Biologically Inspired Mobile Robot Control Robust to Hardware Failures and Sensor Noise

Fabio DallaLibera[1,3], Shuhei Ikemoto[2], Takashi Minato[3], Hiroshi Ishiguro[3,4], Emanuele Menegatti[1], and Enrico Pagello[1]

[1] Department of Information Engineering, University of Padua, Padova, Italy
[2] Department of Adaptive Machine Systems, Osaka University, Osaka, Japan
[3] ERATO, Japan Science and Technology Agency, Osaka, Japan
[4] Departmemt of Systems Innovation, Osaka University, Osaka, Japan

Abstract. Some bacteria present a movement which can be modeled as a biased random walk. Biased random walk can be used also for artificial creatures as a very simple and robust control policy for tasks like goal reaching. In this paper, we show how a very simple control law based on random walk is able to guide mobile robot equipped with an omnidirectional camera toward a target without any knowledge about the robot's actuators or about the robot's camera parameters. We verified, by several simulation experiments, the robustness of the random biased control law with respect to failures of robot's actuators or sensor damages. These damages are similar to the ones which can occur during a RoboCup match. The tests show that the optimal behavior is obtained using a bias which is roughly proportional to the random walk step, with a coefficient dependent on the physical structure of the robot, on its actuators and on and its sensors after the damage. Finally, we validated the proposed approach with experiments in the real world with a wheeled robot performing a goal reaching task in a Middle-Size RoboCup field without any prior knowledge on the actuators and without any calibration of the very noisy omnidirectional camera mounted on the robot.

1 Introduction

When considering challenging environments like forests [1], planetary explorations [2] or game fields where the robots can collide to each other, it is almost impossible to predict in advance all the problems which could arise in the task execution and the possible failures the robot hardware might encounter. However, the current technology seems still to be lacking in providing reliable robots able to cope with hardware failure or uncertainties. Most techniques require a previous identification of the possible accidents that can occur to the robot and the design of specific workarounds for each of them. Some advanced self-modeling techniques were presented [3] but they request intensive computation to estimate the current status of the robot from the sensor information.

Conversely, very simple living beings like bacteria can cope with very complex and variable environments, and often present a highly adaptive and robust behavior despite their structural simplicity. In particular, bacteria are able to sense the concentration of nutrients and direct their movements toward the food molecules while escaping from poisoning substances, without any complex planning strategy or fault detection system. This process is called *chemotaxis*.

The behavior of Escherichia Coli, in the following referred as E. Coli, has been deeply studied [4]. These organisms utilize a simple biased random walk for their movement. In particular, this bacterium has only two way of moving, rotating clockwise or counter-clockwise. When it rotates counter-clockwise the rotation aligns its flagella into a single rotating bundle and it swims in a straight line. Conversely clockwise rotations break the flagella bundle apart and the bacterium tumbles in place. The bacterium cannot therefore choose the direction of its movement, but just keeps alternating clockwise and counterclockwise rotations. In absence of chemical gradients the length of the straight line paths, i.e. the counter-clockwise rotations, is independent of the direction. The bacterium therefore essentially performs a random walk. In case of a positive gradient of attractants, like food, E. Coli instead reduces the tumbling frequency. In other terms, when the conditions are improving the bacterium proceeds in the same direction for a longer time. This simple strategy allows to bias the overall movement toward increasing concentrations of the attractant despite the simplicity of the mechanism and the difficulties in precisely sensing the gradient.

In this paper, we propose a bioinspired method that applies the same principle for robust mobile robot navigation. Actually this kind of behavior has already been applied for controlling a mobile robot [5,6]. Literature shows that while gradient descent is faster for tracking a single source, the biased random walk performs better in the presence of multiple and dissipative sources and noisy sensors and actuators. Demonstrations of the effectiveness of introducing a random term in the algorithm for preventing the robot from ending up in local minima are also provided.

However, the robustness to hardware damages and noisy sensory information achievable by biased random walk were not fully exploited. Expressly, in [6] the hardware already provides two basic movements, proceed straight and change direction randomly, and the biased random walk is performed at the behavior level. This approach limits the robustness for unexpected hardware failures. In fact, if due to hardware failures one of these basic movements does not operate as expected in many cases it will not be possible to accomplish the tasks. In our work, conversely, a biased random walk is executed directly in the *motor command space*, i.e. the behaviors themselves are determined online through the random walk. This gives great robustness in case of hardware failures since new behaviors that exploit the current hardware behavior are found online by biased random walk.

In general, performing a random walk in the motor command space allows to determine at runtime how to exploit the dynamics of the hardware, which can change due to hardware failures. With the approach presented in this work, therefore, there is no need to explicitly identify the failure and use preprogrammed alternative behaviors, which can be difficult to design beforehand [7]. We will show that the proposed bioinspired method is able to provide high adaptability despite the extreme simplicity of the algorithm and the absence of any additional hardware to cope with failures. We would like to stress that other, more task specific techniques could probably be designed to achieve higher performance for the navigation task, but in this work we are interested in showing the biologically inspired approach of biased random walks is sufficient to achieve the task and that it is robust to hardware failures and sensor noise.

The remainder of the paper is organized as follows: Section 2 will present the details of our control algorithm. Section 3 reports simulation results indicating that a biased random walk in the space of the actuator signals is able to drive the robot toward the target even without any knowledge of the robot structure (e.g. the orientation of the wheels). We then show that the robot is able to reach the target in presence of several hardware damages such as obscuration of part of the camera image or variation of the size, change of the rotation axle or uncontrollability of one wheel.

Another interesting point analyzed in this work is the effect of varying the noise and bias amplitudes. If the bias is too strong the system keeps repeating the same behavior even for small positive gradients, wasting time executing very small condition improvements. Conversely, if the noise is too strong the system changes its behavior very often trying many inefficient behaviors. We can therefore expect to have an optimal balance between noise and bias (see Fig. 1).

Results show that in our settings the performance of the behavior is determined essentially solely by the ratio of the two quantities, with an optimal ratio value dependent on the hardware state. We then verified by practical experiments with a real robot that our approach is applicable to real world problems with noisy sensors and actuators, as highlighted in Section 4. We conclude by section 5 illustrating future works.

2 Control Algorithm

As stated in the introduction our approach takes inspiration from the chemotaxis of E. Coli, which can be interpreted under the very general framework of biological fluctuations [8]. Expressly assuming to have a continuous time system we can model it by the equation

$$\dot{u} = \alpha A(x) f(u) + \beta \eta. \tag{1}$$

where $x \in \mathbb{R}^n$ is the state and $u \in \mathbb{R}^m$ is the control signal. The function $f : \mathbb{R}^m \rightarrow \mathbb{R}^m$ is a deterministic function and η is a random variable. These two quantities are multiplied by two constant scaling coefficients, $\alpha \in \mathbb{R}$ and $\beta \in \mathbb{R}$. The output values of f are multiplied by $A : \mathbb{R}^n \rightarrow \mathbb{R}$ which is a function of the state, called "activity", that indicates the fitness, or "quality" of a particular condition.

Intuitively, when the conditions improve, the value of $A(x)$ increases and control becomes mainly deterministic (approximately $f(u)$) whereas, when the conditions worsen, the control becomes more and more stochastic. As a practical

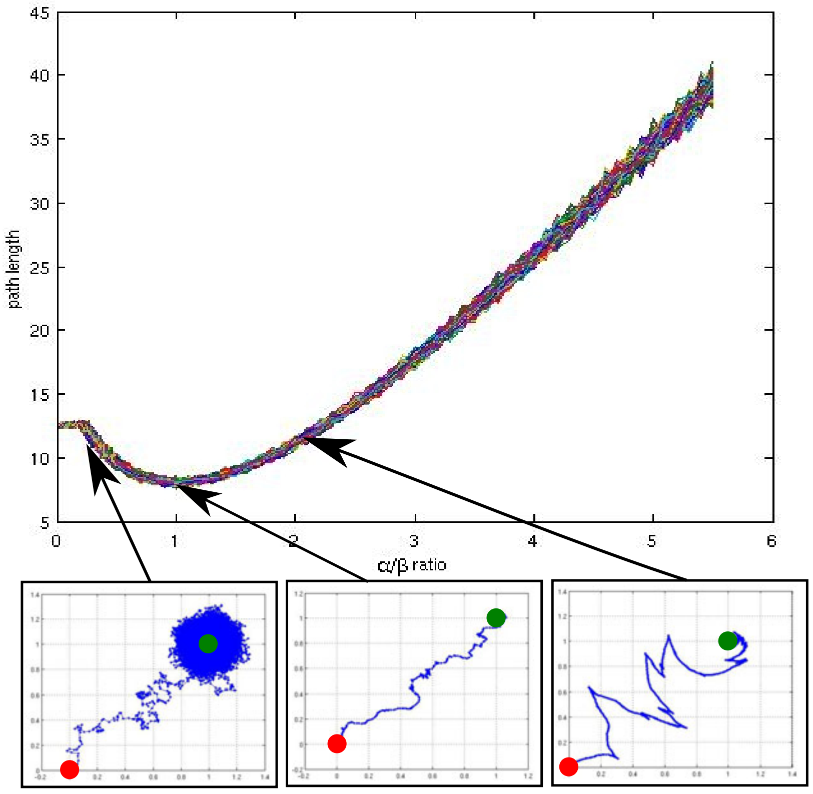

Fig. 1. Effect of the bias in random walk. The top figure plots the relationship between the bias to noise ratio and the path length, obtained by 100 tests. In the three graphs in the lower part of the figure the red circle denotes the start point and the green one denotes the target.

example in the E. Coli case, when the bacterium doesn't sense any food gradient it proceeds by a random walk, given by straight (deterministic) movements alternated by random changes in the direction due to tumbles. When the food concentration increases the frequency of tumbles is reduced and therefore the behavior becomes more and more deterministic.

This simple, biologically inspired framework revealed to be very robust and perform well in many applications of artificial systems, for instance for robot navigation, control of pneumatic actuated robotic arms or even routing in overlay networks [9].

While the robot presented in [6] has basic behaviors already implemented at a low level, we aim at having the system determining on-line also these motion primitives. A big advantage of this approach is that it can recover from unexpected and even quite serious damages in the robot hardware, giving the system a high robustness without the need of introducing any self modeling or damage identification.

In other terms, using the approach presented in [6] when the robot finds good conditions it increases the frequency of "deterministic" commands, i.e. "go forward". If due to a hardware failure, the behavior corresponding to such commands will be different from the expected one the task will not be accomplished. Imagine for instance to have a mobile robot with two wheels powered by independent motors and that due to an encoder problem one of the motors starts to rotate in the opposite direction. In this case when the "go forward" motor command is provided the robot spins around itself and the target will never be reached. With our approach, instead, the robot will explore new motor commands, until it finds that rotating the motors in opposite directions the distance from the target can be decreased.

In our experiments, the control signal u corresponds to the angular velocity of each motor and the state x corresponds to the information coming from the camera, i.e. the number of pixels whose color is similar to the target color.

To simplify as much as possible we decided to set $f(u)$ as follows:

$$f(u) = \frac{u}{\|u\|} \tag{2}$$

i.e. we maintain only the direction of u, and

$$A(x) = sgn(\frac{dx}{dt}) \tag{3}$$

where sgn is the sign function.

Concretely, we simply apply a bias in the current direction if we are following an increasing gradient, and bias in the opposite direction if the values of the activity is decreasing and no bias in case of a constant activity.

3 Robustness under Hardware Failures

Using ODE[1], we simulated a mobile robot equipped with three spherical wheels. The two front wheels are directly actuated by two independent motors whose maximum velocity is 0.5 rad/s while the rear wheel is free to rotate in any direction. The task is to reach a red hemisphere of radius 4 m placed at a distance of 30m. The robot is equipped with an omni-directional camera and the value of x fed to the controller is the number of red pixels in the image, determined by a filter that given the RGB components of each pixels counts the pixels whose R component is more than double the maximum of the G and B component values.

The controller receives information on the red pixels with a sampling frequency of 0.2Hz a value and provides a 2 dimensional velocity command u.

[1] Open Dynamics Engine, a free library for simulating rigid body dynamics. For details see `http://www.ode.org`

We chose to employ such a low sampling frequency to validate the robustness of the method even in case of low cost hardware with very poor performances. The controller implements the discrete time equivalent of equation 1, expressly

$$u_{t+1} = u_t + \alpha A(x)\frac{u_t}{\|u_t\|} + \beta\eta \tag{4}$$

where

$$A(x) = sgn(x_t - x_{t-1}) \tag{5}$$

We simulated four types of damages (see Figure 2):

1. the right wheel size is reduced to two thirds of its normal size
2. the right wheel becomes uncontrollable, i.e. its movement is completely random
3. the right wheel rotation axis direction is turned 90 degrees along the Z axis and becomes parallel to the longitudinal axis, i.e. the rotation of the wheel instead pushing the robot forward and backward pushes the robot to the left or to the right
4. 20% of the camera image becomes obscurated

We assumed $\eta \sim \mathcal{N}(0,1)$ as a Gaussian variable of variance one and studied the behavior for several values of α (the scaling coefficient of the bias) and β (the scaling coefficient of the noise). In particular for each condition (no damage or one of the damages listed) we determined the time spent contacting with the goal over 20000 seconds. Each condition is simulated one time for 128 different positions of the target, in particular assuming the robot's chassis is placed at (0,0) we set the target in each of the positions as $(R \cdot cos(\theta_i), R \cdot sin(\theta_i))$, $\theta_i = \frac{i \cdot 2\pi}{128}$, $i = \{0, \ldots, 127\}$ where $R = 30$ m.

The bias should be strong enough to drive the robot toward the target in short time but small enough not to keep performing the same action if the reduction of the distance to the goal is too little. We can expect therefore the existence of an optimal value of the bias. This intuitive idea can be easily exemplified simulating the movement of a point in the Cartesian plane. Expressly suppose a point to be initially at the origin and assume the presence of a target point, $(1,1)$ in Fig. 1. Define the velocity as

$$v_{t+1} = \alpha\frac{v_t}{\|v_t\|} * sgn(\Delta) + \beta\mathcal{N}(0,1) \tag{6}$$

where Δ is the variation of the distance to time t to time $t-1$. If the α/β ratio is too low, although the bias eventually drives the point in proximity of the target, much time is required to reach the goal because of unnecessary path deviations. Conversely, suppose to have a very high α/β ratio. Then, when the velocity becomes nearly tangential to the path that leads to the target, even though it is slightly in the direction of the target, the point keeps moving in that direction, which results in traveling long paths that only achieve a short reduction of the distance to the goal.

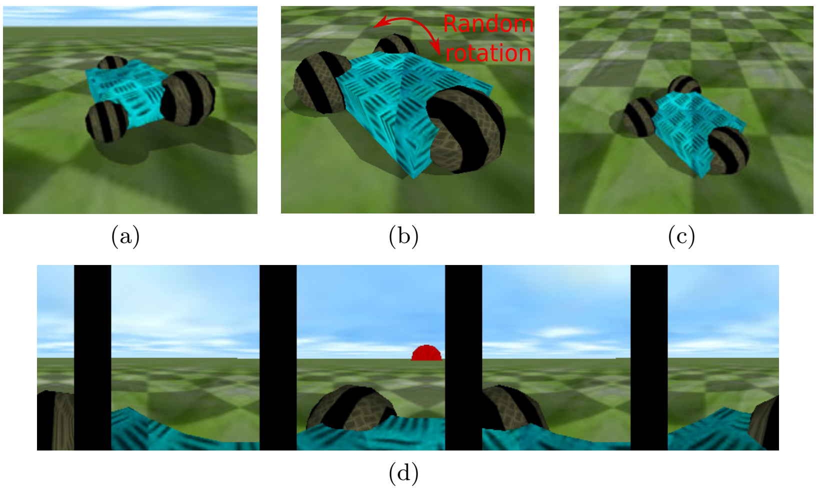

Fig. 2. The four damages simulated. (a) Reduced size of one wheel. (b) Uncontrollability of one wheel (c) Rotation of the rotation axis of one wheel (note the direction of the stripes) (d) Obscuration of part of the acquired image.

We can ask ourselves whether the optimal bias is constant or changes in case of hardware damages. The reaching experiment was therefore repeated for different settings of the values of α and β both for the undamaged robot and for the four robot conditions described beforehand.

Figure 3 depicts the results for the undamaged robot and for the four damages previously listed. The x and y axis indicate the values of α and β respectively, while the color represents the performance, in terms of ratio between the time spent touching the target and the total simulation time (20000 seconds). For all damages the graphs presents non zero values, i.e. the robot is able to reach the target and touch it. It is worth to notice that some damages, especially the rotation of the rotation axis of one wheel, completely changes the effect of motor commands. However, our simple algorithm is able to identify new efficient motor commands on the fly, without the need of any failure detection.

As expected, a completely deterministic behavior ($\beta = 0$) is often not able to drive the robot to the target, since, without "exploration" of motor commands done by the random part, the system can just provide a single type of motor command. Similarly when $\alpha = 0$ the probability of touching the goal with a completely random movement is so low that in no experiment the robot could reach the target within the simulation time.

We can see that the color zones are approximately triangles departing from the origin, i.e. the performance depends just on the ratio between α and β and not on their value.

Figure 3(f) shows the average performance for various $\frac{\alpha}{\beta}$ ratios. We notice that for the first type of damage (reduced wheel size) a ratio close to 2 gives the best performances, while in the case of changed rotation axis the best performance

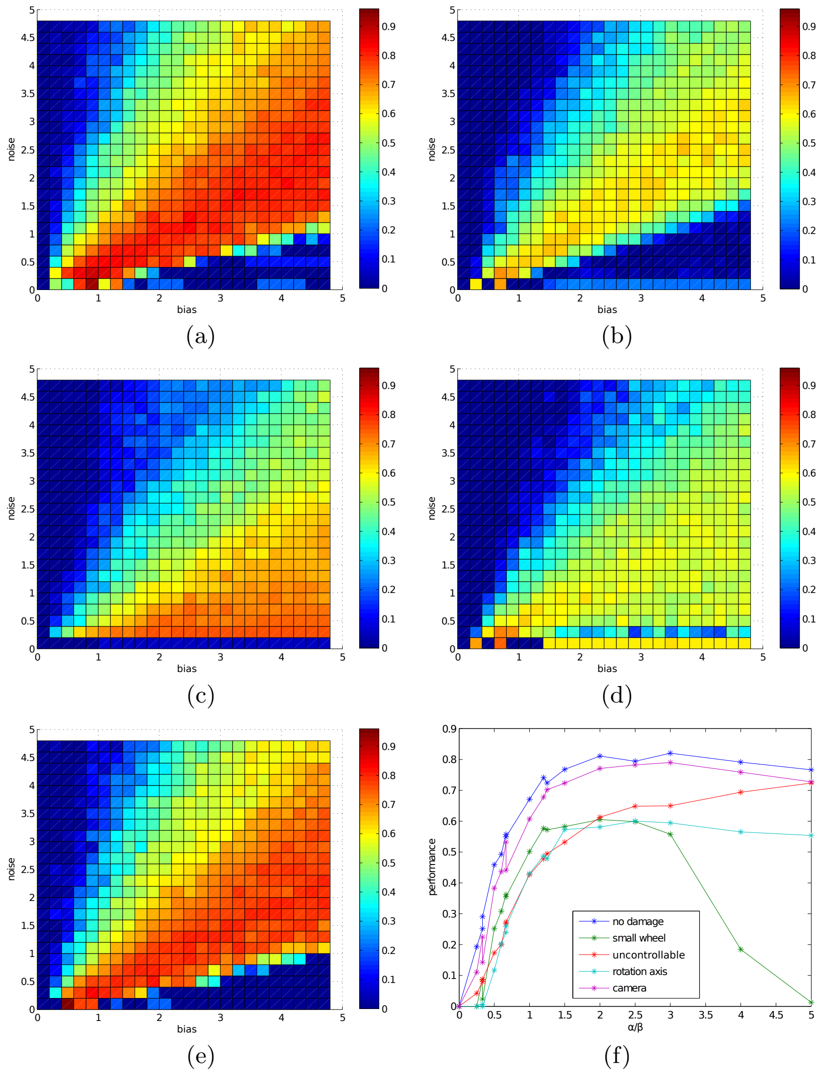

Fig. 3. Performances (ratio between the time spent touching the target and the total simulation time) of the robot for various settings of α and β: (a) Without any damage, (b) Reduced size of one wheel, (c) Uncontrollability of one wheel, (d) Rotation of the rotation axis of one wheel, (e) Obscuration of part of the image, (f) plot of α/β vs. average performance

is obtained with $\frac{\alpha}{\beta} \approx 2.5$. The undamaged robot and the robot with damaged camera instead performs best with $\frac{\alpha}{\beta} \approx 3$. For the uncontrollable wheel higher values for $\frac{\alpha}{\beta}$, around 5, gives the best performance. In this case, probably the noise introduced by the hardware itself reduces the noise required in the control signal.

Observing Fig. 3, we notice that some damages seems easier to recover than others, in detail the performance decreases more abruptly when the size of one wheel is reduced and when the rotation axis is changed by 90 degrees, than when the camera is partially obscurated or when one wheel become uncontrollable. In these cases a lower $\frac{\alpha}{\beta}$ is more beneficial, i.e. intuitively speaking when the task is difficult it appears that the more stochastic the control is the better it is.

4 Robustness to Real World Noise

Often the effect of noise, environment uncertainties and modeling errors prevents algorithms to work in a real world setup. We validated the practical applicability of our algorithm by using a mobile robot, namely B12 by Real World Interface. The robot was equipped with a 640×480 Logitech webcam placed on an omnidirectional mirror and an off-the-shelf mobile PC. A red blanket placed on a chair was used as target, as visible in Fig. 4

As in the simulation experiment the control loop consists in acquiring the image, counting the number of the red pixel, performing the biased random

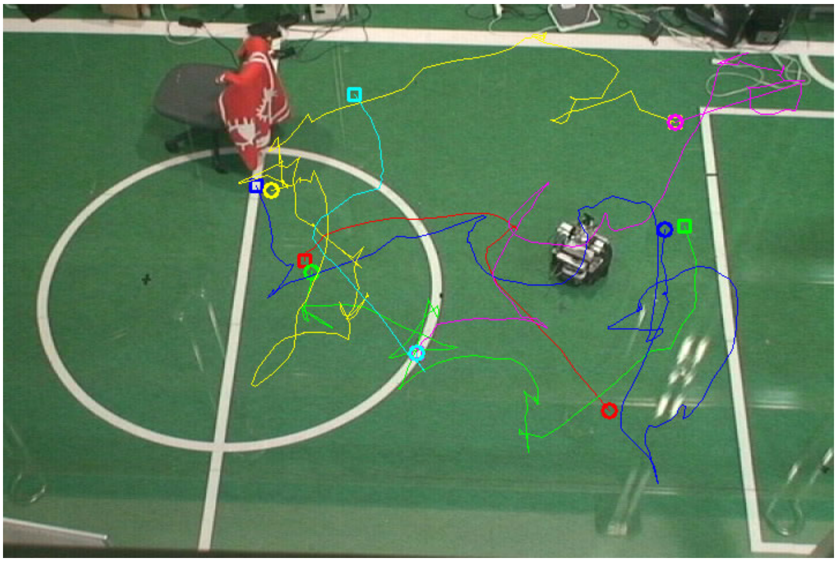

Fig. 4. Paths covered by the robot for six different initial positions and goal settings. Each path is drawn with a different color. Circles indicate starting points and squares indicate end points (for some subsequent paths the starting point is the ending point of the previous path.)

walk and setting the motor velocities. Once again to show the applicability of our approach even in case of very low-cost hardware, no particular hardware choice or software optimization were performed, leading to a quite slow and jittery control loop time, 861 ms with a standard deviation of 7.3 ms.

It is interesting to notice that the motor command space of the robot used in the experiment is quite different from the one of the simulated robot. Expressly, while the simulated robot has two driven-wheels independently actuated and a third rear castor wheel, the real robot is a Synchro Drive platform, in which one motor drives the wheels, while the other changes the wheel orientation. However, since our approach makes no assumption on the hardware, no change in the algorithm is required when passing from the simulation setup to the real robot experiment. The only parameter adjustment required was the α/β ratio. For the real world experiment, we adopted $\alpha = 0.8, \beta = 0.025$, determined experimentally with four trials to understand the order of magnitude.

The motors are equipped with encoders, which were used exclusively for data logging and reconstruction of the robot path. The short length of the paths covered by the robot in the experiments actually make the error accumulated by the encoders negligible. A validation was nonetheless performed using image processing on a video captured from the top of the experiment field.

Figure 4 shows some of the paths covered by the robots for different goal and robot initial position settings. Initial distance were set with values from 164 to 364 cm, and for all the 11 tests performed by the robot achieved target reaching,

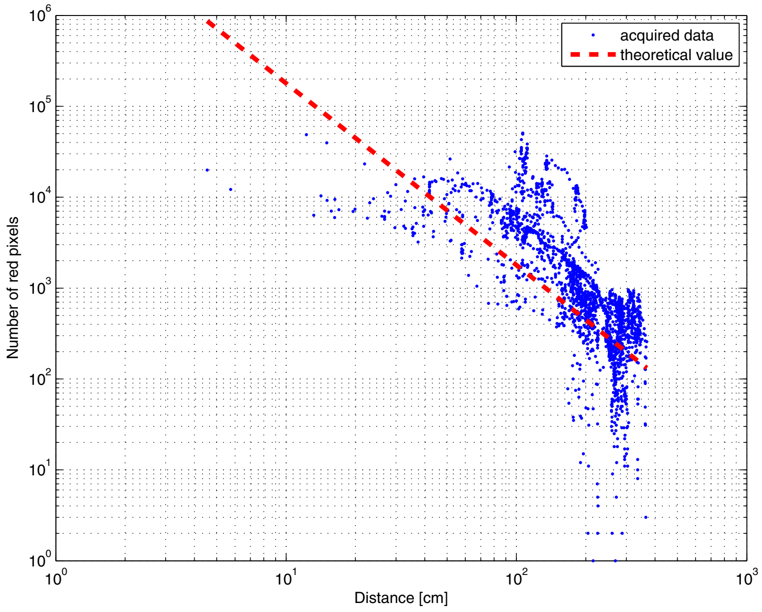

Fig. 5. Relationship between the distance from the goal and the number of red pixels measured during the experiments

with times ranging from 32 to 167 seconds. The measured path lengths were in average 3.38 times the optimal path. Notice that the performances could be improved by tuning the α/β ratio.

To illustrate the robustness to the real world noise Fig. 5 reports the relationship between the distance in a straight line to the goal, obtained off-line from the encoder values, and the number of red pixels measured during the robot motion. We can notice that, although the number of pixels is approximately a decreasing function of the distance, for similar distances we often have very different values for the number of pixel and conversely for a similar number of pixels the distances can be very different. Impressively, despite these strong uncertainties, in the actual target distance our simple algorithm was always able to drive the robot to the goal in a reasonably short time.

5 Conclusions and Future Works

In this paper, we presented an approach for mobile robot control based on biased random walk. The algorithm here proposed is actually biologically inspired, and in particular is based on the Escherichia Coli chemotaxis. These bacteria perform a random walk, but they bias their movement toward food by decreasing the number of random direction changes when the concentration of attractants increases.

We showed that biased random walk in the motor command space can be sufficient to drive a robot toward a goal without any knowledge of the robot structure. More importantly, since we make no assumptions on the actuators of the robot, in case of hardware failure the control is able to adapt to the new hardware conditions without the need of self modeling or failure identification.

We validated the methodology both by simulation and real world experiments. In detail, we showed that a mobile robot equipped with an omnidirectional camera is able to reach the goal even in case of severe damages to the sensors and actuators. The robot was hindered by damages similar to the ones which could occur during a Middle-Size RoboCup match. Our simple algorithm proved to be effective even in case of strong environmental noise, a big problem for all real world setups. We finally provided an experimental study on the optimal bias in the random walk, and showed that performances can be optimized by adjusting the ratio between α and β. The results suggest that the optimal bias is proportional to the random term with a coefficient dependent on the hardware. Initial tests seems to suggest that the bias should be higher in case of stronger damages, but this should be investigated more systematically with a clear definition of the "strength" or "hardness" of damages. Automatic estimation of the optimal bias will be tackled in future works. We notice, though, that even if the value is not optimal, the robot is still able to accomplish the task, i.e. choosing the optimal value of the ratio can increase the performances but suboptimal values are still sufficient to achieve the task. Future works will also aim at identifying whether other types of noise, like Levy walk, are more efficient than Gaussian Noise. The current experiments will finally be extended by the inclusion of fixed and moving obstacles to study the collision avoidance performance of our algorithm.

This paper solely aims at proving the feasibility and robustness of the very simple and biologically inspired approach of biased random walk in the motor command space. Obtaining high performance in terms of reaching time is out of scope. No performance comparisons were therefore performed with classical approaches like potential field navigation or reinforcement learning, for which we could actually expect better performances. However, we are able to notice that the underlying assumptions are different. In fact, when virtual force fields [10] approaches are used the results of the motor commands must be known a priori, while our approach does not assume any knowledge on the underlying actuators, as clearly appears by observing the difference in the structure of the simulated and real robot. Using reinforcement learning the ignorance on the motors can be overcome, but an appropriate and careful definition of states and rewards is required to allow successful learning.

References

1. Saiki, Y.M., Takeuchi, E., Tsubouchi, T.: Vehicle localization in outdoor woodland environments with sensor fault detection. In: ICRA, Pasadena, USA, pp. 449–454 (2008)
2. Plagemann, C., Fox, D., Burgard, W.: Efficient failure detection on mobile robots using particle filters with gaussian process proposals. In: IJCAI, Hyderabad, India, pp. 2185–2190 (2007)
3. Bongard, J., Zykov, V., Lipson, H.: Resilient machines through continuous self-modeling. Science 314(5802), 1118–1121 (2006)
4. Adler, J.: The sensing of chemicals by bacteria. Scientific American 234, 40–47 (1976)
5. Holland, O., Melhuish, C.: Some adatpive movements of animats with single symmetrical sensors (1996)
6. Dhariwal, A., Sukhatme, G.S., Requicha, A.A.G.: Bacterium-inspired robots for environmental monitoring. In: IEEE International Conference on Robotics and Automation (ICRA 2004), New Orleans, USA, pp. 1436–1443 (2004)
7. Scheutz, M., Kramer, J.: Reflection and reasoning mechanisms for failure detection and recovery in a distributed robotic architecture for complex robots. In: IEEE International Conference on Robotics and Automation (ICRA 2007), Roma, Italy, pp. 3699–3704 (2007)
8. Yanagida, T., Ueda, M., Murata, T., Esaki, S., Ishii, Y.: Brownian motion, fluctuation and life. Biosystems 88, 228–242 (2006)
9. Leibnitz, K., Wakamiya, N., Murata, M.: Biologically inspired self-adaptive multi-path routing in overlay networks. Communications of the ACM 49, 63–67 (2006)
10. Arkin, R.: Motor schema based navigation for a mobile robot: An approach to programming by behavior. In: IEEE International Conference on Robotics and Automation (ICRA 1987), Raleigh, USA, pp. 264–271 (1987)