A Quantitative Method for Comparing Trajectories of Mobile Robots Using Point Distribution Models

Pierre Roduit, Alcherio Martinoli and Jacques Jacot

Abstract— **In the field of mobile robotics, trajectory details are seldom taken into account to qualify robot performance. Most metrics rely mainly on global results such as the total time needed or distance traveled to accomplish a given navigational task. Indeed, usually mobile roboticists assume that, by using appropriate navigation techniques, they can design controllers so that the error between the actual and the ideal trajectory can be maintained within prescribed bounds. This assumption indirectly implies that there is no interesting information to be extracted by comparing trajectories if their variation is essentially resulting from uncontrolled noisy factors. In this paper, we will instead show that analyzing and comparing resulting trajectories is useful for a number of reasons, including model design, system optimization, system performance, and repeatability. In particular, we will describe a trajectory analysis method based on Point Distribution Models (PDMs). The applicability of this method is demonstrated on the trajectories of a real differential-drive robot, endowed with two different controllers leading to different patterns of motion. Results demonstrate that in the space of the PDM, the difference between the two controllers is easily quantifiable. This method appears also to be extremely useful for comparing real trajectories with simulated ones for the same set-up since it affords an assessment of the simulation faithfulness before and after appropriate tuning of simulation features.**

I. INTRODUCTION

Behavioral analysis based on trajectories is a principal field of research, mostly developed for security applications. However, in mobile robotics, methods to quantify differences in trajectories are lacking, even though they can help analyze robotic experiments more scientifically [1], [2]. As a mobile robot can be completely designed and controlled, one might think that there is no need to analyze its trajectories, as they could be predicted in advance. However, as miniaturization tends toward an increase of the sensor or control noise, an evaluation of the hardware influence on the quality of the robot trajectories can help the design optimization. Moreover, a quantitative analysis of a model for the mobile robot based on its trajectories can assist in improving its correspondence with reality. Quantitative trajectory analysis has already been developed for pedestrian and vehicle trajectories [3], [4] and also for human motion in virtual environments [5].

In this article, we want to introduce the Point Distribution Model (PDM) as a tool for analyzing trajectories. The PDM is a deformable template that was first used in computer vision to detect shapes in an image [6], but it can also be extended to model trajectories [7]. Trajectories represent a good point to evaluate the overall performance of a mobile robotic system and/or its corresponding models: they are influenced by the overall design of the targeted system and/or its models and express the interplay of several hardware and software components (sensors, actuators, motion control, and so on). Possible applications in mobile robotics are as follows:

1) Comparing trajectories generated by the same vehicle in order to analyze their repeatability.
2) Comparing trajectories generated by different vehicles (hardware or software differences) for classification purposes [8].
3) Optimizing the performance of a system or its cost and quantitatively assess the impact of different hardware and software potential choices on the resulting trajectory during the design and development phase.
4) Comparing trajectories produced by a model and those generated by the targeted system itself. In this case, the goal is to increase the model's faithfulness.

This paper will illustrate how PDMs can be used for applications 2 and 4. The objectives are two-fold. First, we will demonstrate that the classification performance achieved in simulation in [8] can be reproduced on a real setup. Second, we will show that our trajectory-analysis method is a helpful tool for quantitatively improving the simulation faithfulness to reality.

The paper is organized as follows. A short presentation of the Point Distribution Model (section II) will be followed in section III by the description of both real and simulated experimental setups. Then section IV will present the results and a discussion will close the paper.

II. POINT DISTRIBUTION MODEL

The basic premise of the PDM is to model shapes using their key points. The method for selecting these points is closely bound to the experimental case, but, as soon as the points are selected, the method is completely general and can be applied to all kind of trajectories. However, the quality of the resulting model will substantially depend on the trajectory sampling.

Thus, each trajectory k is represented as an ordered set of N points corresponding to the sampled points. Each point is represented by its spatial position. For our sampling method (section III-C), without loss of generality, this spatial position can be expressed as the position π_i^k on the ith sampling gate. Therefore the trajectory τ_k can be expressed as:

Pierre Roduit is with the Swarm-Intelligent Systems Group and the Laboratoire de Production Microtechnique, École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland pierre.roduit@epfl.ch

Alcherio Martinoli is with the Swarm-Intelligent Systems Group alcherio.martinoli@epfl.ch

Jacques Jacot is with the Laboratoire de Production Microtechnique jacques.jacot@epfl.ch

$$\tau_k = \left[\pi_1^k \dots \pi_N^k\right]^T. \quad (1)$$

The covariance matrix of the trajectories is

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

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 is the number of degrees of freedom of the set. As a trajectory data set cannot have more degrees of freedom than the number of spatial dimension (N in this case; $2N$ if, for instance, x and y coordinates of each point are used) and than the number of trajectories in the set minus one ($K-1$):

$$R \leq \min(N, K-1). \quad (3)$$

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

$$\tau_k = \overline{\tau} + \boldsymbol{P}\cdot B_k \quad (4)$$

$$B_k = \boldsymbol{P}^{-1}(\tau_k - \overline{\tau}). \quad (5)$$

Equations 4 and 5 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) [9] of the trajectory set. The first vector P_1 corresponds to the direction of maximal variance in the trajectory space and is also called the first deformation mode. The second vector P_2 corresponds to the direction of maximal variance orthogonal to P_1. The other vectors are found similarly. 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* corresponds to the representation of the trajectories by a set of chosen points transformed from the space of the trajectories (τ_k) to the space of the modes (B_k).

The utility of PCA is arguable. For classification purpose, the Linear Discriminant Analysis (LDA) [10] will return the axis of best separation between the trajectory clusters and is thus more efficient. If the goal is to extract the principal axes of variation of a unique trajectory cluster, PCA is also not always the best solution, as it can only model linear deformations. More complex techniques such as Laplacian Eigenmaps [11] or Locally Linear Embedding [12] can reduce dimensions in non-linear manifolds. However, as our goal is to demonstrate if two set of trajectories are similar or not, the PCA gives us entire satisfaction. As an unsupervised technique, it is less sensitive to noise than LDA and shows differences in the main dimensions of variation, allowing for an incremental analysis of deformation modes according to their energy and, therefore, to disregard low-energy deformation modes heavily influenced by noise in trajectory acquisition where appropriate.

A. Inter-cluster Distance

The transformation to the space of the PCA is not sufficient to compare trajectories. A suitable measure of the similarity in this space is needed to achieve a quantization of the difference between the trajectory clusters in the space of the PCA. For this, Euclidean distance is not the best solution, as it does not take into account the size or covariance of the clusters. Thus, a measure based on the Mahalanobis distance is used. The Mahalanobis distance (r) from a point X to a cluster of points takes into account the covariance matrix of the cluster $S_{cluster}$ and the cluster mean $\mu_{cluster}$.

$$r = \sqrt{(X-\mu_{cluster})^T \cdot \boldsymbol{S_{cluster}}^{-1} \cdot (X-\mu_{cluster})} \quad (6)$$

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 from a cluster form an ellipsoid.

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 respectively the second cluster,

$$\boldsymbol{W} = \frac{\sum_{i=1}^{n_x}(X-\overline{X})(X-\overline{X})^T + \sum_{i=1}^{n_y}(Y-\overline{Y})(Y-\overline{Y})^T}{n_x+n_y-2}. \quad (7)$$

Thus, similarly to Eq. 6, the distance d between the two clusters can be calculated as:

$$d = \sqrt{(\overline{X}-\overline{Y})^T \cdot \boldsymbol{W}^{-1} \cdot (\overline{X}-\overline{Y})}. \quad (8)$$

d can be linked to the Hotelling's T^2 statistics [13]:

$$t^2 = \frac{n_x \cdot n_y}{n_x+n_y} d^2 \quad (9)$$

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

III. CASE STUDY

To demonstrate the usability of our method, we choose a simple case study. However, our methods 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. Fig. 1 shows the setup. A miniature differential-drive robot, the e-puck [14], was made to drive continuously around this circuit. The e-puck is endowed with eight proximity sensors (Fig. 4), and a more thorough description of its controllers can be found in section III-B. To extract the robot trajectories, an overhead camera was fixed above the arena. SwisTrack [15], a video-based tracking system, was used to compute the agent's position. Background subtraction, updated with the average intensity of the current image, was used to make the resulting segmentation less sensitive to variations in ambient lighting conditions. For the tracking

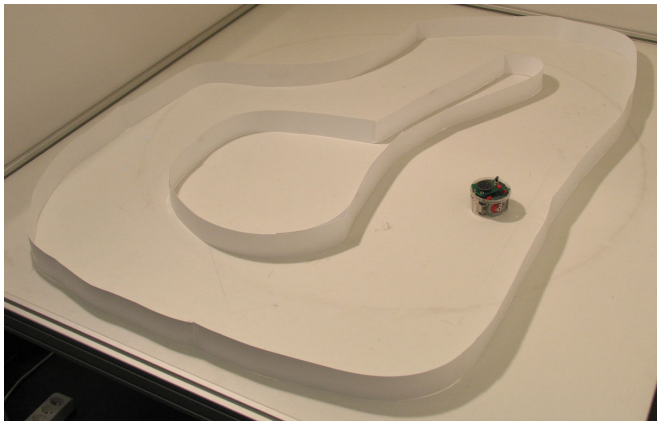

Fig. 1. The real setup used for the experiments. We can see the e-puck robot and the circuit walls

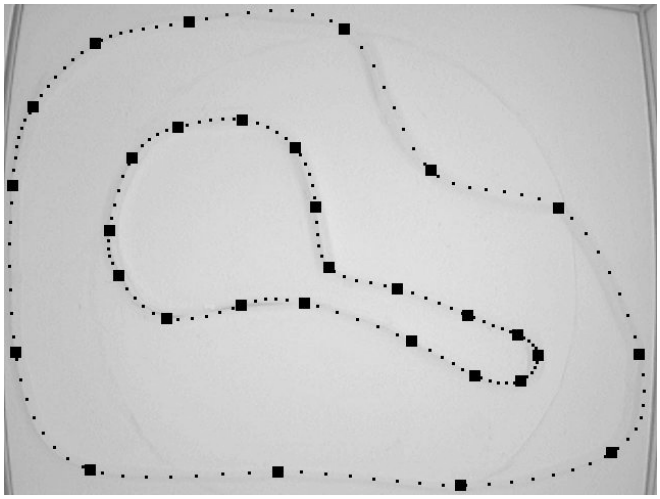

Fig. 2. Approximation of the circuit walls by multiple b-splines. The separation between the splines are indicated with black squares

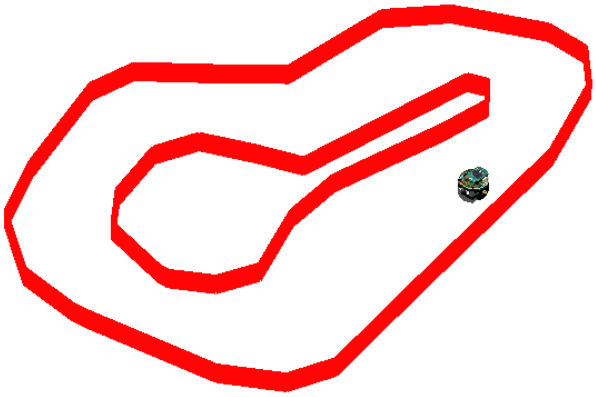

Fig. 3. The simulated setup used for the experiments that reproduce the real experiment of Figure 1

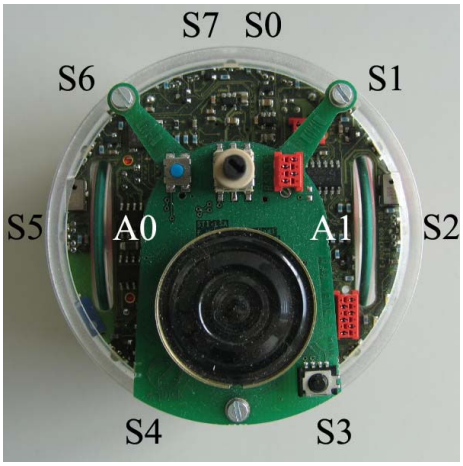

Fig. 4. Top view of the e-puck robot with 8 sensors (S0, . . . ,S7) and two actuators (A0,A1). The front of the robot is facing towards the top of the image

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.

A. Simulation Reproduction

To recreate the real setup, we used *Webots* [16], a realistic simulator whose faithfulness for other robotic platforms has been demonstrated in several previous papers (see for instance [17]). 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. 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 segments, corresponding *Webots* boxes were then created in the simulated world. Fig. 3 shows the resulting setup with the approximated walls and the simulated e-puck.

B. 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

TABLE I

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 4 (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 4 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	

found in Table I. Fig. 4 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.

C. 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 [8]. 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. Fig. 5 shows the sampling gates, the b-spline approximations of the walls and a sampled trajectory.

This sampling method is well adapted for our experimental setup. However, a similar principle can be used to sample trajectories in other environments. The goal is not to sample trajectories based on the time elapsed or on the covered distance: the trajectories are sampled in specific places. This method is quite close to the way a human would do it naturally (e.g., at a specific time, the car went through the crossroad, the robot arrived in a specific area, or the pedestrian went through the door). Comparing trajectories near specific landmarks is our underlying intention.

IV. RESULTS

To demonstrate the performance of our analytical method, two experiments were run. The first one aimed to compare trajectories generated by two different controllers and was run on the real setup presented previously. The second experiment analyzed the quality of the reproduction of the real setup in Webots when tuning the simulation features.

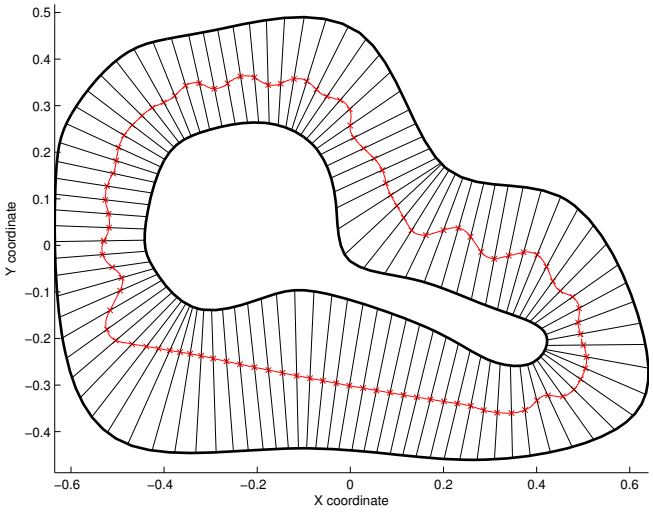

Fig. 5. Sampling of a trajectory with gates as orthogonal to the walls as possible. The crosses indicated the intersections between the gates and the trajectory that will be used for the analysis

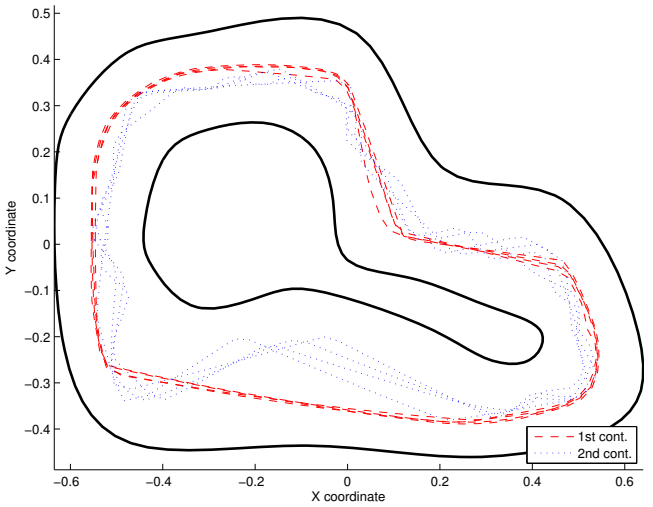

Fig. 6. Four trajectory samples for each controller presented in section III-B. The main difference between the two controllers lies in the lower part of the circuit

A. 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 III-B, and for each run, twenty trajectories were extracted with SwisTrack and sampled using the method presented in section III-C. At the beginning of each run, the e-puck's sensors were initialized, but lighting condition changes (sunlight) happened during and between the runs. Fig. 6 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.

Fig. 7 shows the locations of the 160 trajectories in the space formed by the first two modes of the PDM. The

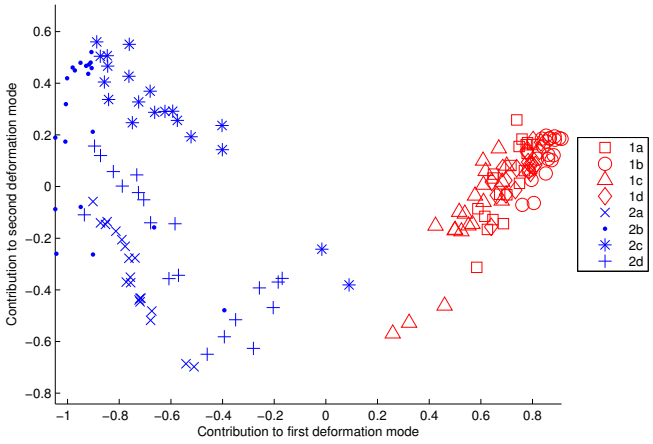

Fig. 7. The first two modes of the PDM analysis of the two controllers (4x20 trajectories per controller) using the real setup. The different clusters of the two controllers can be easily separated

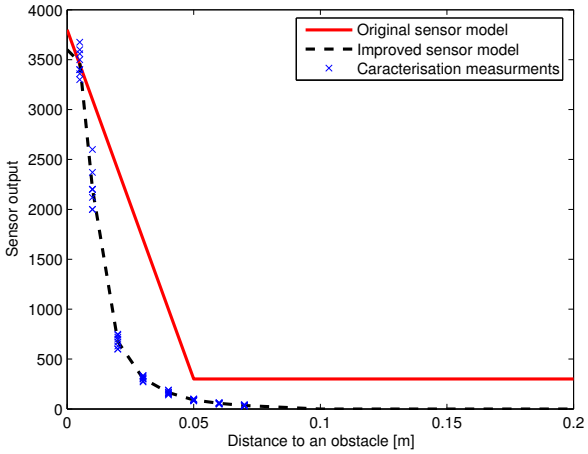

Fig. 8. Real e-puck sensor output and two candidate sensor models. We can see that the sensor output is non-linear and that the usable range is quite short

separability noticeable in the trajectory plot (Fig. 6) also exists in the PDM space. Inter-cluster distance, as defined in section II-A, 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 Eq. 8, 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.

B. Simulation Faithfulness Analysis

Quantifying the similarity of trajectories using a PDM analysis is not only useful to compare two controllers. When we are tuning simulation features and parameters, if we quantify the similarity between simulated and real trajectories, we can evaluate whether or not the modification of specific characteristics is increasing the simulation's faithfulness to reality. To show an example, we simulated an e-puck robot in Webots, as explained in Section III-A 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 command. These three examples could be extended to any other feature or parameter of a 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 command representing the wheel slip.

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

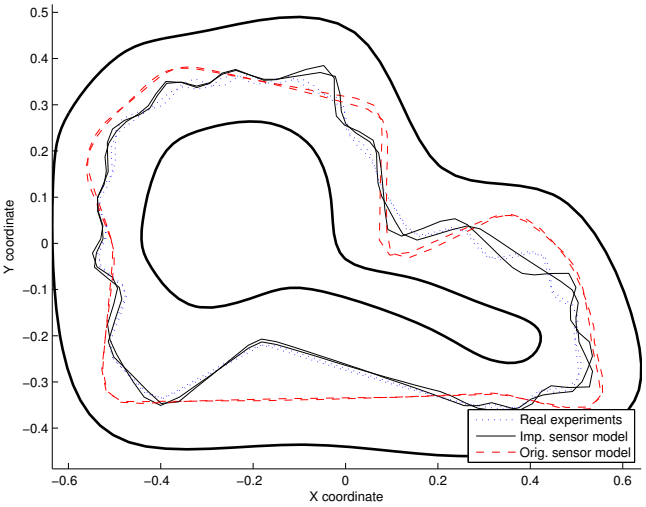

Fig. 9. Two trajectory samples for the real experiments and simulations done with the original and improved sensor models. The trajectories of the improved sensor model are more similar to the trajectories of the real experiments than the ones of the original sensor model

extrapolated from output measurements of the real sensors. Fig. 8 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 while 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

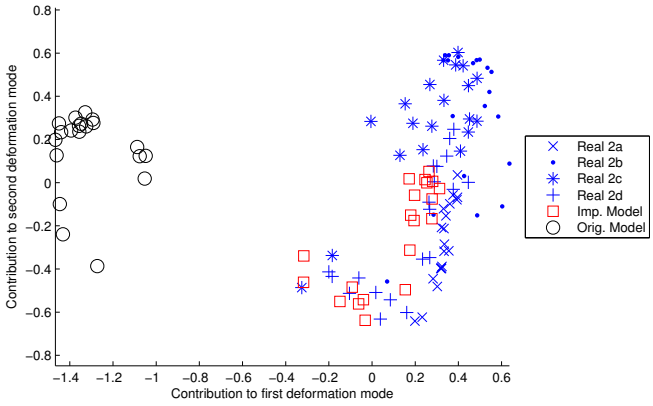

Fig. 10. PDM analysis showing agreement between simulation (2×20 trajectories) and real experiments (4×20 trajectories), for the two proximity sensor models. The cluster of the improved sensor model is much closer to the four real experimental runs than the one of the original sensor model

TABLE II

INTER-CLUSTER DISTANCES BETWEEN THE CLUSTERS REPRESENTED IN FIG. 10, 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

two sensor models. Only the Braitenberg controller was used for this purpose (2nd controller in Table I). 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. 9 shows trajectory samples for the different experiments. All the trajectories were extracted and then sampled as presented in Section III-C. Afterward, we applied our PDM analysis to the trajectories. Fig. 10 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. 9).

Inter-cluster distance (measured using Eq. 8) can be used to quantify the improvement in the accuracy of the simulation. Table II 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.

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

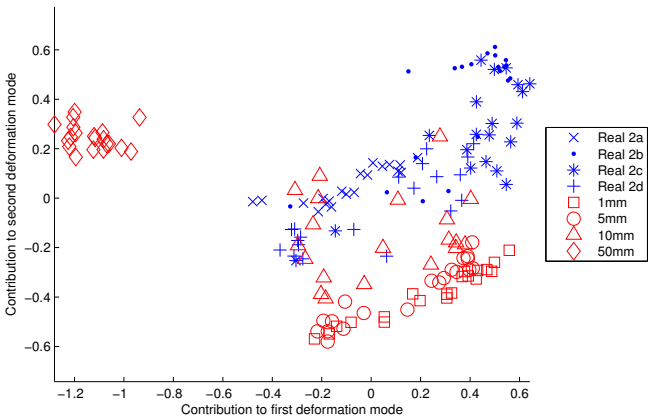

Fig. 11. PDM analysis showing agreement between simulation (4x20 trajectories) and real experiments (4x20 trajectories) for the four values of wall approximation error. We can see that less precision on the approximation of the walls (50 mm) leads to a cluster farther away from the real experiments. 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

(rectangular boxes). To create this representation, we used the b-spline model of the wall used for the sampling (Fig. 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 I). 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 11 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 III.

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

TABLE III

INTER-CLUSTER DISTANCES BETWEEN THE CLUSTERS REPRESENTED IN FIG. 11, 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

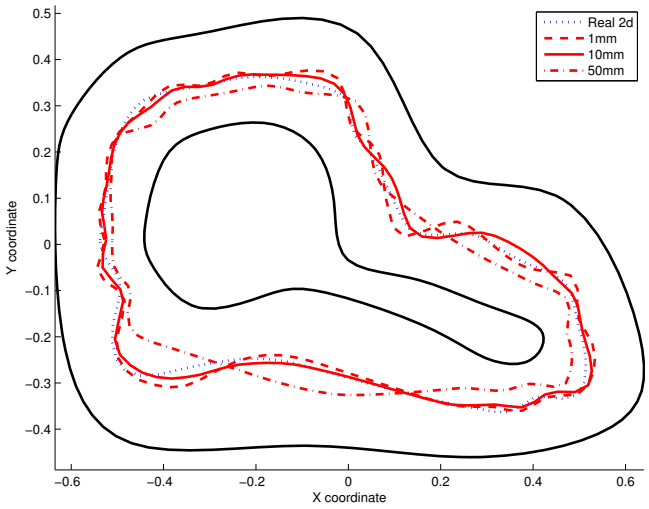

Fig. 12. Average trajectory for the the real experiment (Real 2d) and for the three simulated runs with a wall approximation error of 1, 10 and 50 mm respectively. We can see that for a reduced precision (50 mm), the average trajectory is really different than the real average trajectory (Real 2d). Moreover, 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

average trajectory than more precise ones (1 and 5 mm). This offset can also be seen in the trajectory space (Figure 12). 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.

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 13 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 I) 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

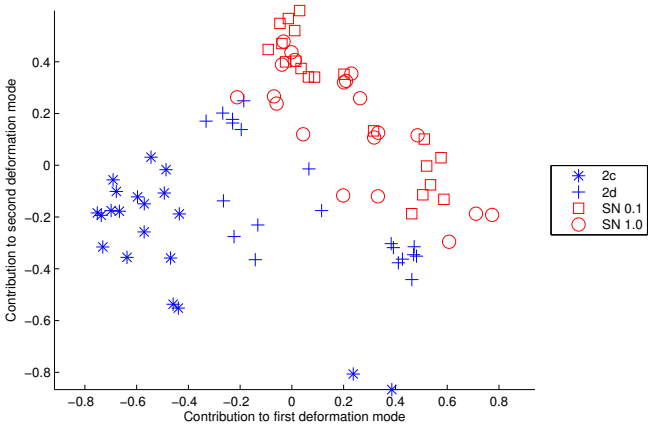

Fig. 13. PDM analysis showing agreement between simulation (2x20 trajectories) and runs 2c and 2d of the real experiments (4x20 trajectories) for the two values of noise (10% and 100%) representing the wheel slip. We can see that the value of the noise has not a big influence on the trajectories

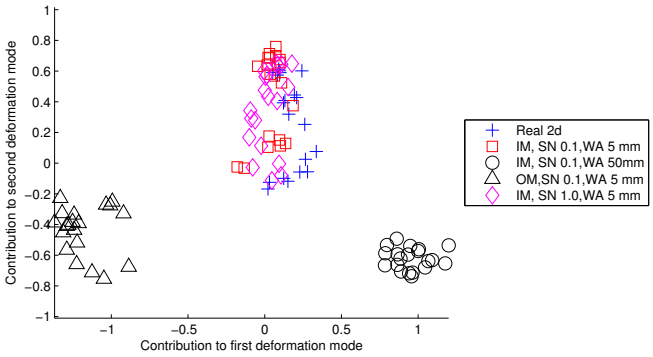

Fig. 14. PDM analysis of one real experimental run (Real 2d) together with the three simulation parameters: the sensor model (original and improved, respectively OM and IM), the reproduction of the wheel slip (SN) and the wall approximation error (WA). We can see that modifying the sensor model and the wall approximation error have major influences on the accuracy of simulation in comparison to the minor ones introduced by a different level of slip noise on the wheels

have had a major impact on the simulation accuracy.

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 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 14 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.

C. 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 do 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 faithfulness.

The inter-cluster distance, as presented in Section II-A, 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 [18] 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.

V. CONCLUSION AND FUTURE WORK

We have presented a method for using a PDM to quantitatively compare mobile robot trajectories. Applied to trajectories of the same mobile robot driven by two different reactive controllers, it can be used to quantify the similarities and the differences between the controllers in the space of the first two deformation modes of the PDM. Furthermore, we showed that this method can be used to compare a real experiment with its reproduction in simulation. Inter-cluster distance, as defined in this paper, allows us to quantify the differences, and thus provides a way to evaluate the relative costs and benefits of specific design choices on the simulation fidelity.

While the experimental setup used in our case study may not be overly sophisticated, the analytical performance of our method is clearly demonstrated and its generality affords an application to more complex setups.

In the future, we would like to show the applicability and usefulness of the PDM-based method for experiments in open space, in more or less complex environmental scenarios.

We also intend to implement a measure of the difference between two trajectory clusters based on the Kolmogorov-Smirnov test or on another test that compares the covariance matrices.

VI. ACKNOWLEDGMENTS

Pierre Roduit and Alcherio Martinoli are currently sponsored by two Swiss National Science Foundation grants (Nr. 200020-113795 and PP002-68647 respectively).

REFERENCES

[1] U. Nehmzov, "Quantitative analysis of robot-environment interaction towards "scientific mobile robotics"," *Robotics and Autonomous Systems*, vol. 44, pp. 55–68, 2003.

[2] T. Smithers, "On quantitative performance measures of robot behaviour," *Robotics and Autonomous Systems*, vol. 15, pp. 107–133, 1995.

[3] F. Porikli, "Trajectory distance metric using hidden markov model based representation," in *Proceedings of IEEE European Conference on Computer Vision (ECCV)*, (Prague), Workshop on PETS, 2004.

[4] F. Porikli, "Learning object trajectory patterns by spectral clustering," vol. 2, pp. 1171–1174, IEEE Conference on Multimedia and Expo, IEEE, June 2004.

[5] 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, Jul 2005.

[6] T. Cootes, C. Taylor, and D. Cooper, "Active shape-models - their training and applications," *Vision and Image Understanding*, pp. 38–59, 1995.

[7] 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.

[8] P. Roduit, A. Martinoli, and J. Jacot, "Behavioral analysis of mobile robot trajectories using a point distribution model," in *SAB06, LNAI*, vol. 4095, pp. 816 – 827, Springer-Verlag, 2006.

[9] J. Jackson, "Principal components and factor analysis: part 1," *Journal of Quality Technology*, vol. 12, pp. 201–213, October 1980.

[10] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of Eugenics*, vol. 7, pp. 179 – 188, 1936.

[11] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Computation*, vol. 15, no. 6, pp. 1373 – 1396, 2002.

[12] L. K. Saul and S. T. Roweis, "Think globally, fit locally: Unsupervised learning of low dimensional manifolds," *Journal of Machine Learning Research*, vol. 4, pp. 119 – 155, 2003.

[13] H. Hotelling, "The generalization of student's ratio," *Annals of Mathematical Statistics*, vol. 2, pp. 360–378, 1931.

[14] "www.e-puck.org."

[15] 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 *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2185 – 2191, October 2006.

[16] O. Michel, "Webots: Professional mobile robot simulation," *Journal of Advanced Robotics Systems*, vol. 1, no. 1, pp. 29–42, 2004.

[17] 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.

[18] N. Smirnov, "Table for estimating the goodness of fit of empirical distributions," *The Annals of Mathematical Statistics*, vol. 19, pp. 279 – 281, Jun. 1948.