, it is not necessary to resample the trajectories before comparing them. This solution is also able to compare nearly all kinds of trajectories. However, it remains quite limited, as the model is different from the data it represents. It is also slow to train and require a lot of computational resources. For trajectories with a common frame, the Point Distribution Model is more efficient to clearly evaluate differences between trajectories, but needs perhaps more expertise to provide interesting results. Finally, generating random trajectories with the GMM/HMM model clearly shows that this model does not perfectly fit to the data it represents. However, it is still efficient to classify trajectories.

Another method studied in this chapter was the Correlated Random Walk (CRW) model. The Point Distribution Model is clearly not able to compare trajectories without common frame, as they need to be compared as distributions of features. The CRW model offers thus a good alternative. Even in its original form, it is quite able to measure trajectory differences and it remains sufficient for most of the

applications. Measuring the maximal CDF difference between the CRW distributions corresponding to the trajectories is a nice way to get a quantitative comparison of them. In the case where the original CRW model is not sufficient, it can be extended by adding a HMM model on top of it. The likelihood can then be used to get a quantization of the difference. However, a better solution is to take into consideration sequences of steps instead of a unique step, like in the original CRW model. In this case, the maximal CDF difference between the distributions is clearly a good measure of the trajectory difference. However, it is necessary to take into consideration the risk of overfitting the model.

Finally, the two methods presented in this chapter show how efficient the PDM is to analyze trajectories with common frame. They offer also an alternative to the PDM for the analysis of trajectories without common frame, where it does not suit.

Chapter 9

Conclusion

Trajectory analysis is a quite well studied research field. Multiple techniques to measure curve differences have already been developed, such as the Fréchet or the Hausdorff distances, or methods such as the Dynamic Time Warping or the Least Common Subsequence. However, these techniques focus on curve comparison and not on the characterization of behavioral differences between two moving agents, based on their respective trajectories. Most of the time, and more specifically in mobile robotics literature, this behavioral comparison is performed visually using trajectory plots without referring to a quantitative measure. This thesis therefore presents different tools and techniques to quantitatively analyze trajectories. Even if they were originally developed by different scientific communities such as Statistics and Machine Learning, the fundamental novel aspect covered in this thesis is the application and adaptation of these tools to the analysis of trajectories and of the underlying behavioral differences. This thesis thus offers a fair description and comparison of these techniques, based on trajectories of mobile robots. Using robotic platforms as a trajectory generator can also be considered as an original contribution of this thesis, and this approach proved to be a success, as their behavior can be easily modified and their repeatability make possible an acquisition of statistically significant data. Also, a further innovative approach is presented, which applies the Point Distribution Model (PDM) to the analysis of trajectories for the first time. This active shape model was first developed in computer vision to compare spatial shapes in images. This thesis presents the mathematical reformulation of the PDM in order to fit spatiotemporal data such as trajectories of moving agents. It also demonstrates that in the space of the PDM, the inter-cluster distance defined in Section A.5.2 is a quantitative measure of trajectory differences and provides a more scientific solution to the comparison of trajectories than a qualitative visual evaluation.

This thesis also outlines that there is no *plug-and-play* solution to analyze trajectories, as the different techniques and their parameters must fit the application needs. However, four successive stages of the analysis can be distinguished in our method.

- First, the trajectory characteristics used for the analysis (spatial and temporal positions, derivatives,

etc.) must be tailored to the application and to the information which must be extracted. For example, if spatial differences need to be emphasized, the temporal coordinates are a priori not useful.

- The second stage corresponds to trajectory re-sampling. Most of the time, the analysis technique requires a selection of sampled points representing the whole trajectory. For example, PDMs need a constant number of sampled points for all compared trajectories, which is rarely the case with raw data. Given that sampling techniques, which can be space-based or time-based, have a major influence on the resulting analysis, it is important to use them with a full knowledge of their characteristics and to choose their parameters accordingly. Moreover, the density of sampled points must be correctly selected: a too high density increases the amount of data to be treated and also emphasizes acquisition noise, while a too low density leads to a loss of important trajectory features. This thesis presents an original space-based sampling technique, which was developed for trajectories generated on a track, using its border to create sampling gates. This solution is very efficient and can be ported to a lot of applications, such as pedestrians walking in a hallway, cars driving on a road or planes flying in an air lane.

- The third stage is concerned with the analysis techniques used to emphasize the desired trajectory differences. This thesis puts a strong accent on the PDM, which reduces the dimensionality of trajectory datasets, allowing for a fair visualization of trajectory differences. Moreover, the spatial representations of the PDM deformation modes offer a unique understanding of the main deformations observable in the trajectories. As this model is a linear transformation, it is fairly easy to understand, in comparison with more opaque techniques such as Artificial Neural Networks. Moreover, the PDM ranks the dimensions as a function of their importance in the overall variance. Thus, if no difference between two trajectory datasets can be observed in the first few dimensions of the PDM, it means that if dissimilarities exist, they are negligible in comparison to the trajectory variations. Other techniques, such as Correlated Random Walk (CRW) models, GMM/HMM hybrid models, or different statistical tests stand as substitute solutions, when some of the PDM constraints cannot be fulfilled or when the analysis must be focused on another characteristic than the variance. In all cases, analysis results depend completely on the technique that is being used. For example, a simple statistical test, such as the t-test, can be sufficient to observe and measure the difference between two trajectory datasets. However, if no difference can be pointed out, it does not mean that the trajectories are the same, it only means that the analyzed feature is similar (in this example, the t-test proves that the trajectories are distributed around the same mean). Ultimately, no difference can be found between two trajectories if and only if they possess exactly the same sequence of points. Thus, it is mandatory to know which feature must be analyzed before comparing trajectories, in order to choose the technique accordingly. This thesis presents a non-exhaustive but rich set of techniques applied to different examples. Even if some tools, such as the

PDM, give more insight, there is no universal solution.

- The last stage corresponds to the quantitative measure of differences. For most of the tests, such as the statistical ones, this measure is directly included in the test itself. For other techniques, there is a standard measure used in the scientific community, such as the Maximum Likelihood function used with HMMs; however, for PDMs, we had to define an ad-hoc inter-cluster distance measure to quantify trajectory differences.

This thesis also demonstrates that modifications of the robot hardware and software, as well as modifications of the environment are identifiable by the proposed methods. This was proved both with simulated and real experiments. Thus, it is possible to quantify how much the change of software or hardware modifies the generated trajectories. As a result, it is possible to evaluate how tuning simulation parameters yields trajectories more similar to real ones. Trajectory analysis techniques can also be used to evaluate how hardware and software (i.e. controller) modifications to a mobile robot influence the resulting trajectories. As a consequence, different design trade-offs can be evaluated accordingly.

This thesis eventually also describes how to compare, analyze, and identify behaviors based on agent trajectories. However, it is important to understand that only the differences influencing the trajectories can be observed. For example, when looking at a pedestrian trajectory, it is not possible to recognize if he was smiling or not, even if smiling is part of his global behavior. Likewise, if a robot is driven by a rule-based controller and one rule is not used along the trajectory, even if this rule is modified or suppressed, no variation of the behavior can be observed and, thus, trajectory analysis methods fail to identify the difference. Only the parts of the behavior that are expressed in the trajectories can be analyzed.

As a final word, it is important to understand that no automatic solution exists to analyze trajectories. It is really important to select the correct trajectory features and the best suited techniques for analyzing them. To fulfill this goal, a good understanding of the different techniques is as mandatory as a deep understanding of the application. This thesis, which provides a detailed descriptions of multiples techniques, offers a good starting point to approach trajectory analysis problems.

9.1 Future work

As this thesis focuses on trajectory analysis, it shows how behavioral differences can be observed from trajectories. However, it does not attempt to model the internal parameters of the behavior. System identification techniques such as those developed in control theory are more appropriate to this purpose. The work presented in the introduction [83, 84] uses the NAMRAX technique to model the function linking the inputs (mobile-robot sensors) to the outputs (mobile-robot actuators). Note, however, that their analysis can only be performed based on information available within the robot and therefore is complementary to what we have proposed in this thesis. Potentially, a complete analysis of the robot behavior would include both an identified perception-to-action map as well as a trajectory analysis. A

possible solution consists of a NAMRAX model whose inputs would be the distance from the robot to obstacles and whose outputs would be CRW distributions.

Another possible follow-up of this thesis could fill the existing gap between trajectory analysis and signal processing theory. Signal processing is a well studied science, and extending its rules, such as Nyquist-Shanon theorem, to multidimensional data could give us more insight in the research field of trajectory analysis. This thesis demonstrates that trajectory sampling has a major influence on the analysis results and proposes multiple solutions to sample the trajectories correctly. However, extending sampling theory to trajectory data would further improve this research field, as it would result in an even more scientific approach to this problematic.

In robotics, the development of analysis techniques for multi-agent systems should allow for a better understanding of complex systems, as moving agents are rarely alone is their environment. Currently, the different models used by scientists to abstract systems made of multiple mobile robots rarely encompass trajectory features of the different agents. Taking into account explicitly robot movements should improve the quality of abstraction, leading to more powerful analyses of the global system and to better predictions of the future system states.

Finally, this thesis offers different tools and techniques to analyze trajectories, but the most important remaining work is to apply these techniques and spread out their use in the scientific community.

Appendix A

Mathematical background

This chapter is an introduction to the statistical tests, the models and the classification methods used during the experiments. It provides a mathematical background of the methods used.

A.1 Definition

To help the understanding of this chapter, some conventions, terms and tools will be defined hereby.

A.1.1 Statistical hypothesis testing

Statistics is a mathematical field which arose during the 17^{th} century. It followed the work of Blaise Pascal and Pierre de Fermat on probability theory. Johann Carl Friedrich Gauss (1777-1855) was an important actor of this field and introduced the least squares method. Sir Ronald Aylmer Fisher (1890-1962) and Karl Pearson (1857-1936) had also a great influence in this domain and were at the origin of hypothesis testing. The introduction of computers in the 20^{th} century, led to an expansion of the use of statistics and allowed the computation of tests difficult to handle manually.

Statistical hypothesis testing is a method to assess the validity of a given hypothesis, based on a statistical analysis of the data. Null hypothesis testing has for goal to evaluate how well the data fits the possibility that chance factors are the only cause. The null hypothesis (H_0) defines that the observation is only caused by chance. Thus, the goal is to reject this hypothesis and accept the alternate hypothesis that the observation is caused by another factor. For example, if we want to know if one robot is faster than the other, the null hypothesis is that the observed differences of speed are only caused by chance. The goal of the test is to reject H_0 and accept the alternate hypothesis (H_1) that the observed differences are caused by one robot being faster than the other. However, if we cannot reject H_0 with our current data, it does not mean that the robots have the same speed. It means that our observations do not allow us to state with a given insurance (confidence interval) that one robot is faster than the other. Thus, the confidence

interval determines the probability that we correctly reject the null hypothesis. The p-value, equals to one minus the confidence interval, corresponds to the probability that we reject the null hypothesis when it was false to do so. The power of a statistical test corresponds to the probability that the test will reject a false null hypothesis. Thus a test with a low power will less often reject the null hypothesis, when it has to be rejected, than a high power test. As we want to reject the null hypothesis, it is important to use the test with the highest power, if possible.

The following sections will introduce some of the main statistical hypothesis tests for univariate and multivariate data.

A.1.2 Degrees of freedom

Most of the statistical tests use the notion of degrees of freedom."Degrees of freedom are the number of independent pieces of information available to estimate another piece of information"[1]. For example, if we want to compute the mean of a dataset made of seven samples (one dimension per sample), the number of degrees of freedom is equal to the number of independent samples: seven. However, if we want to compute the variance of the same dataset, the number of degrees of freedom will be different. As the variance is defined as the sum of the square difference to the mean, one degree of freedom is used to compute the mean, leaving only 6 degrees of freedom for the variance. To be more explicit, suppose that the dataset contains only one sample, the variance is null and thus has no degrees of freedom. If there are two samples and we know the mean (1^{st} degree of freedom) and the variance (2^{nd} degree of freedom) from one sample to the mean, we know the value of the second sample, as the mean is in the middle of the two samples. Thus, the variance of the dataset contains only one degree of freedom.

In most of the statistical tests, as the variance is used to compute the test values, the number of degrees of freedom of the dataset is $\nu = n - 1$, if the dataset contains n samples.

A.1.3 Mathematical symbolism

For this whole chapter, a specific symbolism has been used to limit confusion between scalars, vectors and matrix. The following rule was applied: x is a scalar, X is a vector and $\boldsymbol{X}$ is a matrix. Greek letters were also used and follow the same symbolism (ω, Ω and $\boldsymbol{\Omega}$).

A.1.4 Variance, covariance and standard deviation

The variance is a measure of the dispersion. Given a dataset $\{x_1, x_2, \ldots, x_n\}$, its variance, v, is defined as:

$$v = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2 , \tag{A.1}$$

[1] www.wikipedia.org

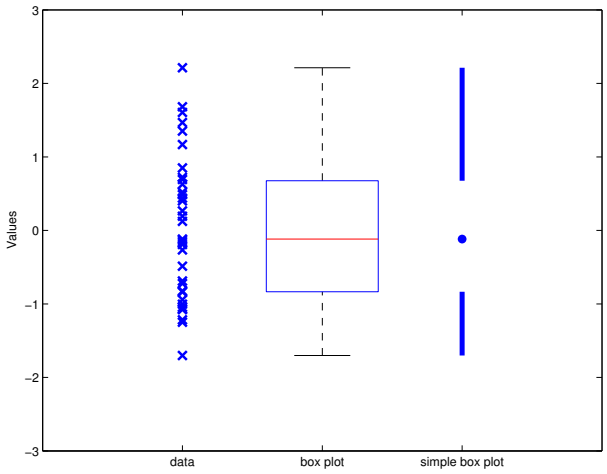


Figure A.1: Box plot and simple box plot of the same dataset proposed by Tufte

where $\bar{x}$ is the mean of the dataset. Another measure commonly used is the standard deviation, σ, defined as:

$$\sigma = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2} = \sqrt{v}. \tag{A.2}$$

Similar measures exist for multidimensional data. The equivalent of the variance is the covariance matrix. Given a dataset $\{X_1, X_2, \ldots, X_n\}$, the covariance matrix $\boldsymbol{V}$ is defined as:

$$\boldsymbol{V} = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \overline{X})(X_i - \overline{X})^T, \tag{A.3}$$

where $\overline{X}$ is the mean of the dataset. The projection of the variance in a given direction, defined by the unitary vector N, is:

$$v_N = N^T \cdot V^{-1} \cdot N. \tag{A.4}$$

And finally, the projected standard deviation in the same direction is:

$$\sigma_N = \sqrt{N^T \cdot V^{-1} \cdot N} = \sqrt{v_N}. \tag{A.5}$$

A.1.5 Box plot

A box plot, also knows as a box-and-whisker diagram, is a convenient way of displaying a dataset. It was invented in 1977 by the American statistician John Tukey. For each dataset, it shows five values: the lowest value, the first quartile, the median, the third quartile and the highest value. It is a great method to show differences between two populations without having an a priori about the distribution (such as the

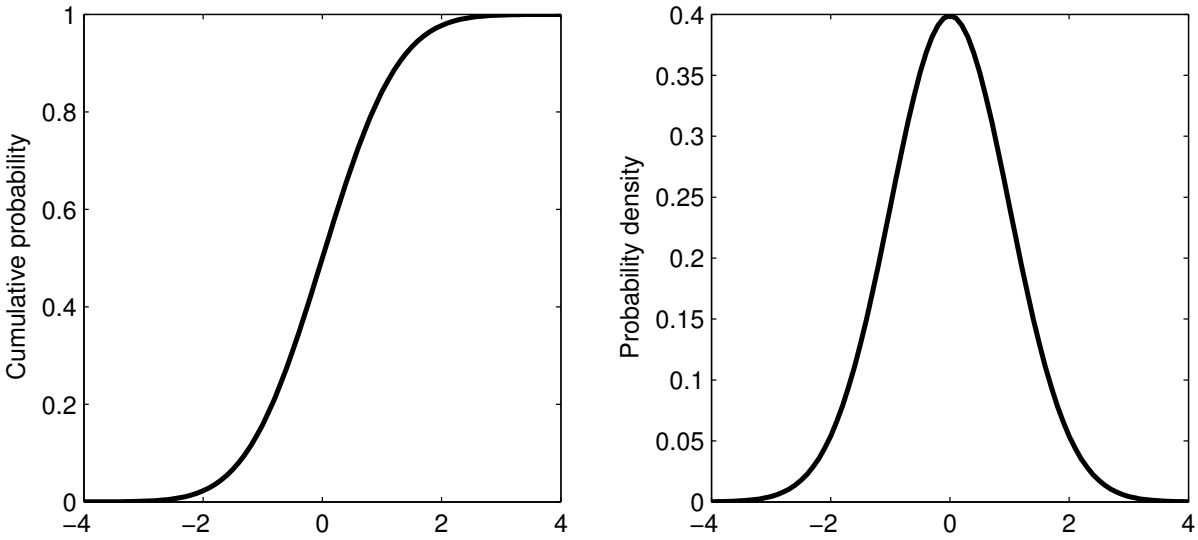


Figure A.2: Normal distribution $\sim \mathcal{N}(0, 1)$: on the left the cumulative probability function and on the right the probability density function

mean and the standard deviation, which are used for centered, compact and symmetric distributions, e.g. Gaussian distributions). Figure A.1 shows a dataset of 35 points and its representation by a box plot. The lower and higher parts of the box correspond to the first and the third quartiles, respectively. The line in the middle of the box represents the median and the whiskers show the distance from the first and third quartiles to the extrema. Edward R. Tufte [100] introduced a simplified version of the box plot, removing all the unnecessary lines to the plot, to focus on the raw data. The median is represented by the point, the quartiles correspond to the extrema of the segments closer to the median and the other segment ends represent the dataset extrema. Most of the time, the box plot shows also the outliers. If a value is farther away from a quartile than 1.5 times the distance between the first and the third quartile, it is considered as an outlier and is not taken into the whiskers. However, this technique will not be used in this thesis and the whiskers will always correspond to the dataset extrema.

A.2 Univariate data

A.2.1 Normal distribution

First, we have to present the normal distribution, as most of the statistical tests are based on it. The normal distribution, also called Gaussian distribution was introduced in 1734 by Abraham de Moivre. A lot of physical measurements can be approximated well by a normal distribution.

$$x \sim \mathcal{N}(\mu, \sigma) \tag{A.6}$$

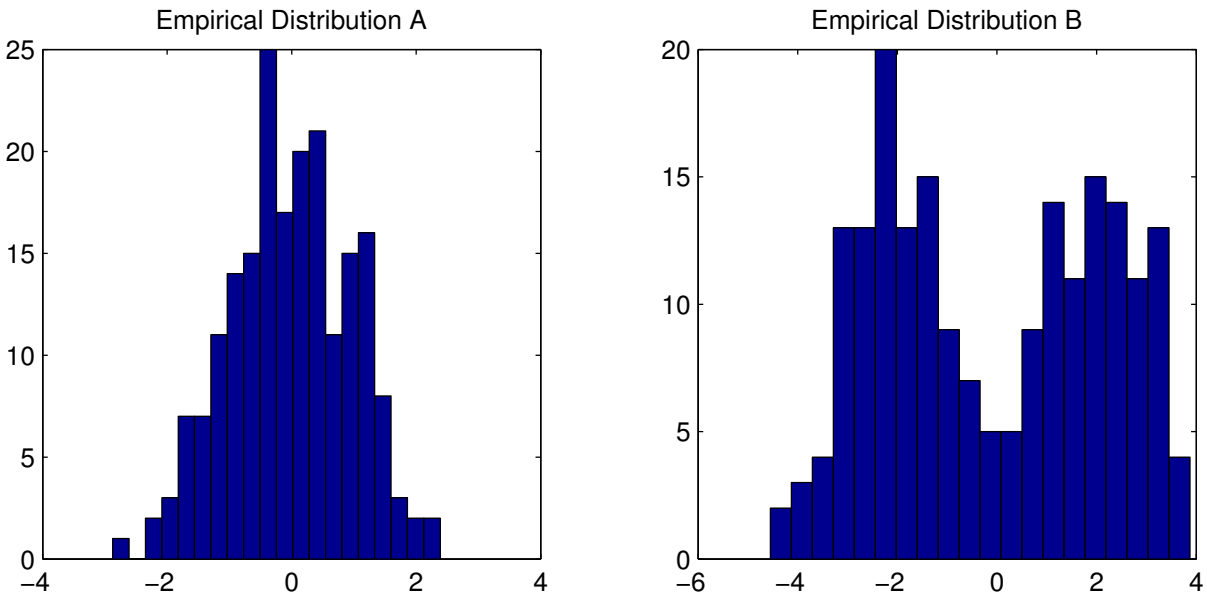


Figure A.3: Histogram of two sets of 200 samples. The first set comes from a normal distribution ($\sim \mathcal{N}(0,1)$), but the second does not. For set B, it is really easy to state from the histogram that it does not come from a normal distribution (not compact)

indicates that a random variable x is normally distributed with mean μ and variance σ^2. The probability density function is:

$$f(x|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}}\, e^{\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)}, \tag{A.7}$$

and the cumulative distribution function (CDF) is:

$$F(x|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{x} e^{\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)}. \tag{A.8}$$

Figure A.2 shows the probability density function and the cumulative distribution function for $\mu = 0$ and $\sigma = 1$.

A.2.2 Normal probability plot

It is not always easy to state if a dataset comes from a normal distribution or not. Most of the time, the evaluation of the normality is done on the dataset histogram. However, this method is really not optimal, as the the number of bins and their size can have a huge influence on this representation of a dataset. Thus, a better method is the Normal Probability Plot. This method is a plot of the cumulative distribution function with a modified y axis, such that the CDF of a normal distribution will lie on a line. Thus, the

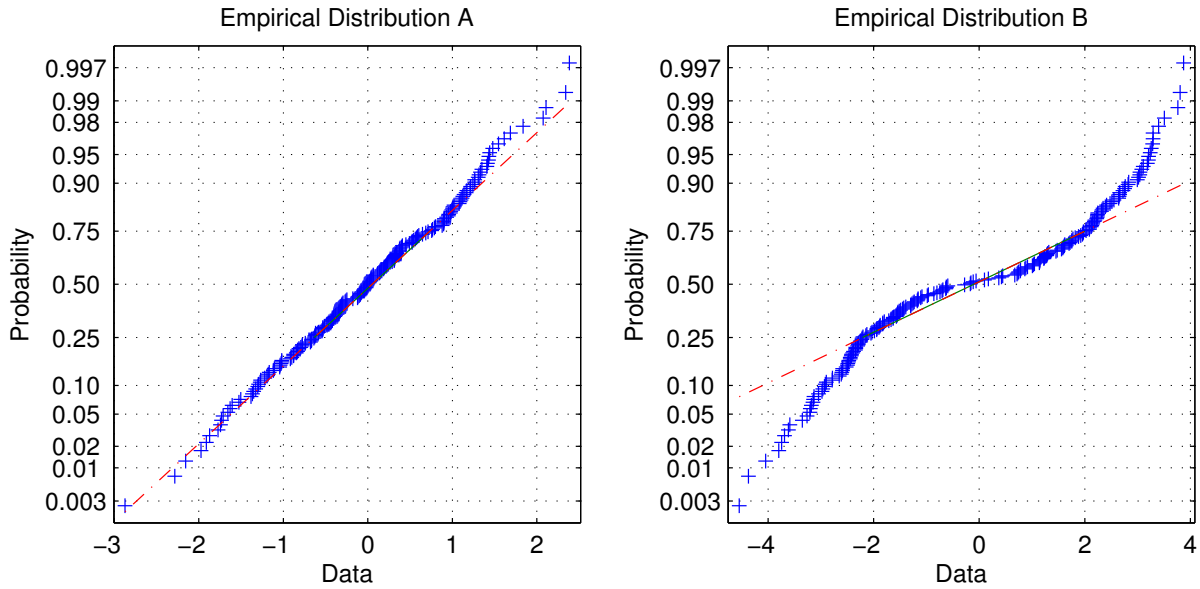


Figure A.4: Normal Probability Plot of the two sets whose histograms are plotted in Figure A.3. We can easily see that the points of the second set does not fall on a line and thus that it does not come from a normal distribution

maximal distance from the line is a measure of the difference between the dataset and the corresponding theoretical normal distribution.

Figure A.3 shows the histograms of two datasets. The first one comes from a normal distribution, but the second does not. Looking at their representation in the Normal Probability Plot (Figure A.4), the points of the second dataset are really not on a line. This implies that the dataset is not normal. By construction, the median value of the dataset will be on the line and for most distributions, the error in the middle part of the distribution is quite small (as most of the time, the distributions are more compact in the center, symmetric and with the maximal value of the Probability Distribution Function (PDF) near the center). Thus, like in this figure, the maximal difference will be on the sides of the plot.

A.2.3 Student's t-test

William Sealy Gosset, better known by his pen name Student, introduced the t-distribution in [101]. The goal of the t-distribution is to evaluate the parameters of a normal distribution with a small number of samples. With more than fifty samples, it is difficult to distinguish the t-distribution from the Gaussian distribution. The t-distribution can be used to compute multiple statistical hypothesis tests.

One of these tests is the Student's t-test. The goal of this test is to demonstrate that two datasets have different means. Thus, the null hypothesis that we want to reject, is that the means of two normally

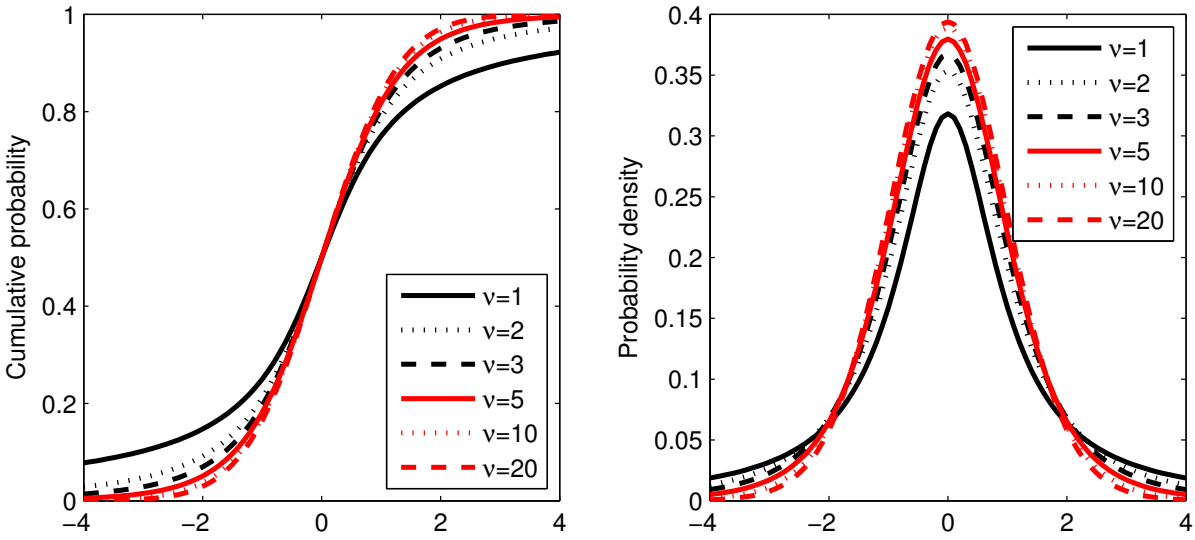


Figure A.5: Student's t-distribution for different degrees of freedom (ν): on the left, the cumulative probability and on the right, the probability density

distributed populations are equal. The two datasets are supposed to be normal with the same variance. If μ_1 and μ_2 are the means of the two datasets, σ_1 and σ_2 are their standard deviation and both distributions have the same number of points, the test value t is:

$$t = \frac{\mu_1 - \mu_2}{\sqrt{\frac{\sigma_1^2+\sigma_2^2}{n}}}. \tag{A.9}$$

If the two datasets do not have the same number of samples (n_1 and n_2 respectively), it becomes:

$$t = \frac{\mu_1 - \mu_2}{\sqrt{\frac{(n_1-1)\cdot\sigma_1^2+(n_2-1)\cdot\sigma_2^2}{n_1+n_2-2} \cdot \left(\frac{1}{n_1}+\frac{1}{n_2}\right)}}. \tag{A.10}$$

The t-value found must be then compared with the cumulative probability of the Student's t-distribution (Figure A.5) or with the Student's t-test tables that can be found in most statistical books. If this t-value is larger than the critical values found in the table for a specific confidence interval, the null hypothesis can be rejected and we can state that the two datasets have different means.

For example, given two datasets, $X_1 = \{-0.4, -1.7, 0.1, 0.3\}$ and $X_2 = \{-1.1, 1.2, 1.2, 0, 0.3\}$, their means are $\mu_1 = -0.425$ and $\mu_2 = 0.32$ respectively, and their standard deviations are $\sigma_1 = 0.90$ and $\sigma_2 = 0.96$ and finally their numbers of points are $n_1 = 4$ and $n_2 = 5$. The computed t-value is $t = -1.19$. If we look into statistical tables, for a one-sided confidence interval of 97.5% (equivalent to a two-sided confidence interval of 95%), for 7 degrees of freedom ($\nu = \nu_1 + \nu_2 = n_1 - 1 + n_2 - 1 = 7$),

the critical value is 2.365. As $|t| < 2.365$, the null hypothesis can not be rejected: the two dataset means can not be statistically distinguished. Looking at the cumulative probability of the t-distribution, for $t = \pm 1.19$, the corresponding p-value is $2 * 13.64\% = 27.28\%$ and the confidence interval is 72.72%. As the two datasets were generated with the normally distributed random generator of Matlab, it is not astonishing that the two datasets are not separable.

In the case where one of the dataset is reduced to one sample (x), the equation becomes:

$$t = \frac{x - \mu}{\sigma} \cdot \sqrt{n}. \tag{A.11}$$

To compute the statistical confidence that a point belongs to a distribution defined by a dataset of points, the t-distribution can be used. For example, if we have a dataset $X = \{-1.1, 1.2, 1.2, 0, 0.3\}$ and a value $x = -1.4$, $\mu = 0.32$, $\sigma = 0.96$ and $n = 5$, thus $t = -4.02$. For 4 degrees of freedom ($\nu = 4$), two-sided test and a confidence interval of 95%, the critical value is ± 2.78. As $|t| > 2.78$, we can reject the hypothesis that x belongs to the same distribution. If we want to compute the boundaries of the confidence interval of 95%, the equation becomes:

$$boundaries = \mu \pm \frac{t_{threshold} \cdot \sigma}{\sqrt{n}}. \tag{A.12}$$

In our case, 95% of the distribution will be between -1.01 and 1.65.

A.2.4 Mann-Whitney U test

The Mann-Whitney U test is the non-parametric counterpart of the Student's t-test. Thus, it does not assume that the two compared datasets are normal. It was introduced by Frank Wilcoxon in [102]. The null hypothesis of this test is that two independent sets have distinct medians.

First, the U statistic must be computed and two methods exist. The first is for small datasets and gives insight about the way this test works:

1. Choose the set of samples containing the lowest values. We call it set_1 and the other one set_2.
2. For each observation in set_1, count the number of elements in set_2 that are smaller (count only one half for each observation that is equal to it).
3. The sum of these counts is u_1. u_2 can be computed in a similar, but a bit more difficult way, as there are more samples in set_1 smaller than the samples in set_2.

For example, if we have two sets $set_1 = [1\ 1\ 3\ 5]$ and $set_2 = [2\ 3\ 3\ 4]$, the counts will be $[0\ 0\ 2\ 4]$, for a total of $u_1 = 6$ ($u_2 = 10$).

The second method is more adapted for bigger sets of samples:

1. Arrange all the samples in the same ranked series.

2. Sum all the ranks of the elements coming from the first set (r_1) and from the second (r_2). The table below shows an example of ranking. If there are multiple elements with the same value, compute the average rank for all the elements (adapted rank). The sum of r_1 and r_2 is:

$$r_1 + r_2 = \frac{n(n+1)}{2}, \tag{A.13}$$

where n is the sum of the size of both sets ($n = n_1 + n_2$).

									Total
Ranked Series	1	1	2	3	3	3	4	5	
Ranks	1	2	3	4	5	6	7	8	36
Adapted Ranks	1.5	1.5	3	5	5	5	7	8	36
set_1	1	1		3				5	
set_2			2	3	3		4		
r_1	1.5	1.5		5				8	16
r_2			3	5	5		7		20

3. U_1 and U_2 can then be computed:

$$u_1 = r_1 - \frac{n_1(n_1+1)}{2} = 6 \tag{A.14}$$

$$u_2 = r_2 - \frac{n_2(n_2+1)}{2} = 10 \tag{A.15}$$

Then, for small datasets, the obtained results can be compared with precomputed tables [103]. Statistical tools, such as Matlab, can also be used to compute it. For our example, we get a p-value of 57.1% and thus we reject the null hypothesis that the two datasets have distinct medians.

For big datasets ($n_1, n_2 > 10$), as an approximation, the statistic can be approximated by a normal distribution:

$$\mu_u = \frac{n_1 \cdot n_2}{2} \tag{A.16}$$

$$\sigma_u = \sqrt{\frac{n_1 n_2 (n_1 + n_2 + 1)}{12}} \tag{A.17}$$

$$z_1 = \frac{|u_1 - \mu_u|}{\sigma_u} = \frac{|u_2 - \mu_u|}{\sigma_u} = z_2 \tag{A.18}$$

As $z_1 = z_2 = z$, it is sufficient to compute just one of these values. Then, z must be compared with a Normal distribution ($\sim \mathcal{N}(0,1)$) to compute its probability in order to reject the null hypothesis (one tail test).

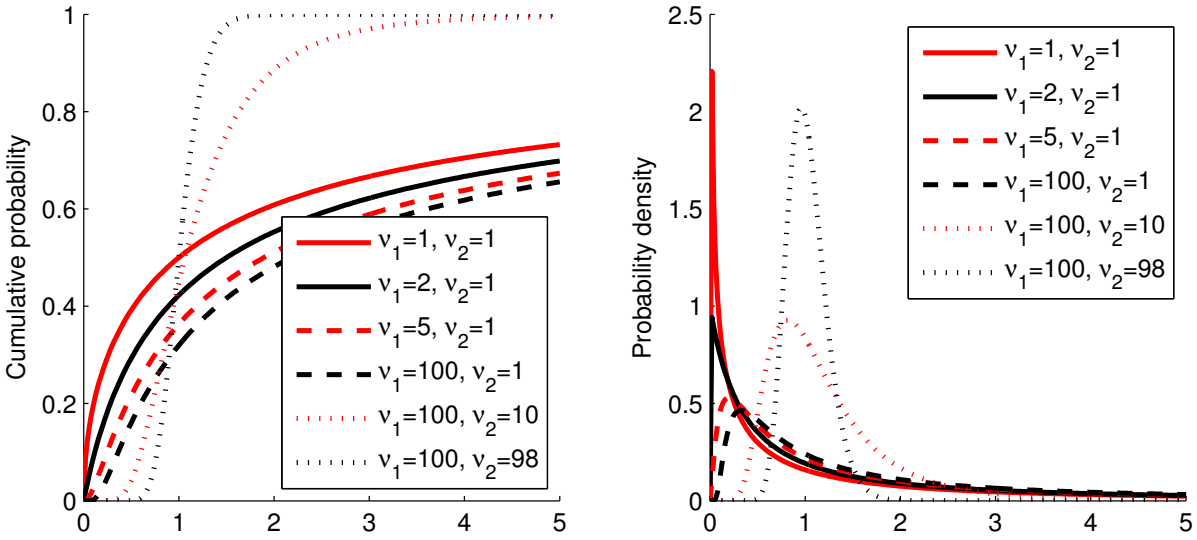


Figure A.6: On the left, the cumulative probability function of the F-distribution for different values of degrees of freedom (ν_1 and ν_2). On the right, the corresponding probability density function

A.2.5 F-test

Sir Ronald A. Fisher introduced in the 1920s a statistic called the variance ratio. Based on it, George W. Snedecor created a test of variance and called it the F-test, in honor of Fisher. If σ_1 and σ_2 are the respective standard deviations of two datasets, n_1 and n_2 are their respective sizes, the null hypothesis is that $\sigma_1 = \sigma_2$. The alternate hypothesis is that $\sigma_1 \neq \sigma_2$, for a two-tailed test. The F statistic is:

$$F = \frac{\sigma_1^2}{\sigma_2^2}. \tag{A.19}$$

The null hypothesis that the standard deviations are equal is rejected at a significance level of α, if one of the following conditions is true :

$$F < F_{(1-\frac{\alpha}{2},n_1-1,n_2-1)} \tag{A.20}$$

$$F > F_{(\frac{\alpha}{2},n_1-1,n_2-1)} \tag{A.21}$$

where $F_{(SL,k,l)}$ is the CDF inverse function of the F-distribution, also called the Fisher-Snedecor distribution, with SL the cumulative probability value, k and l the degrees of freedom. For example, $F_{(0.975,100,98)} = 1.4869$ and $F_{(0.025,100,98)} = 0.6731$. This values can be extracted from the left plot of Figure A.6. The right plot shows the Probability Distribution Function. These values can also be found in statistical tables.

As an example, given two sets, $set_1 = [1\ 1\ 3\ 5]$ and $set_2 = [2\ 3\ 3\ 4]$, $\sigma_1 = 1.91$ and $\sigma_2 = 0.82$. Thus, $F = 5.5$. For a significance level of 5% and a two-tailed test, the two critical values are $F_{(0.975,3,3)} =$

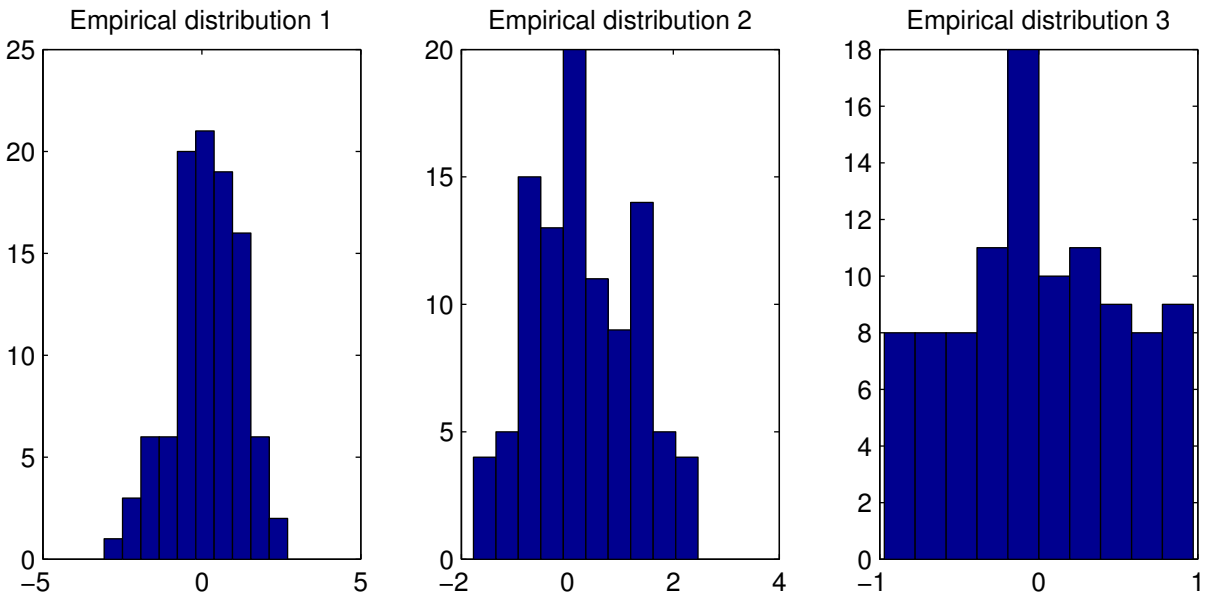


Figure A.7: Histogram of three datasets of 100 samples. The first two datasets come from a normal distribution ($\sim \mathcal{N}(0,1)$) and the third come from a uniform distribution between $[-1 \dots 1]$. It is quite difficult to evaluate from these plots what kind of distributions they are coming from

15.43 and $F_{(0.025,3,3)} = 0.06$. As $0.06 < F < 15.43$, the null hypothesis cannot be rejected and thus there is no statistical difference between the standard deviation of the two sets.

A.2.6 Kolmogorov-Smirnov test

Student's t-distribution is only valid for normal distributions. However, this supposition is not alway true and if the distributions are not known, the Kolmogorov-Smirnov (K-S) test is an excellent alternative. The χ^2 can also be used in this case. However, as the Kolmogorov-Smirnov test is considered as more powerful, the χ^2 will not be presented here. Two variants of the Kolmogorov-Smirnov exist: one and two samples tests.

In the case of the one sample test, the goal of the test is to compare a dataset with a theoretical distribution and to quantify the probability that the dataset has not been generated by the theoretical distribution. For normal or uniform distributions, a histogram is often used to qualify a dataset, even if it is not an efficient way to observe dataset differences. Comparing the cumulative distribution functions is a much more reliable technique. From a dataset made of n observations y_i, the cumulative distribution

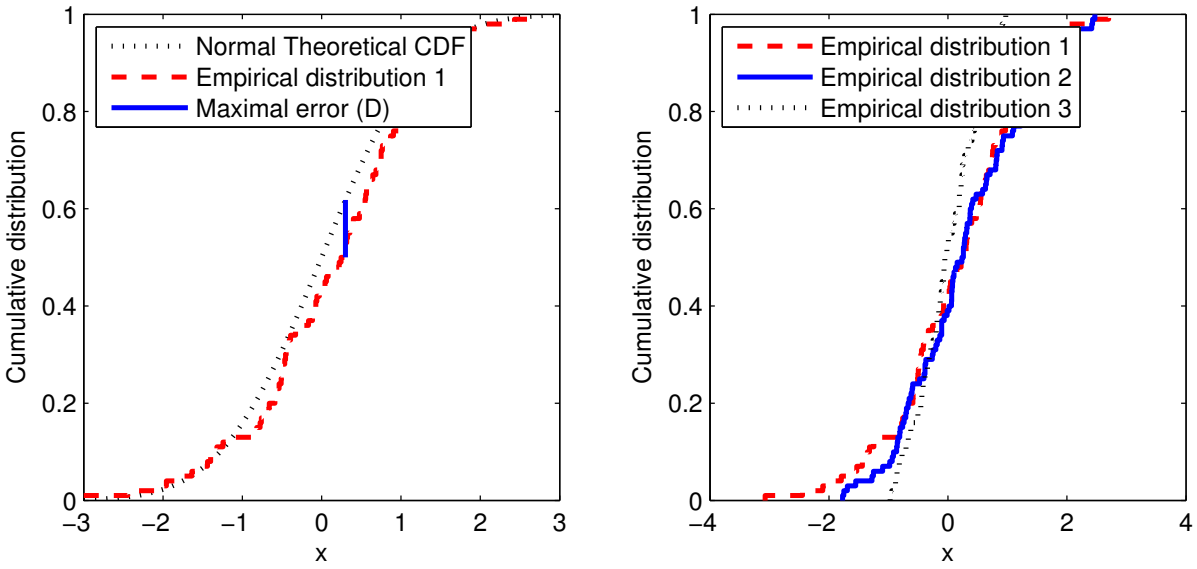


Figure A.8: On the left, illustration of the D statistics used in the K-S test. The first dataset from Figure A.7 is compared to the normal theoretical distribution. On the right, the three empirical datasets of Figure A.7 are compared. It can be spotted that the third is clearly different than the first two and that it is not normal

can be approximated in this way:

$$F_n(x) = \frac{1}{n}\sum_{i=1}^{n}\begin{cases} 1 & \text{if } y_i \leq x, \\ 0 & \text{otherwise.} \end{cases} \tag{A.22}$$

K-S test bases the comparison of datasets on the difference of their cumulative distribution functions $F_1(x)$ and $F_2(x)$, and computes the maximal difference between them:

$$D = max(|F_1(x) - F_2(x)|) \tag{A.23}$$

The D statistics of Equation A.23 is then compared with a table such as those provided by Smirnov [104], Massey [105] or Miller [106]. It must be pointed out that comparing datasets using their empirical cumulative distributions allows a better observation of the differences, than using their histograms.

Figure A.8 shows the use of the K-S test with the three datasets presented in Figure A.7. On the left the first dataset is compared with a theoretical normal distribution, and the D statistics (maximal difference of the CDFs) used in the K-S test is also shown. On the right plot, the three cumulative distribution functions of the datasets are plotted. If we apply the K-S test to these sets, the hypothesis that the third dataset is normal can be rejected, with a p-value of 0.61%. For the first and the second sets, this hypothesis can not be rejected with a p-value of 11.5% and 8.62% respectively.

Table A.1: Resulting p-value of the K-S test when comparing the three datasets of Figure A.7. When the p-value is smaller than 5%, the hypothesis that the two sets come from the same distribution can be rejected. Thus, the third set is clearly different from the two others

	Set 1	Set 2	Set 3
Set 1	1	0.89	0.008
Set 2	0.89	1	0.003
Set 3	0.008	0.003	1

The K-S test can also be used to compare two datasets. Table A.1 shows the p-values resulting of the comparison of the datasets shown in Figure A.7. From theses p-values, we can assess that the third dataset is different from the two others and the the first two are similar.

A.3 Multivariate data

If there is more than one measure for one sample, the sample is multivariate. For example, if we take a crate of apples and for each apple we measure its weight and size, we are working with multivariate data. If the different measures are independent (i.e. knowing the value of one measure gives no hint about the other measure), they can be analyzed independently. For our example, it is obviously not the case, as a bigger apple will weight more. Thus, it is important to analyze dependent measurements together.

A.3.1 Multivariate normal distribution

Figure A.9 represents the probability distribution of a bivariate independent normal distribution ($\mathcal{N}_2(0,1)$). It means that the standard deviation is 1 for the two dimensions. Figure A.10 shows a bivariate dependent normal distribution. In this case, knowing where you are on axis x_1 gives you a hint about the probability to be at a given place on axis x_2.

For a multivariate normal distribution, the probability distribution function is:

$$f_{X(x_1,\ldots,x_n)} = \frac{1}{(2\pi)^{\frac{n}{2}}|\boldsymbol{V}|^{\frac{1}{2}}} e^{\left(-\frac{1}{2}(X-M)^T\boldsymbol{V}^{-1}(X-M)\right)} \tag{A.24}$$

where $\boldsymbol{V}$ is the covariance matrix, M is the mean vector of the distribution and n is the number of dimensions.

For a bivariate case and an identity covariance matrix $\left(\boldsymbol{V} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\right)$, the probability to have a random sample inside a circle of radius r is:

$$P = \iint_{Circle} f_{X(x,y)} dx\, dy = 1 - e^{-\frac{1}{2}r^2} \tag{A.25}$$

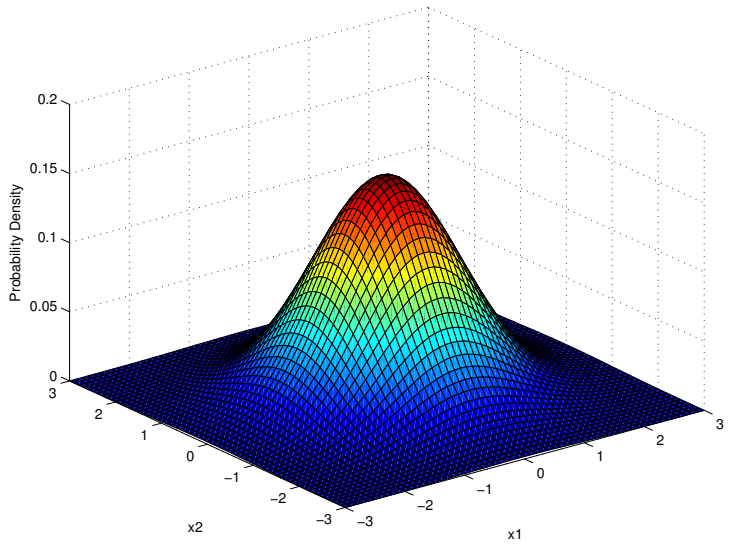


Figure A.9: Example of a bivariate independent normal distribution ($\mathcal{N}_2(0,1)$). The graphic shows the resulting probability distribution for two independent variables x_1 and x_2

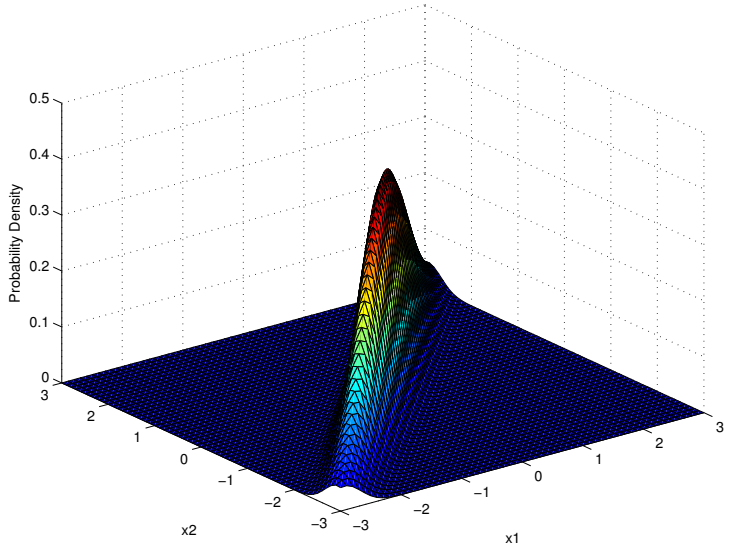


Figure A.10: Example of a bivariate dependent normal distribution. If $x_1 = -3$, the probability that $x_2 = -3$ is much greater than if $x_1 = 3$. Knowing x_1 gives a hint about the distribution of x_2

A.3.2 Hotelling's T^2 statistic

Hotelling's T^2 statistics, introduced by Harold Hotelling [107, 108], is the generalization of Student's t statistics to multivariate data. It is defined as:

$$t^2 = n(X - M)^T \boldsymbol{W}^{-1}(X - M) \tag{A.26}$$

where n is the number of points of the dataset, X is a column vector of p elements (number of dimensions) and $\boldsymbol{W}$ is a pxp matrix.

If $X \sim \mathcal{N}_p(M, \boldsymbol{V})$ is a random variable with a multivariate normal distribution and $\boldsymbol{W} \sim W_p(n - 1, \boldsymbol{V})$ has a Wishart distribution with the same non-singular covariance matrix $\boldsymbol{V}$ and n is the number of vectors in the dataset, then the distribution t^2 is $T^2(p, n - 1)$, a Hotelling's T^2 distribution with parameters p and $n - 1$. It can be demonstrated that:

$$\frac{n - p}{p(n - 1)} T^2 \sim F_{p,n-p} \tag{A.27}$$

where F is a F-distribution. Figure A.6 shows examples for the F-distribution for various degrees of freedom (In Equation A.27, $k = p$ and $l = n - p$).

Hence, for a multidimensional dataset coming from a multivariate normal distribution, the probability that a vector belongs to the dataset can be computed using the the F-distribution with n the number of vectors in the dataset (or the number of degrees of freedom of the dataset) and p the number of dimensions of the vectors.

A.3.2.1 Hotelling's two samples T^2 statistic

Harold Hotelling extends also the t-test to compare two multivariate normal datasets ($[X_1, \ldots, X_{n_x}]$ and $[Y_1, \ldots, Y_{n_y}]$).

We can compute the unbiased pooled covariance matrix estimate:

$$\boldsymbol{W} = \frac{\sum_{i=1}^{n_x}(X_i - \overline{X})(X_i - \overline{X})^T + \sum_{i=1}^{n_y}(Y_i - \overline{Y})(Y_i - \overline{Y})^T}{n_x + n_y - 2} \tag{A.28}$$

Then Hotelling's two samples T^2 statistic is

$$t^2 = \frac{n_x n_y}{n_x + n_y}(\overline{X} - \overline{Y})^T \boldsymbol{W}^{-1}(\overline{X} - \overline{Y}) \sim T^2(p, n_x + n_y - 2), \tag{A.29}$$

and it can be related to the F-distribution (figure A.6) by

$$\frac{n_x + n_y - p - 1}{(n_x + n_y - 2)p} t^2 \sim F(p, n_x + n_y - 1 - p). \tag{A.30}$$

From this F-distribution, we can compute the probability that the two datasets have distinct means. This is very similar to the univariate t-test (Section A.2.3), as we divide the vector relying the two dataset means by the pooled covariance, instead of dividing the difference of means by the pooled variance for the t-test.

A.3.3 Multivariate Kolmogorov-Smirnov test

Hotelling's T^2 has the same flaw than the Student's t-test: it only compares the distribution means. Thus, even if the distribution shapes are completely different (e.g. comparing a triangle and square distribution), if the distributions have the same mean, the test will fail to see a difference. Thus, as the Kolmogorov-Smirnov test is well adapted to univariate data, finding a multivariate equivalent of it is really useful.

In 1983, Peacock presented a method to extend the Kolmogorov-Smirnov test to two dimensions. In one dimension, the K-S test is based on the maximal difference between the two cumulative distribution functions. If a dataset is compared with a theoretical distribution, this distance will be the maximal difference between the theoretical CDF and the empirical CDF. The K-S test computes this difference only at the points of the dataset. If two experimental datasets are compared, then the difference between the two empirical CDF will be used. In one dimension, computing the CDF is straightforward: the probability distribution function is integrated till the desired position. For a dataset, the CDF is just the number of points in the left or the right side of the desired position, divided by the total number of points. Moreover, the CDF difference will be the same if the integrals are computed from $+\infty$ or $-\infty$.

However, in two dimensions, the CDF absolute difference is not the same if the CDF are computed from the four extrema of the 2D space ($[-\infty, -\infty]$, $[-\infty, +\infty]$, $[+\infty, -\infty]$ and $[+\infty, +\infty]$). Thus, Peacock proposed to compute the four CDF in the four quadrants delimited by a point. Figure A.11 shows an example of these four quadrants. In each quadrants, the CDF of a distribution corresponds to the integral of the PDF on this quadrant and for a dataset, the CDF is approximated by the number of points of the dataset in the quadrant divided by the total number of points. In the Peacock version, these CDF are computed for the combination of all the x and y coordinates of all the points (for the x coordinate of a point, this computation is done at the y coordinates of all the points). The complexity of the Peacock algorithm is in n^3 and thus needs a lot of computation power. In the version presented by Fasano and Franceschini in [109], they propose to compute these CDF only at the points of the dataset(s). The resulting complexity is reduced to n^2. Considering the dataset $\{\{1, 2\}, \{3, 4\}\}$, Fasano and Franceschini method computes the CDF difference at the positions $\{1, 2\}$ and $\{3, 4\}$ and Peacock method computes it at the positions $\{1, 2\}$, $\{1, 4\}$, $\{3, 2\}$ and $\{3, 4\}$. In [110], Lopes et al. compared both algorithms and assessed that they are approximately as powerful to distinguish dataset differences. Thus, Fasano and Franceschini algorithm can be used instead of the Peacock algorithm.

When the maximal difference of CDF is found for all the desired positions (D_{BKS}), this value is then weighted by the number of points (n) to get the critical value (Z_n) :

$$Z_n = D_{BKS}\sqrt{n}. \tag{A.31}$$

When comparing two datasets, this equation becomes:

$$Z_n = D_{BKS}\sqrt{\frac{n_1 n_2}{n_1 + n_2}}, \tag{A.32}$$

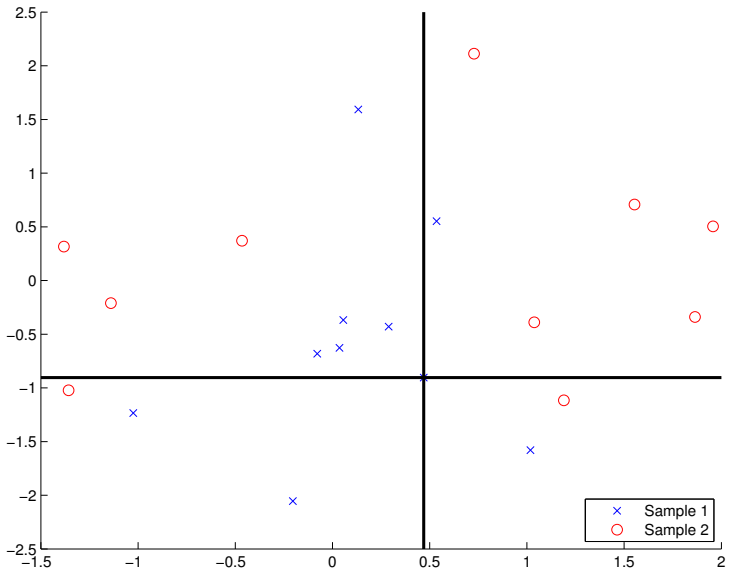


Figure A.11: Graphic explaining how the difference of cumulative distribution functions is computed for the Fasano-Francechini extension of the Kolmogorov-Smirnov test. At the location of a sample, the number of points of a dataset in each quadrant is counted and divided by the total number of points. Only the maximal difference of CDF between the two datasets is then used. As an example, in the top right quadrant, we have 5 points from a dataset and only one from the other one. The respective CDF are 0.5 and 0.1 for a difference of 0.4

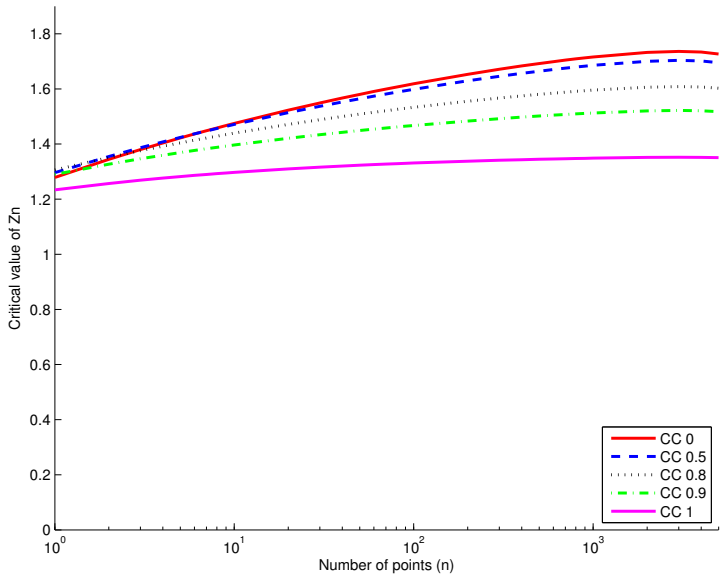


Figure A.12: Critical values of the Fasano-Franceschini test for a significance level of 95% and for different values of correlation coefficient. If the value Z_n computed for given data is bigger than this limit, the hypothesis that the distributions are similar is rejected with a significance level of 95%. The number of points or the correlation coefficient does not have a major influence on the critical value of Z_n

with n_1 and n_2 the numbers of points of the two datasets.

The critical value Z_n can then be compared with the values in Figure A.12, or with the tables computed by Fasano and Franceschini in their article [109]. The correlation coefficient of the dataset is also taken into account by this test. For a two samples test, if the correlation coefficients of both datasets is not too different, the mean value of these coefficients can be used. In this case, $n = \frac{n_1 n_2}{n_1+n_2}$ must be taken.

An extension to the third dimension is also presented in [109].

A.4 Statistical tests summary

The table hereunder is provided as a summary of all the statistical tests presented in this chapter and helps the reader to have a general picture of the uses of the different tests.

	Univariate	Multivariate
Means difference	Student's t-test Section A.2.3	Hotelling's T^2 test Section A.3.2
Medians difference	Mann-Whitney U test Section A.2.4	
Variances difference	F-test Section A.2.5	
CDF difference	Kolmogorov-Smirnov Section A.2.6	Fasano-Franceschini Section A.3.3

A.5 Measure of distance

The most standard distance between two points X and Y is the Euclidean distance:

$$d = \sqrt{(X-Y)^T \cdot (X-Y)}. \tag{A.33}$$

However, when measuring a distance from a known distribution (e.g. Gaussian distribution), dividing this distance by the standard deviation provides a better measure, as it is linked to the distribution and not to the metric used. For example, the $\pm 3\sigma$ rule in statistical control of industrial production is based on the standard deviation of the distribution.

A.5.1 Mahalanobis distance

For multivariate data, the standard deviation is replaced by the covariance matrix of the distribution. In [111], Mahalanobis introduced a new measure (r) of the distance from a point X to a distribution

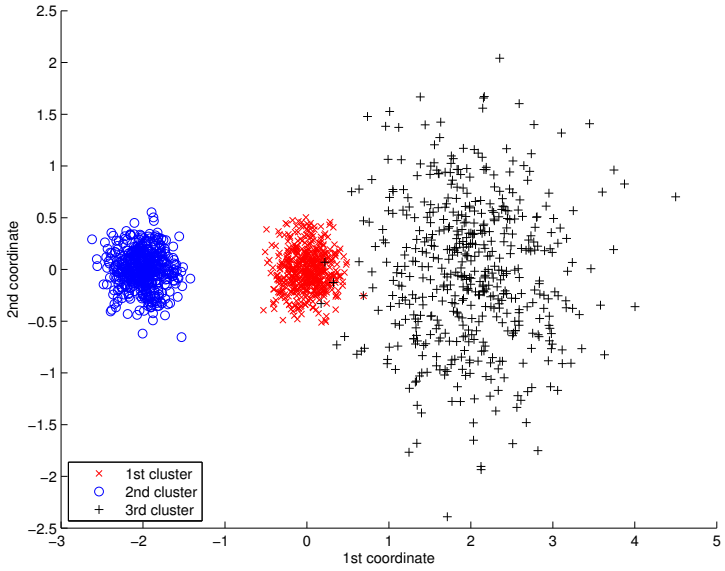


Figure A.13: Three clusters of points used to compare the Euclidean and the inter-cluster distances. The distance measured can be found in Table A.2

(cluster) of points $[Y_1 \dots Y_n]$, taking into account the covariance matrix of the cluster $\boldsymbol{V}_Y$ and the cluster mean $\overline{Y}$.

$$r = \sqrt{(X - \overline{Y})^T \cdot \boldsymbol{V}_Y^{-1} \cdot (X - \overline{Y})} \tag{A.34}$$

In one dimension, the Mahalanobis distance corresponds to the distance from the cluster mean to the sample, divided by the standard deviation. For multivariate data, instead of using the standard deviation, the vector from the sample to the cluster mean is divided by the cluster covariance in the direction of this vector.

The Mahalanobis distance can also be seen as a transformation of space, dividing all the vectors by the cluster variance. Thus, in the new space, the cluster will have an identity matrix as variance. If we make a link with Equation A.25, for a normally distributed cluster, the points at the same distance from the cluster mean have the same probability to belongs to the cluster.

In the original space, the points at the same distance from a cluster form an ellipsoid.

A.5.2 Inter-cluster distance

As a measure of distance between two clusters, we can use a modification of the Mahalanobis distance using their pooled covariance $\boldsymbol{W}$. If $[X_1 \dots X_{n_x}]$ and $[Y_1 \dots Y_{n_y}]$ are the points forming the first and

Table A.2: Euclidean and inter-cluster distances between the clusters shown in Figure A.13

	1st cluster		2nd cluster		3rd cluster	
	Euclid.	Inter-cl.	Euclid.	Inter-cl.	Euclid.	Inter-cl.
1st cluster	0.0	0.0	2.0	9.9	2.0	4.1
2nd cluster	2.0	9.9	0.0	0.0	4.0	8.3
3rd cluster	2.0	4.1	4.0	8.3	0	0

respectively the second cluster,

$$\boldsymbol{W} = \frac{\sum_{i=1}^{n_x}(X_i - \overline{X})(X_i - \overline{X})^T + \sum_{j=1}^{n_y}(Y_j - \overline{Y})(Y_j - \overline{Y})^T}{n_x + n_y - 2}. \tag{A.35}$$

Thus, similarly to Eq. A.34, the distance d between the two clusters can be calculated as:

$$d = \sqrt{(\overline{X} - \overline{Y})^T \cdot \boldsymbol{W}^{-1} \cdot (\overline{X} - \overline{Y})}. \tag{A.36}$$

d can be linked to the Hotelling's T^2 statistics (Section A.3.2, compare Equation A.26 and Equation A.36):

$$t^2 = \frac{n_x \cdot n_y}{n_x + n_y} d^2 \tag{A.37}$$

If the clusters are following multivariate Gaussian distributions, Hotelling's T^2 statistics can be used to compute the probability that two clusters have not the same mean or that a trajectory belongs to a cluster.

Figure A.13 shows three bivariate Gaussian clusters. These clusters will be used to demonstrate the interest lying in the measure of distance which has just been introduced. Table A.2 shows the Euclidean and the inter-cluster distance between the three clusters. Looking at the clusters, it is obvious that the first cluster is closer to the third than to the second. It is however not well represented by the Euclidean distance, as the three clusters are equally separated. However, the inter-cluster distance introduced in this section gives a measure more in accord with the visual perception.

A.6 Dimensionality reduction techniques

The problems handled in classification or pattern recognition often contain a high number of dimensions in the extracted feature vectors. In sound or image classification, a huge amount of data has to be treated. As complex models and techniques, such as Hidden Markov Models or Gaussian Mixture Models, require a lot of computation, reducing the dimensionality of the system decreases the needs in both computation time and memory resources to acceptable levels, considering the constraints in the realization of the recognizer. Thus, the Principal Component Analysis and the Linear Discriminant Analysis are presented hereby, as they are major techniques used in classification.

A.6.1 Principal component analysis

Suppose $[X_1 \dots X_k]$ a set of vectors in $\Re^n$ and $\overline{X}$ its mean, the covariance matrix $\boldsymbol{V}$ of the dataset is:

$$\boldsymbol{V} = \frac{1}{k-1}\sum_{i=1}^{k}(X_i - \overline{X})(X_i - \overline{X})^T = \boldsymbol{P}\boldsymbol{\Lambda}\boldsymbol{P}^{-1}, \tag{A.38}$$

where $\boldsymbol{P} = [P_1 \dots P_r]$ is the matrix of the eigenvectors, $\boldsymbol{\Lambda}$ is the diagonal matrix containing the eigenvalues of $\boldsymbol{V}$ and $r = argmin(n, k-1)$ is the maximal dimensionality of the dataset, if all the vectors are independent. Each vector of the dataset X_i can be transformed into linear combination of modes B_i, and inversely:

$$X_i = \overline{X} + \boldsymbol{P}B_i \tag{A.39}$$

$$B_i = \boldsymbol{P}^{-1}\left(X_i - \overline{X}\right). \tag{A.40}$$

The transformation from the original space (X_i) to the space of the deformation modes (B_i) corresponds to the Principal Components Analysis (PCA) introduced by Jackson [112], also called Karhunen-Loève transform. If λ_i are the sorted eigenvalues of $\boldsymbol{\Lambda}$, the cumulative energy c_d of the d first eigenvectors of the PCA can be defined as:

$$c_d = \frac{\sum_{i=1}^{d}\sqrt{\lambda_i}}{\sum_{i=1}^{r}\sqrt{\lambda_i}}. \tag{A.41}$$

This cumulative energy corresponds to the proportion of the variance contained in the first d dimensions of the PCA. It is frequent that 90% of the variance is contained in the first 3-4 dimensions and thus the system can be reduced to these 3-4 dimensions with small losses. The square root of the eigenvalues are used, because they have the same unit as the covariance matrix, which is squared. As an intuitive explanation, think about the standard deviation which is the square root of the variance.

A.6.1.1 Practical use

At first, we will provide a simple graphical example of an application of the PCA. The Figure A.14 on the left, shows 100 points coming from two bivariate Gaussian distributions with the same variance of 1 and respective means of $[5, 5]$ and $[-1, -3]$. Both formed clusters are really easy to distinguish. Figure A.14 on the right, shows the result of the PCA applied to all the points of both clusters. In this case, the first mode axis is nearly parallel to the axis going through both cluster means. Also, in this case, this axis is close to the best axis separating both clusters. The separability performance of the PCA is only valid, when the distance between the clusters is much bigger than the intra-cluster variability (like in this case). Moreover, as the PCA is just an orthonormal transformation, the distances are kept unmodified. Thus, we can observe that the distance between the two cluster means is identical in the original space and in

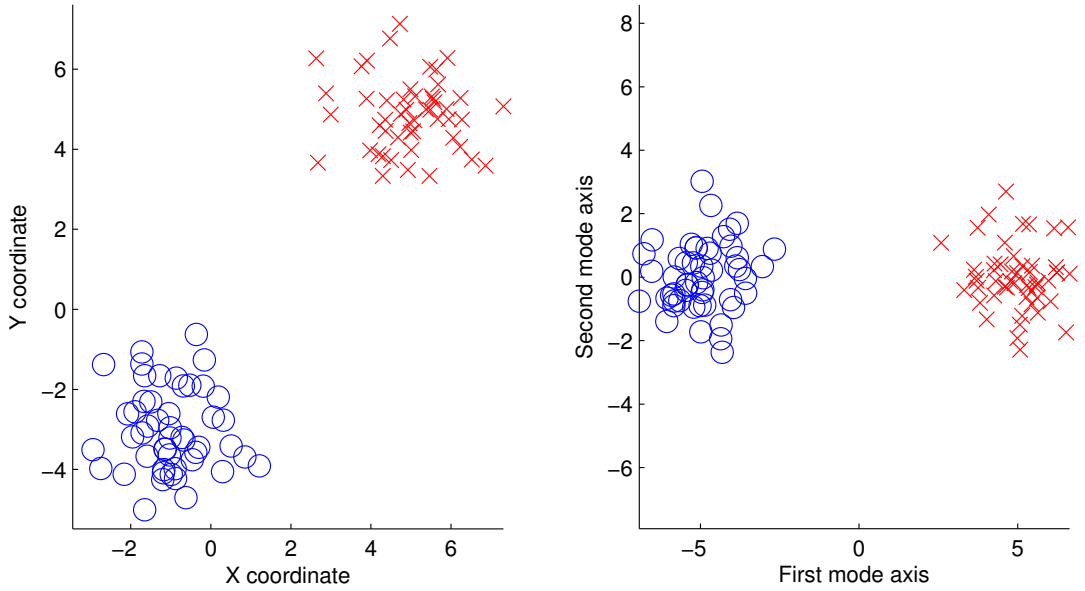


Figure A.14: PCA applied to two Gaussian clusters of 50 points each. On the left, the points in the original space and on the right, their contribution to the first two modes of the PCA

the space of the PCA modes. This characteristics is very useful as the distances in the PCA space will be in the same metric as in the original space. However, when multiple data are mixed, such as temporal and spatial position, it becomes a problem.

Finally, in most of the practical applications, the dimensions with the smallest variance are mainly the result of noise and thus can be discarded. Thus, the system can be reduced to the dimensions containing variance (or energy). In classification, PCA is often used as the first dimensionality reduction technique, followed sometimes by a LDA and afterward, by more complex models, such as Hidden Markov Models or Gaussian Mixture Models.

The PCA is also the core of the Point Distribution Model, presented in Chapter 3.

A.6.2 Linear discriminant analysis

Ronald Fisher presented in [113] the Linear Discriminant Analysis (LDA). This method aims to find the axis of maximal separation between multiple datasets, and is thus supervised. It is different than the best hyperplane of linear classification, even if most of the time, this plane will be orthogonal to it. The Principal Components Analysis works similarly, excepts that it searches for the axis of maximal variance. To get the best separation axis, the goal is to maximize the ratio of the variance between the classes to the variance within classes. If $\overline{X}$ and $\overline{Y}$ are the means of the two datasets, $\boldsymbol{V}_X$ and $\boldsymbol{V}_Y$ are their respective

covariance matrices, and L is the vector of projection we are optimizing, this ratio r becomes:

$$r = \frac{\sigma^2_{beetween}}{\sigma^2_{within}} = \frac{\left(L^T(\overline{X} - \overline{Y})\right)^2}{L^T(\boldsymbol{V}_X + \boldsymbol{V}_Y)L}. \tag{A.42}$$

It can be shown that this ratio is maximized when:

$$L = (\boldsymbol{V}_X + \boldsymbol{V}_Y)^{-1}(\overline{X} - \overline{Y}). \tag{A.43}$$

If both datasets have the same number of points, the pooled covariance as defined in Equation A.35 is equal to:

$$\boldsymbol{W} = (\boldsymbol{V}_X + \boldsymbol{V}_Y). \tag{A.44}$$

To show the link with the inter-cluster distance, if the projection vector $L = (\overline{X} - \overline{Y})$, the ratio becomes:

$$\begin{aligned} r &= \frac{\left((\overline{X} - \overline{Y})^T(\overline{X} - \overline{Y})\right)^2}{(\overline{X} - \overline{Y})^T(\boldsymbol{V}_X + \boldsymbol{V}_Y)(\overline{X} - \overline{Y})} \\ &= (\overline{X} - \overline{Y})^T(\overline{X} - \overline{Y})\left((\overline{X} - \overline{Y})^T\boldsymbol{W}(\overline{X} - \overline{Y})\right)^{-1}(\overline{X} - \overline{Y})^T(\overline{X} - \overline{Y}) \\ &= (\overline{X} - \overline{Y})^T\boldsymbol{W}^{-1}(\overline{X} - \overline{Y}) = d^2, \end{aligned} \tag{A.45}$$

where d is the inter-cluster distance, as defined in Equation A.36.

A.6.2.1 Multi-class LDA

When there are more than 2 classes, an extension of the Fischer discriminant analysis can be used to find a subspace containing all the class variability. If M_k is the mean of the k^{th} class, M is the mean of all the classes means and c is the number of classes, the variance between classes, $\boldsymbol{V}_b$ is:

$$\boldsymbol{V}_b = \frac{1}{c}\sum_{k=1}^{c}(M_k - M)(M_k - M)^T, \tag{A.46}$$

where c is the number of classes. If $X_1^k \dots X_n^k$ are the vectors of the class C_k, then the intraclass variance is:

$$\boldsymbol{V}_k = \frac{1}{n-1}\sum_{j=1}^{n}(X_j^k - M_k)(X_j^k - M_k)^T \tag{A.47}$$

If we suppose that all the classes have approximately the same covariance within class, we can approximate this variance within class, $\boldsymbol{V}_1$, by the average value of all the $\boldsymbol{V}_k$:

$$\boldsymbol{V}_i = \frac{\sum_{k=1}^{c} V_k}{c} \tag{A.48}$$

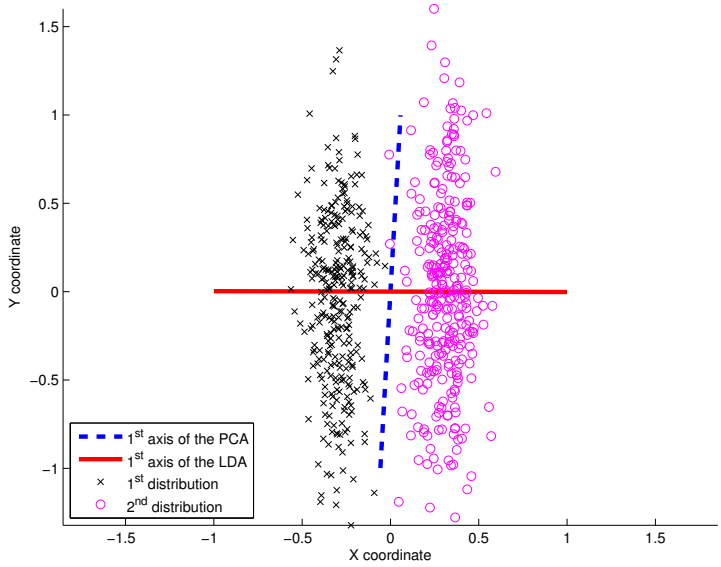


Figure A.15: PCA and LDA applied to two Gaussian clusters of 300 points each. The first vector of the PCA will focus on the axis of maximal variance. However, LDA will focus on the axis of best separation. Thus, if the differences between the datasets does not lie in the dimensions of maximal variance, PCA is not the right tool to use

Thus the ratio r of the variance between the classes to the variance within classes in direction L is:

$$r = \frac{L^T \boldsymbol{V}_b L}{L^T \boldsymbol{V}_i L} = \frac{L^T \left(\boldsymbol{V}_b {\boldsymbol{V}_i}^{-1}\right) \boldsymbol{V}_i L}{L^T \boldsymbol{V}_i L} \tag{A.49}$$

Thus, when L is an eigenvector of $\boldsymbol{V}_b {\boldsymbol{V}_i}^{-1}$, the ratio will be equal to the corresponding eigenvalue. As $\boldsymbol{S_b}$ has a maximum rank of $c-1$, the non-zero eigenvectors will describe a subspace separating the classes (Separation hyperplane).

A.6.2.2 Practical use of the LDA

The LDA is mainly used for dimensionality reduction, like the PCA. It is also really sensitive to noise, especially when the dataset is small and they are a lot of features. However, it is commonly used to reduce the dimensionality of a dataset, before applying more complex models, such as Hidden Markov Models (HMM) or Gaussian Mixture Models (GMM). Sometimes, the LDA is also combined with a PCA.

Figure A.15 shows the results of the application of the PCA and the LDA to two Gaussian clusters of points. At the difference of the PCA which focuses on the variance of the clusters, the LDA focuses on the separation of theses specific clusters.

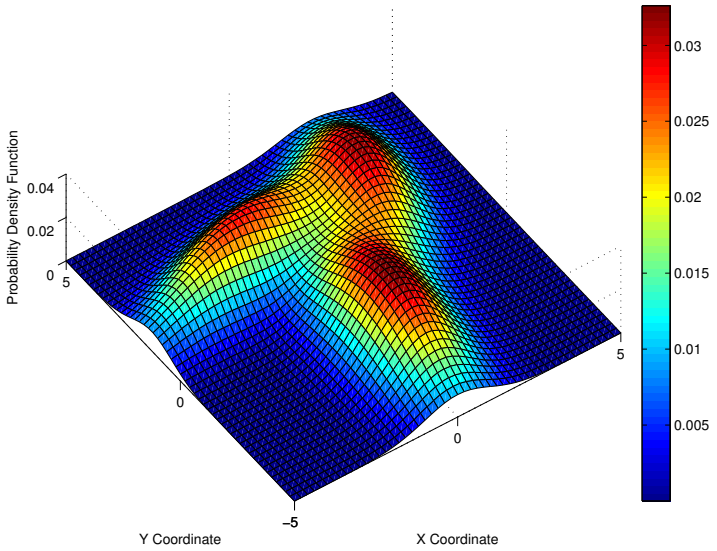


Figure A.16: Probability density function of an example Gaussian Mixture Model made of three bidimensional Gaussians

A.7 Gaussian mixture model

A mixture model is a mix of multiple models. Given n probabilistic models defined by their probability density function (PDF): $f_{Y_i}(x)$, for $i \in [1 \dots n]$, the global PDF becomes:

$$f_X(x) = \sum_{i=1}^{n} a_i f_{Y_i}(x), \tag{A.50}$$

for mixture proportions a_i $(0 < a_i < 1)$, where $\sum_{i=1}^{n} a_i = 1$.

In the specific case where Gaussian models are added, it is called a Gaussian Mixture Model (GMM). The PDF becomes, in an unidimensional case:

$$f_X(x) = \sum_{i=1}^{n} \frac{1}{\sigma_i \sqrt{2\pi}} exp\left(-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right). \tag{A.51}$$

Thus, the model has $2n$ parameters. If the modeled data is multidimensional, with d dimensions, the number of parameters is quickly growing: $n \cdot d$ parameters are needed for the means and $n \cdot \frac{d(d+1)}{2}$ are needed to represent the covariance matrices. GMM are widely used in sound and image processing to model multiple clusters that are clearly separated at different spots and thus cannot be modeled with a unique Gaussian.

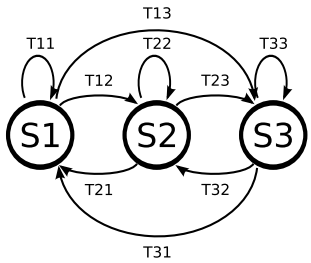


Figure A.17: Exemple of a discrete Markov model made of three states. S_1, S_2 and S_3 are the three possible states and T_{ij} is the transition probability from state S_i to state S_j

Figure A.16 shows the probability density function of a Gaussian mixture model made of three two-dimensional Gaussian distributions. The maximums of the three underlying distributions are clearly visible on this plot. It shows also how complex cluster shapes can be represented with GMM.

A.8 Hidden markov model

A Markov Chain, named after Andrey Markov (1856-1922), is a stochastic process with the Markovian property. This property is often called memoryless. For discrete process, it means that the current state depends only on the last prior state and that is it completely independent of the previous states. Expressed more formally, for a discrete system:

$$P(x_n = s \mid x_i,\ \forall i \leq n-1) = P(x_n = s \mid x_{n-1}), \tag{A.52}$$

where x_n is the output value of the process at the n^{th} step and s in one of the possible output value of the process. Dice rolling games, such as the Monopoly, are examples of Markovian process. The result of a new dice roll does not depend on the history of the dice rolls. On the contrary, in card games, the probability to get a specific card is not independent of history.

Figure A.17 shows an example of a discrete Markov model. The model is made of 3 states and the transition probabilities between the states are shown: $T_{ij} = P(X_n = S_j | X_{n-1} = S_i)$. The chain $[S_1\ S_3\ S_2\ S_1\ S_1\ S_2]$ is an example of a Markov chain generated with this model. If $X_1 = S1$, the probability of this chain occurrence is $P(X = [S_1\ S_3\ S_2\ S_1\ S_1\ S_2] | X_1 = S_1] = T_{13} \cdot T_{32} \cdot T_{21} \cdot T_{11} \cdot T_{12}$.

The Hidden Markov Model (HMM), was introduced by Leonard E. Baum in the second half of the 1960s and was first used for speech recognition. Now, this model is widely used in applications such as speech, handwriting, gesture and gait recognition. An HMM is composed of an underlying Markov Model, whose states are not observable. These states have a direct influence (output probabilities) on the occurrence of the next visible observation. Figure A.18 shows an example of a discrete HMM made of 3 hidden states (upper part) and two possible observations (lower part). If the following output is

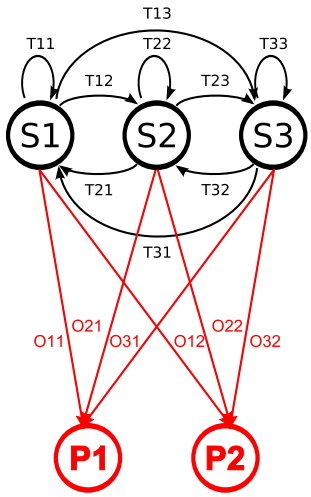


Figure A.18: Example of an Hidden Markov Model. S_i represent the hidden states, P_i are the possible observations, T_{ij} are the state transition probabilities and O_{ij} are the output probabilities

observed, $[P_1\ P_1\ P_2\ P_1\ P_2\ P_2]$, it is not possible to know which list of states led to this output. However, from the transition and output probabilities, we can compute the probability that a given chain (e.g. $[S_1\ S_2\ S_1\ S_3\ S_3\ S_2]$) led to the output and thus the most probable chain can be found.

Three main algorithms used with the Hidden Markov Model are:

- Given the parameters of the model, the forward-backward algorithm is used to compute the probability of a specific output sequence and the probabilities of the hidden states given this output sequence.
- Given the parameters of the model and the output sequence, The Viterbi algorithm [114] is used to find the most likely sequence of hidden states.
- Given the output sequence, the Baum-Welch algorithm [115] evaluates the parameters of the HMM.

A good tutorial on the HMM can be found in Lawrence R. Rabiner article [116].

A.8.1 GMM/HMM hybrid model

Hybrid models based on HMM are often used in recognition systems. A common example is the hybrid GMM/HMM model. Figure A.19 shows an example of this model. It is made of a Markov Model with each state being linked to a Gaussian Mixture Model. It is not an exact aggregation of a HMM and a GMM, as the output states of the HMM disappear. But as the underlying Markov Model remains hidden, the name is kept.

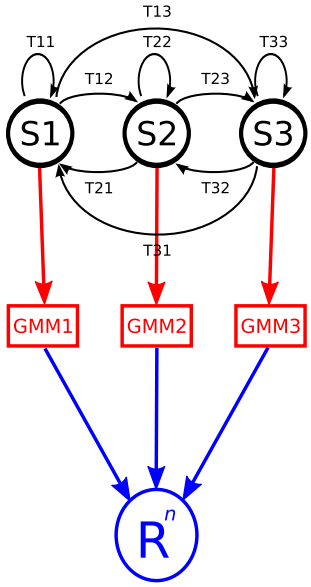


Figure A.19: Example of a GMM/HMM hybrid model. On the upper part of the graph lies the hidden Markov Model. To each state of the model corresponds a GMM. These GMM are used to represent multidimensional clusters in R^n

This model is clearly more interesting than the HMM, as it can be used with continuous data. It is thus easy to use with real data and is often used for speech recognition [117, 99]. The HMM/GMM hybrid model has quite a lot of parameters:

$$n_{parameters} = n_{HMM} + n_{GMM} = s^2 + s\left(m \cdot d + m \cdot \frac{d(d+1)}{2}\right), \tag{A.53}$$

where s is the number of hidden states of the HMM, d the dimensionality of the output space and m the number of Gaussian distributions used in each GMM.

As a result, these models use a lot of computation power and memory, and need a lot of time to evolve.

A.9 Random walk

"A random walk is a mathematical formalization of a trajectory that consists of taking successive random steps" [2]. This model of random process has been used to represent the movement of a molecule in a liquid, the travel path of a foraging animal or insect, or the price of a fluctuating stock. A random walk is assumed to be Markov process: the next position depends only on the current position.

[2] www.wikipedia.org

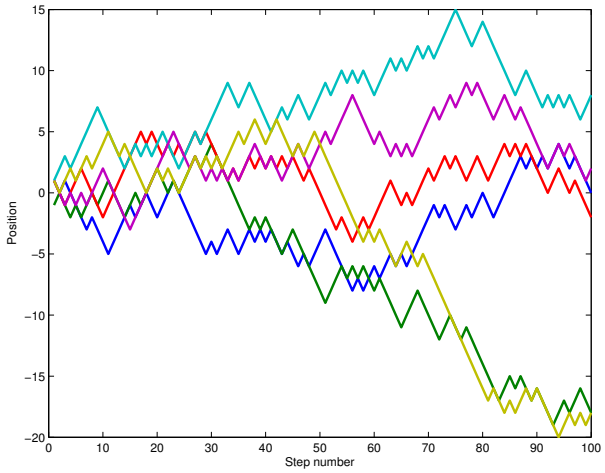


Figure A.20: Examples of a unidimensional random walk. At each step, the position is modified by ± 1 with a probability of 0.5 for each choice. Six examples of the same random walk model are plotted. They result in quite different trajectories

A.9.1 Unidimensional random walk

Figure A.20 shows an example of a unidimensional random walk. If at the n^{th} state $x_n = a$, for the next state $(n+1)$, $P(x_{n+1} = a+1 \mid x_n = a) = 0.5$ and $P(x_{n+1} = a-1 \mid x_n = a) = 0.5$. It means that the next state will be modified by ± 1. After 100 steps, the six examples end at really different positions, as the process is completely random.

A.9.2 Bidimensional random walk

Similarly, a bidimensional example can be built with the following rules:

- $P(x_{n+1} = a+1, y_{n+1} = b \mid x_n = a, y_n = b) = 0.25,$
- $P(x_{n+1} = a-1, y_{n+1} = b \mid x_n = a, y_n = b) = 0.25,$
- $P(x_{n+1} = a, y_{n+1} = b+1 \mid x_n = a, y_n = b) = 0.25,$
- $P(x_{n+1} = a, y_{n+1} = b-1 \mid x_n = a, y_n = b) = 0.25.$

Informally, it means that at each step, only one of the two coordinates is modified by ± 1, with the same random probability. Figure A.21 shows an example of this random walk. The resulting trajectory is quite similar to Brownian motion.

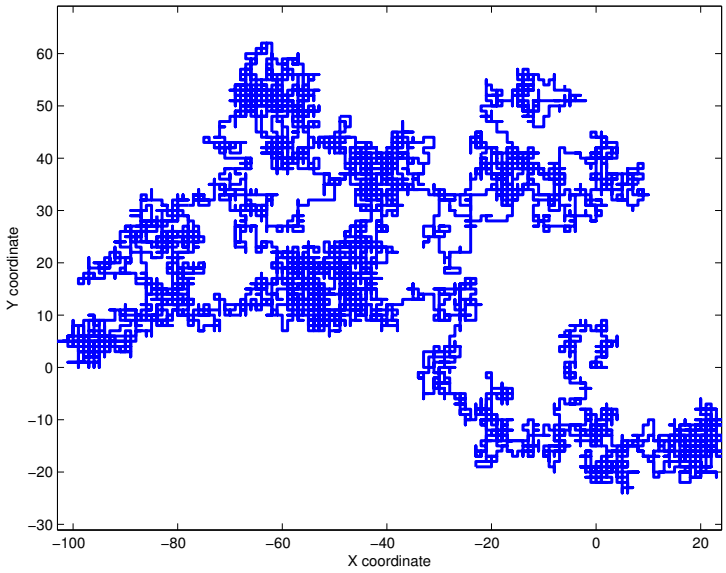


Figure A.21: Example of a bidimensional random walk. At each step, one of the two coordinates is modified by ± 1, with the same probability

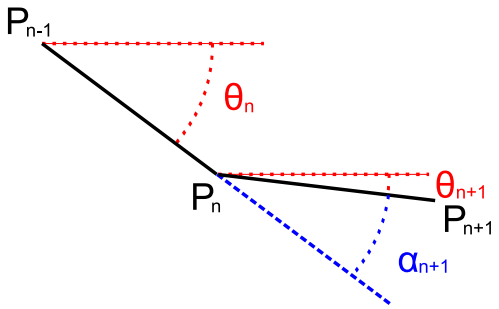


Figure A.22: Principle of the Correlated Random Walk: the direction Θ_{n+1} of the step $[P_n, P_{n-1}]$ is related to the direction Θ_n of the previous step $[P_{n-1}, P_n]$. A distribution of the α_i angles is defined

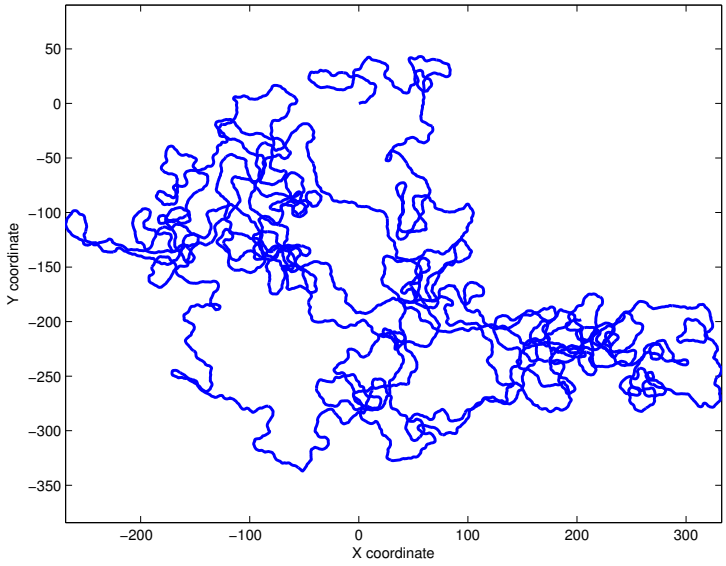


Figure A.23: Example of a Correlated Random Walk made of 10'000 steps. The angle distribution was a Gaussian distribution with a standard deviation of $\sigma = \frac{1}{3}$ radians

A.9.3 Correlated random walk

To represent the movement of animals, Roger Kitching [118] presented an evolution of the random walk model, which will be called the *Correlated Random Walk*. In this model, instead of having a step direction that is completely independent of the previous step direction, a correlation is considered between two successive step directions. Figure A.22 shows the different elements of this model. The direction Θ_{n+1} of the step $[P_n, P_{n+1}]$ is not completely random, it is related to the orientation Θ_n of the previous step. Thus a distribution of the angle α_n between the two directions is defined. This distribution can be represent by flat, Gaussian or more complex distributions. As P_{n+1} depends on the last two positions (P_{n-1} and P_n), this process is no more Markov.

Figure A.23 shows an example of a Correlated Random Walk. The step angles follow a Gaussian distribution with standard deviation of $\sigma = \frac{1}{3}$ radians. If we compare it to the previous bidimensional example, the resulting trajectory is less intermingled. As a result, for the same number of steps (10'000) and the same step length (1), the last trajectory covers an area approximately 16 times greater. Figure A.24 shows the histogram of the angles between the steps. The underlying Gaussian distribution is clearly visible.

Another type of Correlated Random Walk defines a distribution for both the angles between the steps and the step lengths. Most of the time, these two distributions are considered independent (i.e. knowing the length of a step does not give a hint about the angle with the previous step). Figure A.25 shows an

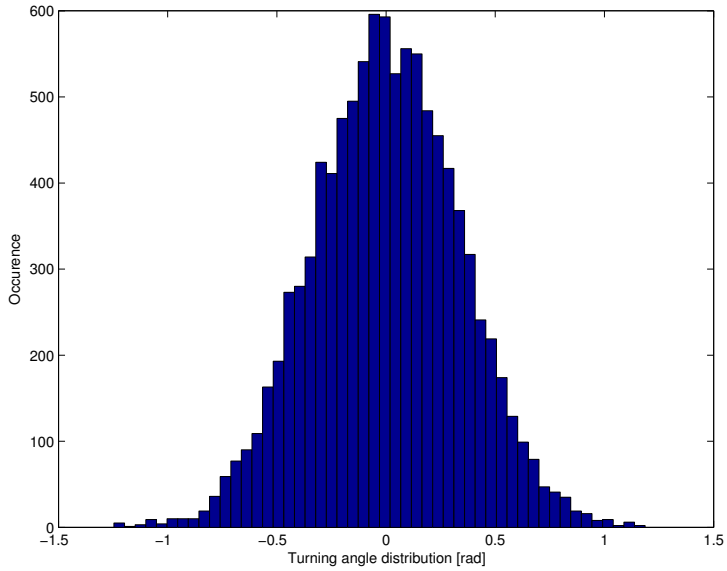


Figure A.24: Histogram of the angles between the steps for the Correlated Random Walk shown in Figure A.24. The underlying Gaussian distribution is clearly visible

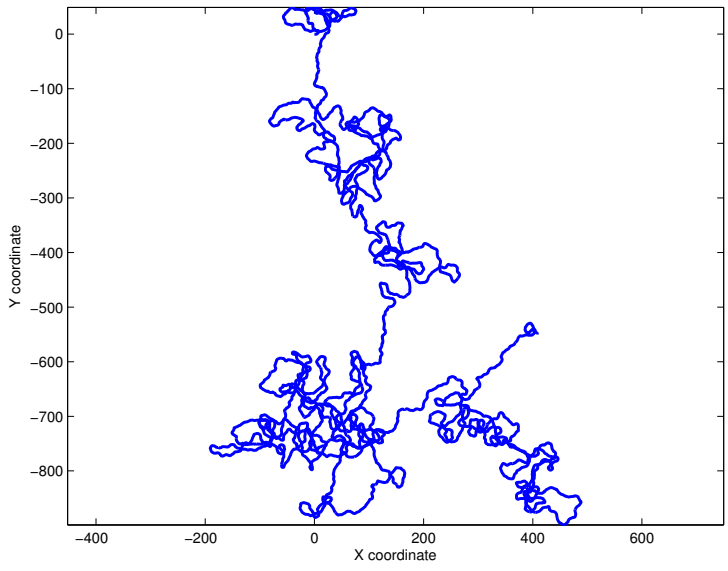
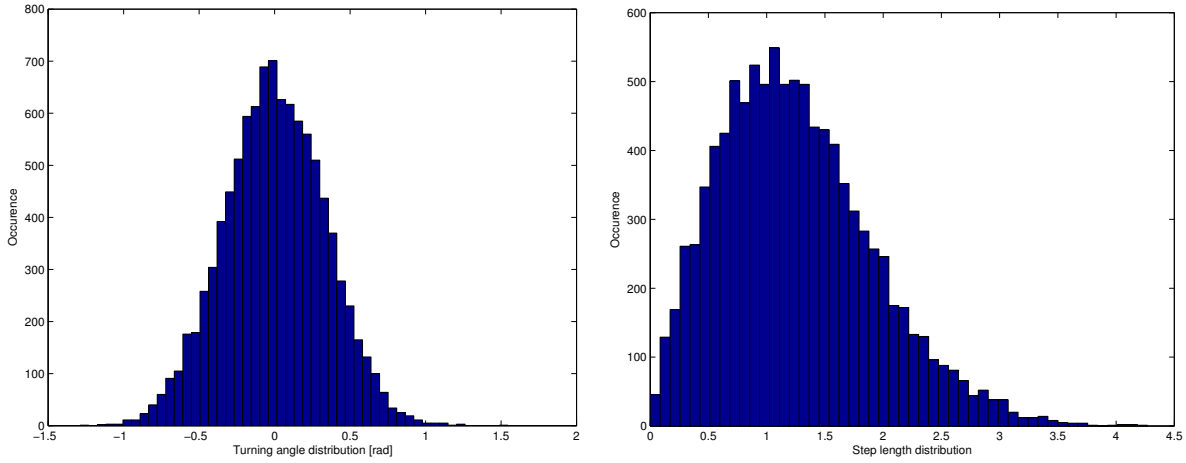


Figure A.25: Correlated Random Walk, where the angles between the steps follow a Gaussian distribution of standard deviation $\sigma = \frac{1}{3}$ and the step lengths follow a Rayleigh distribution with $\sigma = 1$



(a) (b)

Figure A.26: On the left, histogram of the step angles of the Correlated Random Walk shown in Figure A.25. On the right, histogram of the step length of the same Correlated Random Walk

example of this kind of Correlated Random Walk. The step lengths follow a Rayleigh distribution with $\sigma = 1$ and the step angles follow a Gaussian distribution with $\sigma = \frac{1}{3}$. In this case, the coverage is even bigger than in the previous example. However, the trajectories are quite similar. Figure A.26(a) shows the histogram of the step angles and Figure A.26(b) shows the histogram of the step lengths. The two underlying distributions (Gaussian and Rayleigh) are easy to recognize.

These are just simple examples, but Correlated Random Walk is widely used by mathematicians and biologist, such as Byers [71] or Jost [72].

Appendix B

Glossary

Acronym	Full name	Description
CDF	Cumulative Distribution Function	Integral of the PDF. It is bounded below by 0 and above by 1.
CRW	Correlated Random Walk	Stepwise model of motion. It is introduced in Appendix A.9.3.
F-F test	Fasano and Franceschini test	Algorithm proposed by Fasano and Franceschini to extend the K-S test to bivariate data. It is introduced in Appendix A.3.3.
GMM	Gaussian Mixture Model	Model representing multivariate PDF with multiple Gaussian distributions. It is introduced in Appendix A.7.
HMM	Hidden Markov Model	Statistical model introduced in Appendix A.8.
K-S test	Kolmogorov-Smirnov test	Statistical test comparing the CDFs. It is introduced in Appendix A.2.6.
LDA	Linear Discriminant Analysis	Dimensionality reduction technique based on the axes of maximal separation between multivariate datasets. It is introduced in Appendix A.6.2.
PCA	Principal Components Analysis	Dimensionality reduction technique based on maximal variance. It is introduced in Appendix A.6.1.
PDF	Probability Density Function	Function introduced in statistics to represent the probability density.
PDM	Point Distribution Model	Active shape model based on the PCA. It is introduced in Chapter 3.

Bibliography

[1] A. Wägli and J. Skaloud, "Turning point - trajectory analysis for skiers," *Inside GNSS*, pp. 24 – 34, 2007.

[2] A. Wägli, *Trajectory Determination and Analysis in Sports by Satellite and Inertial Navigation*. Thesis n° 4288, Ecole Polytechnique Fédérale de Lausanne, 2009.

[3] A. Sanfeliu and J. J. Villanueva, "An approach of visual motion analysis," *Pattern Recognition Letters*, vol. 26, pp. 335–368, 2005.

[4] T. Cootes, C. Taylor, and D. Cooper, "Active shape-models - their training and applications," *Vision and Image Understanding*, pp. 38–59, 1995.

[5] Y. Zhong, A. K. Jain, and M.-P. Dubuisson-Jolly, "Object tracking using deformable templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 544–549, May 2000.

[6] J. T. Betts and S. O. Erb, "Optimal low thrust trajectories to the moon," *Journal of Applied Dynamical Systems*, vol. 2, no. 2, pp. 144–170, 2003.

[7] G. Vulpetti, "General 3D H-reversal trajectories for high-speed sailcraft," *Acta Astronautica*, vol. 44, no. 1, pp. 67–73, 1999.

[8] A. Sturman and P. Zawar-Reza, "Application of back-trajectory techniques to the delimitation of urban clean air zones," *Atmospheric Environment*, vol. 36, pp. 3339–3350, 2002.

[9] S. S. Raza, R. Avila, and J. Cervantes, "A 3-d Lagrangian stochastic model for the meso-scale atmospheric dispersion applications," *Nuclear Engineering and Design*, vol. 208, pp. 15–28, 2001.

[10] R. B. Oza, N. S. Panchal, K. S. V. Nambi, and T. M. Krishnamoorthy, "Coupling of mesoscale meteorological model with particale trajectory model to study the atmospheric dispersion under sea breeze conditions," *Environmental Modelling and Software*, vol. 16, pp. 63–71, 2001.

[11] K. B. Mcgrattan, "Smoke plume trajectory modelling," *Spill Science and Technology Bulletin*, vol. 8, no. 4, pp. 367–371, 2003.

[12] W. N. Meier and J. A. Maslanik, "Improved sea ice parcel trajectories in the artic via data assimilation," *Marine Pollution Bulletin*, vol. 42, no. 6, pp. 506–512, 2001.

[13] M. Marghany, "Radarsat for oil spill trajectory model," *Environmental Modelling and Software*, vol. 19, pp. 473–483, 2004.

[14] S. Munasinghe, M. Nakamura, S. Goto, and N. Kyura, "Precise control of industrial robot arms considering trajectory allowance under torque and speed constraints," in *International Conference on Robotics and Automation*, vol. 4, (Seoul), pp. 3949–3954, May 2001.

[15] C. Fanti, L. Zelnik-Manor, and P. Perona, "Hybrid models for human motion recognition," in *International Conference on Computer Vision and Pattern Recognition*, vol. 1, (San Diego), pp. 1166–1173, June 2005.

[16] M. C. Shin, L. V. Tsap, and D. B. Goldgof, "Gesture recognition using Bezier curves for visualization navigation from registered 3-D data," *Pattern Recognition*, vol. 37, pp. 1011–1024, 2004.

[17] C.-L. Huang and M.-S. Wu, "Gesture image sequence interpretation using the multi-PDM method and hidden Markov model," in *International Conference on Acoustics, Speech and Signal Processing*, vol. 6, (Phoenix), pp. 3541–3544, March 1999.

[18] D. D. Vecchio, R. M. Murray, and P. Perona, "Decomposition of human motion into dynamics-based primitives," *Automatica*, vol. 39, pp. 2085–2098, December 2003.

[19] J. Han, B. Bhanu, and A. K. Roy-Chowdhury, "A study on view-insensitive gait recognition," in *International Conference on Image Processing*, vol. 3, (Genoa), pp. 297–300, September 2005.

[20] S. Wu and Y. F. Li, "On signature invariants for effective motion trajectory recognition," *The International Journal of Robotics Research*, vol. 27, no. 8, pp. 895–917, 2008.

[21] R. Urtasun, D. J. Fleet, and P. Fua, "Temporal motion models for monocular and multiview 3–D human body tracking," *Computer Vision and Image Understanding*, vol. 104, pp. 157–177, December 2006.

[22] J. P. Lauffenburger, M. Basset, F. Coffin, and G. L. Gissinger, "Driver-aid system using path-planning for lateral vehicule control," *Control Engineering Practice*, vol. 11, pp. 217–231, 2003.

[23] W. Hu, X. Xiao, D. Xie, and T. Tan, "Traffic accident prediction using vehicle tracking and trajectory analysis," in *International Conference on Intelligent Transportation Systems*, (Shanghai), pp. 220–225, October 2003.

[24] W. Hu, D. Xie, and T. Tan, "A hierarchical self-organizing approach for learning the patterns of motion trajectories," *IEEE Transactions on Neural Networks*, vol. 15, pp. 135–144, January 2004.

[25] N. Johnson and D. Hogg, "Learning the distribution of object trajectories for event recognition," *Image and Vision Computing*, vol. 14, pp. 609–615, 1996.

[26] N. Oliver and A. Pentland, "Graphical models for driver behavior recognition in a smart car," in *Intelligent Vehicles Symposium*, (Dearborn), October 2000.

[27] J. Owens and A.Hunter, "Application of the self-organising map to trajectory classification," in *International Workshop on Visual Surveillance*, (Dublin), pp. 77–83, July 2000.

[28] J. Owens, A. Hunter, and E. Fletcher, "Novelty detection in video surveillance using hierarchical neural networks," *Lecture Notes in Computer Science*, vol. 2415, pp. 1249–1254, 2002.

[29] A. Hunter, J. Owens, and M. Carpenter, "A neural system for automated CCTV surveillance," in *Symposium on Intelligence Distributed Surveillance Systems*, (London), pp. 1411–1415, February 2003.

[30] W. E. L. Grimson, C. Stauffer, R. Romano, and L. Lee, "Using adaptive tracking to classify and monitor activities in a site," in *International Conference on Computer Vision and Pattern Recognition*, (Santa Barbara), pp. 22–29, June 1998.

[31] R. Wallace, *Finding Natural Clusters through Entropy Minimization*. PhD thesis, CMU, CMU-CS-89-183, 1989.

[32] K. K. Lee and Y. Xu, "Boundary modeling in human walking trajectory analysis for surveillance," in *International Conference on Robotics and Automation*, (New Orleans), pp. 5201–5206, April 2004.

[33] N. Oliver, B. Rosario, and A. P. Pentland, "A Bayesian computer vision system for modeling human interactions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, August 2000.

[34] R. T. Collins, A. J. Lipton, T. Kanade, H. Fujiyoshi, D. Duggins, Y. Tsin, D. Tolliver, N. Enomoto, O. Hasegawa, P. Burt, and L. Wixson, "A system for video surveillance and monitoring," tech. rep., The Robotics Institute, Carnegie Mellon University, 2000.

[35] N. P. Cuntoor, B. Yegnanarayana, and R. Chellappa, "Interpretation of states sequences in HMM for activity representation," in *International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, (Philadelphia), pp. 709–712, March 2005.

[36] N. Vaswani, A. R. Chowdhury, and R. Chellappa, "Activity recognition using the dynamics of the configuration of interacting objects," in *International Conference on Computer Vision and Pattern Recognition*, vol. 2, (Los Alamitos), p. 633, June 2003.

[37] R. J. Evans, E. Turkbeyler, and E. P. Sparks, "Analysis of visual feature motion for event and threat detection," in *2nd EMRS DTC Technical Conference*, (Edinburgh), June 2005.

[38] P. Remagnino, T. Tan, and K. Baker, "Multi-agent visual surveillance of dynamic scenes," *Image and Vision Computing*, vol. 16, pp. 529–532, 1998.

[39] P. Remagnino, T. Tan, and K. Baker, "Agent orientated annotation in model based visual surveillance," in *International Conference on Computer Vision*, (Bombay), pp. 857–862, January 1998.

[40] F. V. Jensen, *Bayesian Networks and Decision Graphs*. Springer, 2001.

[41] M. Bertini, A. D. Bimbo, and W. Nunziati, "Highlights modeling and detection in sports videos," *Pattern Analysis and Application*, vol. 7, no. 4, pp. 411–421, 2005.

[42] N. Johnson and D. Hogg, "Representation and synthesis of behaviour using Gaussian mixtures," *Image and Vision Computing*, vol. 20, pp. 889–894, 2002.

[43] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *International Conference on Computer Vision and Pattern Recognition*, vol. 2, (Ft. Collins), pp. 246–252, June 1999.

[44] C. Stauffer and W. E. L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 747–757, 2000.

[45] F. Porikli, "Learning object trajectory patterns by spectral clustering," in *International Conference on Multimedia and Expo*, vol. 2, (Taipei), pp. 1171–1174, June 2004.

[46] F. Porikli, "Trajectory pattern detection by HMM parameter space features and eigenvector clustering," in *International Conference on Multimedia and Expo*, (Taipei), June 2004.

[47] F. Porikli, "Trajectory distance metric using hidden Markov model based representation," in *European Conference on Computer Vision, Workshop on PETS*, (Prague), May 2004.

[48] J. Alon, S. Sclaroff, G. Kollios, and V. Pavlovic, "Discovering clusters in motion time-series data," in *International Conference on Computer Vision and Pattern Recognition*, vol. 1, (Los Alamitos), pp. 375–378, June 2003.

[49] D. Buzan, S. Sclaroff, and G. Kollios, "Extraction and clustering of motion trajectories in video," in *International Conference on Pattern Recognition*, vol. 2, (Cambridge), pp. 521–524, August 2004.

[50] C. Piciarelli and G. L. Foresti, "Toward event recognition using dynamic trajectory analysis and prediction," in *International Symposium on Imaging for Crime Detection and Prevention*, (London), pp. 131–135, June 2005.

[51] C. Sas, G. O'Hare, and R. Reilly, "Virtual environment trajectory analysis: a basis for navigational assistance and scene adaptivity," *Future Generation Computer Systems*, vol. 21 (7), pp. 1157–1166, July 2005.

[52] C. Sas, G. O'Hare, and R. Reilly, "A performance analysis of movement patterns," *Lecture Notes in Computer Science*, vol. 3038, pp. 954–961, 2004.

[53] C. Sas, "User model of navigation," *Lecture Notes in Computer Science*, vol. 3101, pp. 379–388, 2004.

[54] C. Sas, G. O'Hare, and R. Reilly, "Online trajectory classification," *Lecture Notes in Computer Science*, vol. 2659, pp. 1035–1044, 2003.

[55] J. Ng and S. Gong, "Learning intrinsic video content using Levenshtein distance in graph partitioning," *Lecture Notes in Computer Science*, vol. 2353, 2002.

[56] K. Buchin, M. Buchin, J. Gudmundsson, M. Löffler, and J. Luo, "Detecting commuting patterns by clustering subtrajectories," technical report UU-CS-2008-029, Department of Information and Computing Sciences, Utrecht University, Utrecht, The Netherlands, September 2008.

[57] G. Trajcevski, H. Ding, and P. Scheuermann, "Dynamics-aware similarity of moving objects trajectories," in *International Symposium on Advances in Geographic Information Systems*, (Seattle), pp. 75–82, November 2007.

[58] F. I. Bashir, *MotionSearch: Object Motion Trajectory-Based Video Search and Classification System*. PhD thesis, University of Illinois, 2006.

[59] A. Efrat and Q. Fan, "Curve matching, time warping, and light fields: New algorithms for computing similarity between curves," *Journal of Mathematical Imaging and Vision*, vol. 27, pp. 203–216, April 2007.

[60] J. Van Hoof, "A study of quantitative and qualitative methods for trajectories," Master's thesis, Hasselt University, 2007.

[61] J. H. Fernyhough, A. G. Cohn, and D. C. Hogg, "Generation of semantic regions from image sequences," *Lecture Notes in Computer Science*, vol. 1065, pp. 475–484, 1996.

[62] D. Makris and T. Ellis, "Automatic learning of an activity-based semantic scene model," in *International Conference on Advanced Video and Signal Based Surveillance*, (Miami), July 2003.

[63] S. J. McKenna and H. Nait-Charif, "Learning spatial context from tracking using penalised likelihoods," in *International Conference on Pattern Recognition*, (Cambridge), August 2004.

[64] X. Wang, K. Tieu, and E. Grimson, "Learning semantic scene models by trajectory analysis," *Lecture Notes in Computer Science*, vol. 3953, pp. 110–123, 2006.

[65] M. Egerstedt, T. Balch, F. Dellaert, F. Delmotte, and Z. Khan, "What are the ants doing ? vision-based tracking and reconstruction of controlprograms," in *International Conference on Robotics and Automation*, (Barcelona), pp. 4193–4198, April 2005.

[66] D. M. Gordon, "The expandable network of ant exploration," *Animal Behaviour*, vol. 50, no. 4, pp. 995–1007, 1995.

[67] M. Kohler and R. Wehner, "Idiosyncratic route-based memories in desert ants, Melophorus bagoti: How do they interact with path-integration vectors?," *Neurobiology of Learning and Memory*, vol. 83, pp. 1–12, 2005.

[68] T. Balch, Z. Khan, and M. Veloso, "Automatically tracking and analyzing the behavior of social insect colonies," in *International Conference on Autonomous Agents*, (Montreal), May 2001.

[69] Z. Khan, T. Balch, and F. Dellaert, "Efficient particle filter-based tracking of multiple interacting targets using an MRF-based motion model," in *International Conference on Intelligent Robots and Systems*, (Las Vegas), October 2003.

[70] Z. Khan, T. R. Balch, and F. Dellaert, "An MCMC-based particle filter for tracking multiple interacting targets," in *European Conference on Computer Vision*, (Prague), pp. 279–290, May 2004.

[71] J. A. Byers, "Correlated random walk equations of animal dispersal resolved by simulation," *Ecology*, vol. 82, pp. 1680–1690, 2001.

[72] C. Jost, S. Garnier, R. Jeanson, M. Asadpourand, J. Gautrais, and G. Theraulaz, "The embodiment of cockroach behavior in a micro-robot," in *International Symposium on Robotics*, (Paris), March 2004.

[73] S. Garnier, C. Jost, J. Gautrais, M. Asadpour, G. Caprari, R. Jeanson, A. Grimal, and G. Theraulaz, "The embodiment of cockroach aggregation behavior in a group of micro-robots," *Artificial Life*, vol. 14, pp. 387–408, 2008.

[74] R. Jeanson, C. Rivault, J. L. Deneubourg, S. Blanco, R. Fournier, C.Jost, and G. Theraulaz, "Self-organized aggregation in cockroaches," *Animal Behaviour*, vol. 69, pp. 169–180, 2005.

[75] D. Biro, R. Freeman, J. Meade, S. Roberts, and T. Guilford, "Pigeons combine compass and landmark guidance in familiar route navigation," *Proceedings of the National Academy of Sciences of the USA*, vol. 104, no. 18, pp. 7471–7476, 2007.

[76] G. Antonini, S. Venegas, J. Thiran, and M. Bierlaire, "A discrete choice pedestrian behavior model for pedestrian detection invisual tracking systems," in *International Conference on Advanced Concepts for Intelligent Vision Systems*, (Brussels), pp. 159 – 180, September 2004.

[77] S. Kajikawa, T. Okino, K. Ohba, and H. Inooka, "Motion planning for hand-over between human and robot," in *International Conference on Intelligent Robots and Systems*, vol. 1, (Pittsburgh), pp. 193–199, August 1995.

[78] S. Kajikawa, N. Saito, and H. Okano, "Receiver robot's motion for handing-over with a human," in *International Workshop on Robot and Human Interactive Communication*, (Berlin), pp. 494–499, September 2002.

[79] U. Nehmzov, "Quantitative analysis of robot-environment interaction towards "scientificmobilerobotics"," *Robotics and Autonomous Systems*, vol. 44, pp. 55–68, 2003.

[80] T. Smithers, "On quantitative performance measures of robot behaviour," *Robotics and Autonomous Systems*, vol. 15, pp. 107–133, 1995.

[81] G. Schöner, M. Dose, and C. Engels, "Dynamics of behavior : theory and applications for autonomous robot architecture," *Robotics and Autonomous Systems*, vol. 16, pp. 213–245, 1995.

[82] U. Nehmzow, *Scientific Methods in Mobile Robotics: Quantitative Analysis of Agent Behaviour*. Springer, 1 ed., 2006.

[83] O. Akanyeti, T. Kyriacou, U. Nehmzov, R. Iglesias, and S. A. Billings, "Visual task identification and characterization using polynomial models," *Robotics and Autonomous Systems*, vol. 55, pp. 711–719, 2007.

[84] R. Iglesias, T. Kyriacou, U. Nehmzov, and S. Billings, "Task identification and characterisation in mobile robotics through non-linear modelling," *Robotics and Autonomous Systems*, vol. 55, pp. 267–275, 2007.

[85] D. Goldberg and M. J. Matarić, "Coordinating mobile robot group behavior using a model of interaction dynamics," in *International Conference on Autonomous Agents*, (Seattle), May 1999.

[86] D. Goldberg and M. J. Matarić, "Detecting regime changes with a mobile robot using multiple models," in *International Conference on Intelligent Robots and Systems*, (Wailea), pp. 619–624, October 2001.

[87] H. Yan and M. J. Matarić, "General spatial features for analysis of multi-robot and human activities from raw position data," in *International Conference on Intelligent Robots and Systems*, (Lausanne), pp. 2770–2775, September 2002.

[88] F. Luisier, "Analyse de trajectoires pour le sport et la vidéosurveillance," semester project, Laboratoire de Production Microtechnique, EPFL, 2004.

[89] Y. L. de Meneses, P. Roduit, F. Luisier, and J. Jacot, "Trajectory analysis for sport and video surveillance," *Electronic Letters on Computer Vision and Image Analysis*, vol. 5, no. 3, pp. 148–156, 2005. Special issue on Articulated Motion & Deformable Objects.

[90] A. Hitz, "Analysis of mobile-robot trajectories in open space," technical report SWIS-SP23, Swarm-Intelligent Systems Group, EPFL, February 2007.

[91] P. Roduit, A. Martinoli, and J. Jacot, "Behavioral analysis of mobile robot trajectories using a point distributionmodel," *Lecture Notes in Computer Science*, vol. 4095, pp. 819 – 830, 2006.

[92] O. Michel, "Webots: Professional mobile robot simulation," *Journal of Advanced Robotics Systems*, vol. 1, no. 1, pp. 29–42, 2004.

[93] P. Roduit, A. Martinoli, and J. Jacot, "A quantitative method for comparing trajectories of mobile robots using point distribution models," in *International Conference on Intelligent Robots and Systems*, (San Diego), pp. 2442–2448, November 2007.

[94] "www.e-puck.org."

[95] N. Correll, G. Sempo, Y. L. de Meneses, J. Halloy, J.-L. Deneubourg, and A. Martinoli, "A tracking tool for multi-unit robotic and biological systems.," in *International Conference on Intelligent Robots and Systems*, (Beijing), pp. 2185 – 2191, October 2006.

[96] T. Lochmatter, P. Roduit, C. Cianci, N. Correl, J. Jacot, and A. Martinoli, "SwisTrack - a flexible open source tracking software for multi-agent systems," in *International Conference on Intelligent Robots and Systems*, (Nice), pp. 4004–4010, September 2008.

[97] A. T. Hayes, A. Martinoli, and R. M. Goodman, "Distributed odor source localization," *IEEE Sensors Journal, Special Issue in Artificial Olfaction*, vol. 2, no. 3, pp. 260–271, 2002.

[98] S. Anand, "Comparison of machine learning algorithms for trajectory classification," technical report SWIS-SP23, Swarm-Intelligent Systems Group, EPFL, June 2006.

[99] P. Pujol, S. Pol, C. Nadeu, A. Hagen, and H. Bourlard, "Comparison and combination of features in a hybrid HMM/MLP and a HMM/GMM speech recognition system," *IEEE Transactions on Speech and Audio Processing*, vol. 13, pp. 14–22, January 2005.

[100] E. R. Tufte, *The Visual Display of Quantitative Information*. Graphics Press, 1992.

[101] Student, "The probable error of a mean," *Biometrika*, vol. 6, pp. 1 – 25, March 1908.

[102] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, pp. 80–83, December 1945.

[103] "http://fsweb.berry.edu/academic/education/vbissonnette/tables/mwu.pdf."

[104] N. Smirnov, "Table for estimating the goodness of fit of empirical distributions," *Annals of Mathematical Statistics*, vol. 19, pp. 279 – 281, Jun. 1948.

[105] F. J. Massey, "The Kolmogorov-Smirnov test for goodness of fit," *Journal of the American Statistical Association*, vol. 46, pp. 68 – 78, March 1951.

[106] L. H. Miller, "Table of percentage points of Kolmogorov statistics," *Journal of the American Statistical Association*, vol. 51, pp. 111 – 121, March 1956.

[107] H. Hotelling, "The generalization of Student's ratio," *Annals of Mathematical Statistics*, vol. 2, pp. 360–378, 1931.

[108] H. Hotelling, "Multivariate quality control," *Techniques of Statistical Analysis*, pp. 111–184, 1947.

[109] G. Fasano and A. Franceschini, "A multidimensional of Kolmogorov-Smirnov test," *Monthly Notices Royal Astronomy Society*, vol. 225, pp. 155–170, 1987.

[110] R. H. C. Lopes, I. Reid, and P. R. Hobson, "The two-dimensional Kolmogorov-Smirnov test," in *International Workshop on Advanced Computing and Analysis Techniques in Physics Research*, (Amsterdam), April 2007.

[111] P. Mahalanobis, "On the generalised distance in statistics," *Proceedings of the National Institute of Science of India*, vol. 12, pp. 49–55, 1936.

[112] J. Jackson, "Principal components and factor analysis: part 1," *Journal of Quality Technology*, vol. 12, pp. 201–213, October 1980.

[113] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of Eugenics*, vol. 7, pp. 179–188, 1936.

[114] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. 13, pp. 260–269, April 1967.

[115] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains," *Annals of Mathematical Statistics*, vol. 41, no. 1, pp. 164–171, 1979.

[116] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, pp. 257–286, February 1989.

[117] E. Rodríguez, , B. Ruíz, Ángel García-Crespo, and F. García, "Speech/speaker recognition using a HMM/GMM hybrid model," *Lecture Notes in Computer Science*, vol. 1206, pp. 227–234, 1997.

[118] R. Kitching, "A simple simulation model of dispersal of animals among units of discrete habitats," *Oecologia*, vol. 7, pp. 95–116, 1971.

Pierre Roduit

firstname.lastname@a3.epfl.ch
1979-07-21
Swiss

Education

2005-2008	Ph.D. at the *Laboratoire de Production Microtechnique* of Prof. J. Jacot and at the *Distributed Intelligent Systems and Algorithms Laboratory* of Prof. A. Martinoli, Ecole Polytechnique Fédérale de Lausanne, Switzerland
1998-2003	M.Sc. in Microengineering, Ecole Polytechnique Fédérale de Lausanne, Switzerland
1993-1998	Maturité scientifique (High School Diploma), Sion, Switzerland