Behavioral Analysis of Mobile Robot Trajectories Using a Point Distribution Model

Pierre Roduit[1 2], Alcherio Martinoli[1], and Jacques Jacot[2]

[1] Swarm-Intelligent System Group (SWIS), École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland
[2] Laboratoire de Production Microtechnique (LPM-IPR), École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland

Abstract. In recent years, the advent of robust tracking systems has enabled behavioral analysis of individuals based on their trajectories. An analysis method based on a Point Distribution Model (PDM) is presented here. It is an unsupervised modeling of the trajectories in order to extract behavioral features. The applicability of this method has been demonstrated on trajectories of a realistically simulated mobile robot endowed with various controllers that lead to different patterns of motion. Results show that this analysis method is able to clearly classify controllers in the PDM-transformed space, an operation extremely difficult in the original space. The analysis also provides a link between the behaviors and trajectory differences.

1 Introduction

The development of vision-based tracking systems brings about an easy way to extract trajectory data. Consequently, an ever-increasing number of domains are using it for behavior and trajectory analysis, like video surveillance [11, 13, 5, 12, 8], sports analysis [1] and ethology [4]. Behavioral analysis has also been done on human trajectories in a virtual environment [14, 15] or on autonomous-robot trajectories [16, 17, 10]. All these applications aim to classify an individual from its trajectory, to analyze the movement differences between individuals, or to create a motion model of animals or insects.

This paper addresses the development of tools to analyze the motion of robots, or more generally people or animals, by means of their trajectories. As we will explain in further detail, robots were chosen as trajectory generators for their repeatability and behavioral controllability which natural being are lacking of. For the analysis, we use a Point Distribution Model (PDM) [2], a kind of deformable template. This model was often used to detect object shapes in an image, but it can also be used for trajectory modeling [3]. It is able to take into account spatial and temporal information, but in our experiments we focused on purely spatial analysis. We are more interested in the way the individual is moving and not by the time it needs to travel a given distance.

The paper is organized as follows. In section 2, the experimental method used will be presented, followed in the next section by a description of the Point Distribution Model. Section 4 will present the results and a discussion will close the paper.

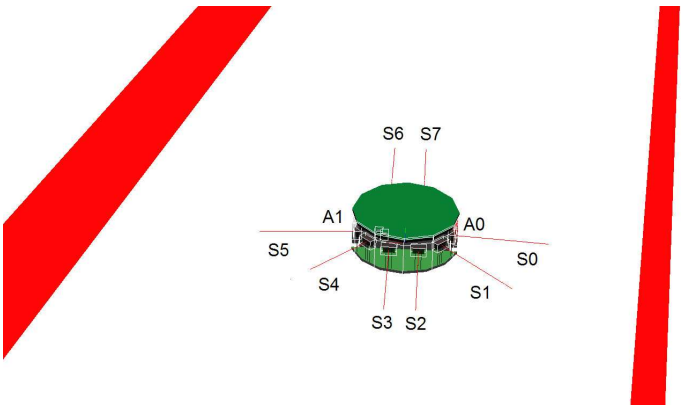
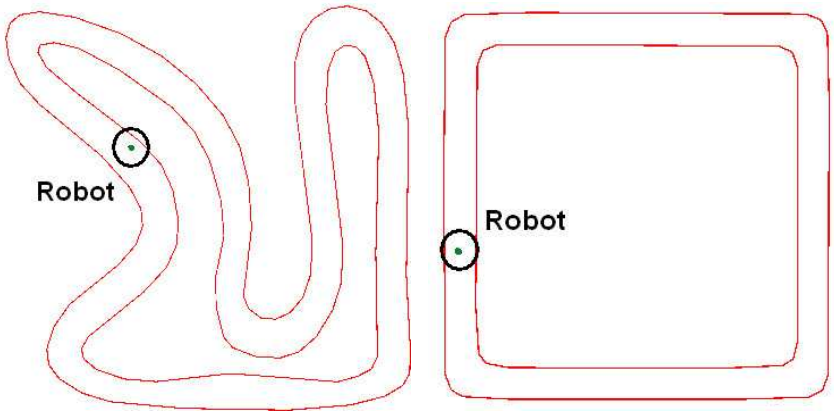


Fig. 1. Image of the simulated mobile robot, with its 8 infrared sensors (S0 to S7) and two wheels (A0 and A1). The two walls of the circuit can also be seen



(a) (b)

Fig. 2. The two circuits simulated in Webots (1 on the left, 2 on the right)

2 Experimental Method

In order to collect hundreds of trajectories in a very short time and perform a first exploration of PDMs as tool for behavioral analysis, we have decided to work with mobile robots. They may be embedded in a physical environment in the same way as natural creatures, but are more easily programmed to produce repeatable behaviors. Moreover, they can be small and therefore the experimental setup can easily fit into a room and simple video tracking systems can extract their positions. Finally, their behavioral repertoire can be quite rich and their controllers can achieve fairly high degrees of complexity, spanning from purely reactive to more deliberative behaviors.

In order to further speed up the trajectory generation, we carried out our experiments using a simulated, miniature differential-wheel robot endowed with eight proximity sensors (Fig. 1). We simulated it in *Webots* [9], a realistic simulator reproducing individual sensors and actuators with noise, nonlinearities, and dynamic effects such as slipping and friction. The resulting simulation is sufficiently faithful for the controllers to be transferred to real robots without changes and for the simulated robot behaviors to be very close to those of the real robots, as shown in several previous papers (see for instance [6]).

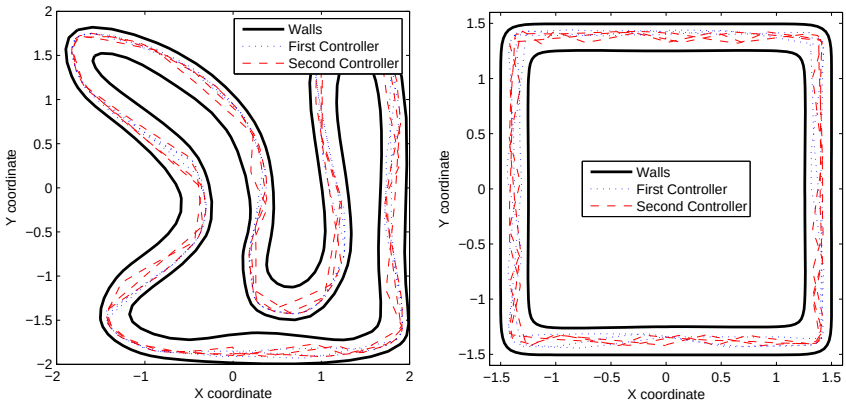
Figure 2 shows the shapes of the two circuits used for our simulation. They share common features such as being characterized by only one lane and a closed loop, however, the length and curvature differ between them. We use two circuits for our experiments in order to test the validity of our analysis.

Table 1. Description of the two controllers used for the experiments

Controller 1	Controller 2
If $\sum_0^2 S_i < T \Rightarrow \begin{cases} A_0 = V \\ A_1 = -V \end{cases}$ Else if $\sum_3^5 S_i < T \Rightarrow \begin{cases} A_0 = -V \\ A_1 = V \end{cases}$ Else $\begin{cases} A_0 = V \\ A_1 = V \end{cases}$	$S_l = \sum_0^2 S_i$ $S_r = \sum_3^5 S_i$ $A_0 = V \cdot \left(1 + \frac{K \cdot (S_l - S_r)}{2 \cdot (S_l + S_r)}\right)$ $A_1 = V \cdot \left(1 + \frac{K \cdot (S_r - S_l)}{2 \cdot (S_l + S_r)}\right)$
$S_0 \ldots S_5$ are the robot sensors as shown in Figure 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 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	

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 around the track at the same average speed, but using different methods of avoiding 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 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.



(a) (b)

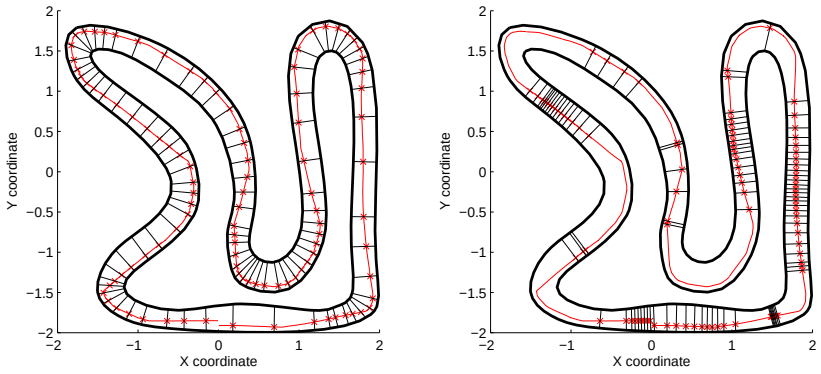
Fig. 3. 6 trajectories (3 for each controller) of the robot's movement simulated on the 2 circuits

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 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.

2.1 Trajectory Sampling

In order to apply the *Point Distribution Model* presented in section 3, each trajectory must be sampled with the same number of points. For our previous experiments, presented in [3], the sampling of the trajectories was done with lines orthogonal to a reference trajectory that needed to be chosen. However, this solution is cumbersome and limits the number of sampling points. Therefore, we developed a new sampling methodology based on the circuit instead of a reference trajectory. The inner and outer wall of

the circuit were modeled with b-splines and sampling gates were created as orthogonal as possible to the two walls. A fixed number of gates was then selected based on three different criteria: the distance between gates (Sampling Method A), the curvature (SM-B), and the linearity of the circuit (SM-C). The first criterion selects gates that are equidistant from each other (the distance is measured between the midpoints of the gates), the second places more gates in the curves (as shown in Figure 4(a)), and the last places more gates in the straight sections (Figure 4(b)). This method allows us to very easily modify the number of gates per circuit. The performance of the three placement strategies will be compared in Section 4.3.



(a) (b)

Fig. 4. Sampling of the trajectory with gates as orthogonal to the wall as possible, with more gates in the curves (SM-B, left) and more gates in the straight lines (SM-C, right)

3 Modeling of the Trajectories

We have slightly modified the representation we used in [3] to accommodate the new sampling method presented above. Each trajectory k is represented as an ordered set of N points corresponding to the intersections of the trajectory with the sampling gates. Each point can be expressed as the linear position on the ith sampling gate π_i^k. Each position π_i^k has a value between 0 (the inner wall) and 1 (the outer wall). Therefore the trajectory τ_k can be expressed as:

$$\tau_k = \left[\pi_1^k \dots \pi_N^k\right]^T. \tag{1}$$

The covariance matrix of the trajectories is

$$\boldsymbol{S} = \frac{1}{K-1}\sum_{k=1}^{K}(\tau_k - \overline{\tau})(\tau_k - \overline{\tau})^T = \boldsymbol{P}\cdot\boldsymbol{\Lambda}\cdot\boldsymbol{P^{-1}}, \tag{2}$$

where $\boldsymbol{P} = [P_1 \dots P_r \dots P_R]$ is the matrix of the eigenvectors P_r, $\boldsymbol{\Lambda}$ the diagonal matrix containing the eigenvalues of $\boldsymbol{S}$, K is the number of trajectories in the set, and where $R = \min(2N, K) - 1$ is the number of degrees of freedom of the set.

Each trajectory τ_k in the set can be decomposed into an average trajectory and a linear combination of deformation modes:

$$\tau_k = \overline{\tau} + \boldsymbol{P} \cdot B_k \tag{3}$$

$$B_k = \boldsymbol{P}^{-1}(\tau_k - \overline{\tau}). \tag{4}$$

Equations 3 and 4 correspond to the projection from the deformation space (B_k) to the trajectory space (τ_k) and the projection from the trajectory space to the deformation space, respectively.

The computation of matrix $\boldsymbol{P}$ corresponds to the Principal Component Analysis (PCA) [7] of the trajectory set. The first vector P_1 corresponds to the direction of maximal variance in the trajectory space. The second vector P_2 corresponds to the direction of maximal variance orthogonal to P_1. The other vectors are found likewise. In most cases, this construction implies that most of the deformation energy will be contained in the first few deformation modes.

The *Point Distribution Model* [2] affords the transformation from the space of the trajectories (τ_k) to the space of the modes (B_k).

3.1 Inter-cluster Distance

In this section we will describe the inter-cluster measure we used for our experiments.

Multivariate normal data tends to cluster about the mean vector, $\mu_{cluster}$, falling in an ellipsoidal cloud whose principal axes are the eigenvectors of the covariance matrix. The Mahalanobis distance, r, takes into account the covariance of the cluster, $S_{cluster}$, to calculate the distance from a point X to a cluster.

$$r = \sqrt{(X - \mu_{cluster})^T \cdot {\boldsymbol{S}_{\boldsymbol{cluster}}}^{-1} \cdot (X - \mu_{cluster})} \tag{5}$$

If normal data is projected on a unidimensional axis, a unitary Mahalanobis distance is equivalent to a Euclidean distance of the square root of the data variance along this axis (standard deviation). Thus, the points of unitary Mahalanobis distance to a cluster forms a ellipsoid.

As a measure of distance between two clusters, we can use a combination of the two Mahalanobis distances; from the mean of one cluster to the other, and vice versa. If r_1^2 is the Mahalanobis distance from the first cluster mean to the second cluster and r_2^1 the Mahalanobis distance from the second cluster mean to the first cluster, d_{12} is a measure of the inter-cluster distance.

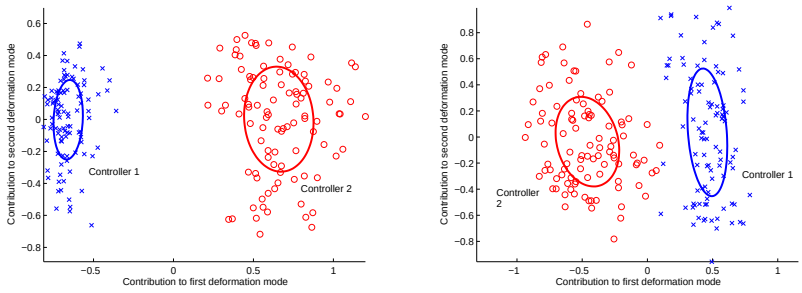
$$d_{12} = \frac{r_1^2 \cdot r_2^1}{r_1^2 + r_2^1} = d_{21} \tag{6}$$

A unitary inter-cluster distance is equivalent to a Euclidean distance between the two cluster means which is equal to the sum of the standard deviation of the projected multivariate cluster data on the axis connecting the two cluster means.

4 Results and Discussion

To show the performance of our method for clustering controller trajectories, we acquired 200 trajectories (100 for each controller presented in Table 1) on the two circuits (Figure 2 and 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). Then, we model all the trajectories using the PDM presented in Section 3. Figure 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.

4.1 Analysis Using the First two Modes of the PDM



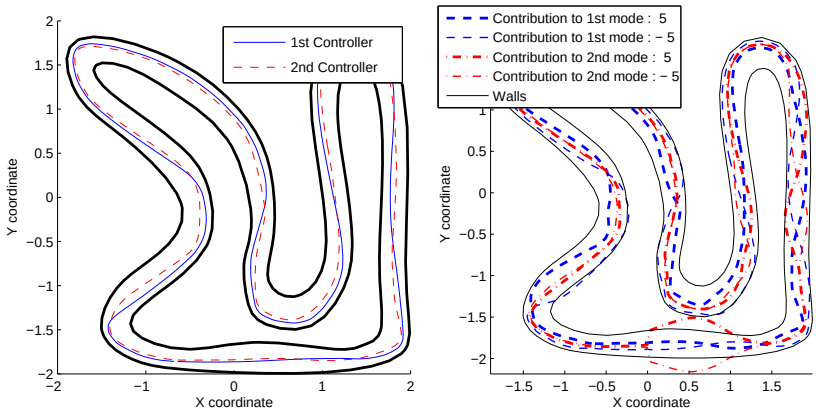
(a) (b)

Fig. 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

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 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 3.1, $d = 4.3$ for the first circuit and $d = 2.4$ for the second circuit.

4.2 Prototype Trajectories

Figure 6(a) displays the mean of each controller's trajectories. These trajectories are prototypes of each controller. Slight differences appear at certain places in the circuit; these are the places where the controllers can be differentiated.



(a) (b)

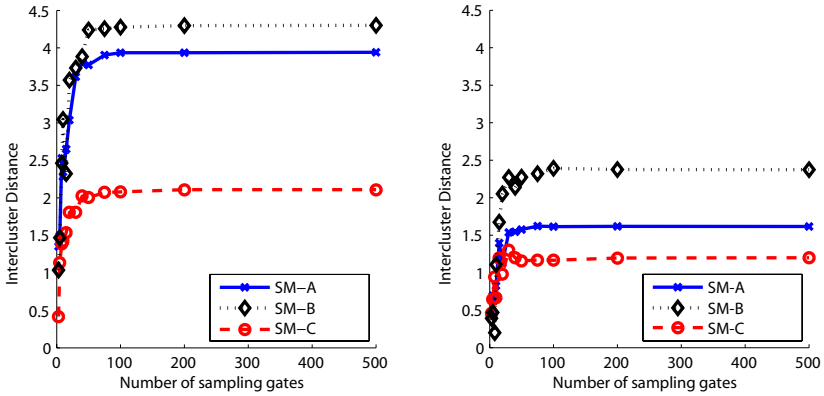
Fig. 6. On the left, prototype trajectory of the two controllers on the first circuit. On the right, synthetic trajectories resulting from a contribution of 5 to one of the first two modes (first circuit)

Figure 6(b) shows the synthetic trajectories resulting from a positive or negative contribution of value 5 to the first or the second deformation mode for the first circuit. In the straight section on the right side of the circuit, we can see that the first mode is out of phase with the second mode, such that they alternate. The second mode corresponds to major trajectory variations in the straight section along the bottom of the circuit.

4.3 Variation of the Trajectory Sampling Methods and Number of Points

To evaluate the number of gates needed to achieve a good classification of the trajectories, we calculate the inter-cluster distance as a function of the number of sampling gates (from 3 to 500) for the three sampling methods described in section 2.1 and for both circuits. Figures 7(a) and 7(b) present the results of our experiments. We can see that a very small number of gates (3) is insufficient to separate the two controllers. Increasing the number of gates results in a fast gain in cluster separability, until we reached 50 gates. After this point, it has hardly any effect on the analysis, aside from requiring additional computational power for calculating the PCA. We can also see that the three gate placement strategies (equidistant, more in the curves, more in the straight sections) do not have the same influence on clustering performance. Placing gates in the straight sections is clearly the worst solution, while using gates in the curves is clearly the best solution for our experimental setup. Equidistant gates yield a solution in between the other two, but are clearly not so good as emphasizing the curves. This result can be directly traced to the structure of the controllers. They have the same behavior in the straight sections, where there is no interaction with the walls. Curves imply more interaction with the walls, and therefore elicit more behavioral differences between the two controllers. It can be foreseen that if the controllers had the same behavior in the curves, but different in the straight sections, the performance would be inverted. As an-

other result of the controller's structure, the separability of the clusters on the second circuit is clearly worse than on the first, because there are not as many curves to separate the behavior of the two controllers.



(a) (b)

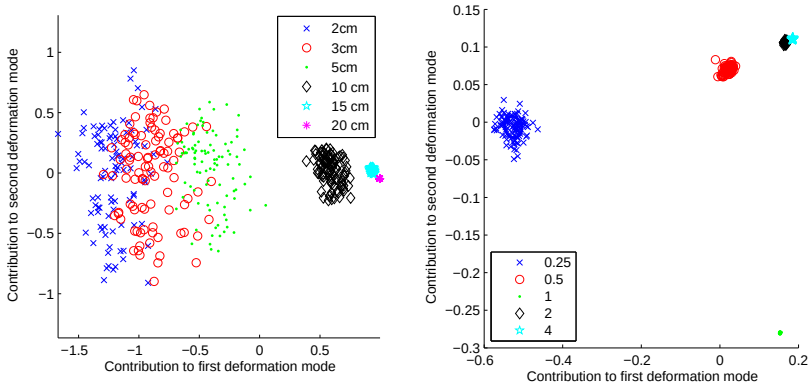
Fig. 7. Variation of the inter-cluster distance based on the first two modes as a function of the number of sampling gates, for the three methods presented in 2.1 and for two circuits (circuit 1 on the left and circuit 2 on the right)

4.4 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 1, the reactivity corresponds to K and the overall speed to V.

Figure 8(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 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 sensory ranges are mainly separated using the first deformation mode in Figure 8(a): one variation axis (sensory range) corresponds to one dimension for classifying the different clusters.

Figure 8(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 8(b).



(a) (b)

Fig. 8. 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

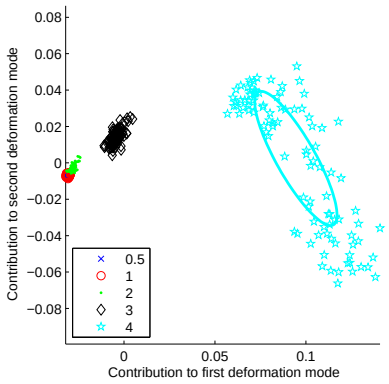


Fig. 9. Analysis of the variation of the overall robot speed (K), for sensory range of 20 cm

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 9 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 9.

5 Conclusion and Perspectives

We have presented a method for using a PDM to analyze a robot's behavior from its trajectory on a closed circuit. Applied to trajectories of the same simulated robot driven by two different reactive controllers, it shows a complete separation of the controllers in the space of the first two deformation modes of the PDM. The controller clusters can be separated by a line, affording an easy classification of the trajectories. Quality of separation depends on the sampling method, the circuit characteristics, and the software and hardware parameters of the robot. The analysis of the prototype trajectory of each controller shows the main differences between the controllers. Variation in three other parameters, such as the robot speed, the controller reactivity, and the sensory range, implies that all of these parameters can be clustered with our method.

Even though the clustering method is not sophisticated (Principal Component Analysis is a common tool), the fact that behavioral features can be distinguished makes it very interesting. So far, only one variation axis has been analyzed at a time (circuit shape, controller description, sensory range, controller reactivity, and overall robot speed). This kind of experiment affords us to separate the trajectories using only the first two dimensions of the PDM. More complex setups might need more than two dimensions to achieve good clustering of the trajectories. Finding two combinations of hardware and software that provide different trajectories which can not be separated with our method would help us to understand its limitations. However, our goal is not to distinguish two different implementations of the same behavior, but rather to classify different behaviors.

To validate the results obtained with the trajectories of the robot simulated in Webots, we will create a similar circuit with a real robot, using a vision-based tracking system. The same analysis will be applied to real trajectories and results will be compared with the results gathered in simulation. Another challenge will be to extend our method to trajectories not bound to a closed circuit. The ultimate goal will be to model robot trajectories in an open space.

5.1 Acknowledgments

Pierre Roduit and Alcherio Martinoli are currently sponsored by two Swiss National Foundation grants (Nr. 200021-105565 and PP002-68647 respectively).

References

1. M. Bertini, A. Del Bimbo, and W. Nunziati. Highlights modeling and detection in sports videos. *Pattern Analysis and Application*, 7(4):411–421, 2005.
2. T. Cootes, C. Taylor, and D. Cooper. Active shape-models - their training and applications. *Vision and Image Understanding*, pages 38–59, 1995.
3. Yuri Lopez de Meneses, Pierre Roduit, Florian Luisier, and Jacques Jacot. Trajectory analysis for sport and video surveillance. *Electronic Letters on Computer Vision and Image Analysis*, 5(3):148–156, 2005.
4. M. Egerstedt, T. Balch, F. Dellaert, F. Delmotte, and Z. Khan. What are the ants doing ? vision-based tracking and reconstruction of control programs. pages 4193–4198. IEEE International Conference on Robotics and Automation, IEEE Computer Society, April 2005.
5. W. E. L. Grimson, C. Stauffer, R. Romano, and L. Lee. Using adaptive tracking to classify and monitor activities in a site. pages 22–29. Conference on Computer Vision and Pattern Recognition, IEEE Computer Society, June 1998.
6. A. T. Hayes, Alcherio Martinoli, and R. M. Goodman. Distributed odor source localization. In Gardner J. W. Nagle H. T. and Persaud, editors, *Special Issue in Artificial Olfaction*, volume 2, pages 260–271. IEEE Sensors Journal, 2002.
7. J. Jackson. Principal components and factor analysis: part 1. *Journal of Quality Technology*, 12:201–213, October 1980.
8. Nail Johnson and David Hogg. Representation and synthesis of behaviour using gaussian mixtures. *Image and Vision Computing*, 20:889–894, 2002.
9. Olivier Michel. Webots: Professional mobile robot simulation. *Journal of Advanced Robotics Systems*, 1(1):29–42, 2004.
10. Ulrich Nehmzov. Quantitative analysis of robot-environment interaction towards ”scientific mobile robotics”. *Robotics and Autonomous Systems*, 44:55–68, 2003.
11. J. Owens, A. Hunter, and E. Fletcher. Novelty detection in video surveillance using hierarchical neural networks. *Lecture Notes in Computer Science*, 2415:1249–1254, 2002.
12. Fatih Porikli. Learning object trajectory patterns by spectral clustering. volume 2, pages 1171–1174. IEEE Conference on Multimedia and Expo, IEEE, June 2004.
13. P. Remagnino, T. Tan, and K. Baker. Agent orientated annotation in model based visual surveillance. page 857. Sixth International Conference on Computer Vision, IEEE Computer Society, 1998.
14. C. Sas, G. O’Hare, and R. Reilly. A performance analysis of movement patterns. *Lecture Notes in Computer Science*, 3038:954–961, 2004.
15. C. Sas, G. O’Hare, and R. Reilly. Virtual environment trajectory analysis: a basis for navigational assistance and scene adaptivity. *Future Generation Computer Systems*, 21 (7):1157–1166, Jul 2005.
16. G. Schöner, M. Dose, and C. Engels. Dynamics of behavior : theory and applications for autonomous robot architecture. *Robotics and Autonomous Systems*, 16:213–245, 1995.
17. Thim Smithers. On quantitative performance measures of robot behaviour. *Robotics and Autonomous Systems*, 15:107–133, 1995.