Bioinspiration & Biomimetics

RECEIVED
3 December 2014

REVISED
20 August 2015

ACCEPTED FOR PUBLICATION
25 August 2015

PUBLISHED
8 October 2015

PAPER

Imparting protean behavior to mobile robots accomplishing patrolling tasks in the presence of adversaries

Daniel-Ioan Curiac[1] **and Constantin Volosencu**

Automation and Applied Informatics Department Politehnica University of Timisoara Bd. V. Parvan nr. 2, 300223 Timisoara, Romania
[1] Author to whom any correspondence should be addressed.

E-mail: daniel.curiac@aut.upt.ro **and** constantin.volosencu@aut.upt.ro

Keywords: protean behavior, mobile robot, adversarial patrolling, path's unpredictability, fictive-temporary obstacle, Arnold's cat map

Abstract

Providing unpredictable trajectories for patrol robots is essential when coping with adversaries. In order to solve this problem we developed an effective approach based on the known protean behavior of individual prey animals—random zig-zag movement. The proposed bio-inspired method modifies the normal robot's path by incorporating sudden and irregular direction changes without jeopardizing the robot's mission. Such a tactic is aimed to confuse the enemy (e.g. a sniper), offering less time to acquire and retain sight alignment and sight picture. This idea is implemented by simulating a series of fictive-temporary obstacles that will randomly appear in the robot's field of view, deceiving the obstacle avoiding mechanism to react. The new general methodology is particularized by using the Arnold's cat map to obtain the timely random appearance and disappearance of the fictive obstacles. The viability of the proposed method is confirmed through an extensive simulation case study.

1. Introduction

In the living world, the concept of protean behavior, as defined by Humphries and Driver [1], covers any unsystematic action that prevents prediction. Such a behavior is the random zig-zag movement of a prey animal aimed to confuse, to delay or to decrease the efficiency of the predator's reactions and is used by a plethora of species, including spiny mice [2], whirligig beetles [3], Thomson's gazelles [4], grasshoppers [5], etc. Individual protean fleeing, considers the unsystematic zig-zag movements either as a mean to prevent an attack or as an emergency response to a predator. In the domain of mobile robots two methods reported the use of protean behavior, both of them as an emergency reaction to an enemy [6, 7]. Conversely, the proposed method is, to the best of our knowledge, the first that uses the protean zig-zagging as a prevention stratagem by transforming the robot's path into a misleading one without endangering the accomplishment of robot's mission. Our approach is designed to confuse the enemy by offering less time to acquire and retain sight alignment and sight picture and is particularly suitable for mobile robots performing patrolling tasks in military theaters of operation.

In the area of mobile robots, coping with adversaries was traditionally addressed from two basic perspectives: game theory and chaotization of the robot's path.

The game theory approaches use three types of models to describe a patrolling mission (the models related to the environment, to the robot itself and to the adversaries) and tackle this problem by converting it to typical pursuit-evasion games. Therefore, relevant methods have been developed for diverse patrolling settings including single [8, 9] or multi-robot patrolling [10, 11], when considering full or partial knowledge [12–14] about opponents. Despite their solid theoretical basis, due to simplifying assumptions and computational complexity, the game-theoretic approaches are unlikely to be used for the time being in complex real-life applications involving autonomous robots in dynamically unknown environments.

While the methods based on game theory mainly refer to patrolling mobile robots that preserve an environment from intrusions, the chaotization of robot's path approaches cover a larger set of patrolling tasks classified by the type of objective under surveillance: area [15–19], perimeter [20] and points of interest [21, 22]. The focal point of these methods lies in the

sensitivity to the initial conditions feature of chaotic systems, also known as 'the butterfly effect', which states that an opponent cannot predict the future position of a robot moving on a chaotic path. These methods involving the chaotic dynamics, are very much task-related due to the specific chaotic system used in the path's chaotization. As a result, these methods are not appropriate for autonomous robots performing more than one type of task.

The method proposed in this paper originates from the idea to transpose a known protean stratagem—the randomly zig-zagging of individual prey animal under threat—into a path planning algorithm. The implementation gravitates around our concept of fictive-temporary obstacle (FTO). Such randomly-generated obstacles are simulated to intentionally mislead the obstacle avoiding mechanism to react. Even though this novel method is similar to the chaos-based approaches in the sense that it randomizes the robot's path, it confers a certain advantage: it is not task-related. Moreover, the novel method can be efficiently implemented on autonomous robots, needing minimal on-board available resources.

The rest of the paper is structured as follows. Section 2 is devoted to the problem formulation, providing also a preparatory analysis for its future solving. Section 3 presents our concept of FTOs stating their role inside our method. In section 4 we describe the method, giving insights on how fictive obstacles can be efficiently generated using 2D Arnold's cat map. Section 5 offers an extensive simulation case study that validates the performances of the proposed method in workspaces with or without real obstacles (ROs). Finally, in section 6, conclusions and final remarks are reported.

2. Problem statement

The proposed bio-inspired method for imparting unpredictability to a robot's trajectory relies on the intrinsic component of every mobile robot path planning system: the obstacle avoidance algorithm. Based on this component, we developed a strategy that forces the robot to overtake a sequence of fictive and temporary obstacles, simulating the zig-zag movements of individual prey animals under threat. By this, the robot's trajectory is altered into a misleading path for enemy entities.

Problem formulation: Consider a mobile robot endowed with onboard computation and sensing capabilities and an infinite or bounded two-dimensional workspace $W \subset \mathbb{R}^2$ in which this robot will evolve accomplishing its predefined task. This workspace contains a set of n real and fixed obstacles O_i, with $i = 1..n$. The task is to alter the optimized trajectory to cope with enemy entities, keeping the task accomplishment goal and using only onboard capabilities.

In our view, this problem can be solved using a low-computational algorithm that can simulate fictive and temporary obstacles which suddenly appear in the workspace. By avoiding these newly introduced obstacles, the path will become unpredictable for external entities. Therefore, we will generate an infinite time ordered sequence of FTOs F_j with $j = 1, 2,\ldots$ that will unexpectedly come into the robot's sight.

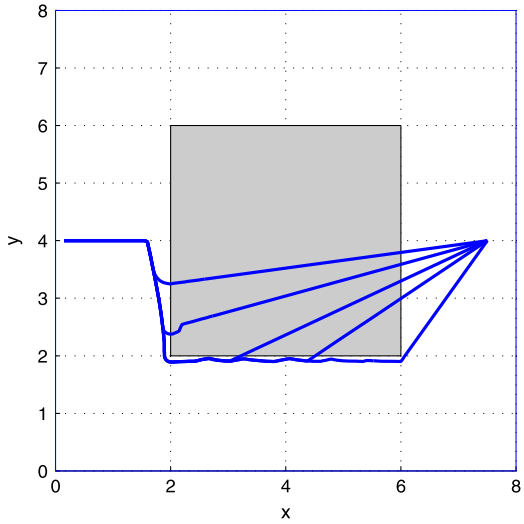

Figure 1. Lifetime duration of the FTO is affecting the path.

Due to the ephemeral character of newly introduced fictive obstacles, the approach can be naturally implemented in a synergy with sensor-based path planning mechanisms, suitable for unknown environments (e.g. bug-like algorithms [23] including Bug0, DistBug or TangentBug). If the mobile robot possess sufficient computational onboard resources to process temporary maps, the method can be used in conjunction with global path planning mechanisms [24] (e.g. Breadth-first search, Dijkstra's algorithm, A∗, Potential Field, Cell Decomposition), too.

In order to exemplify the influence of fictive obstacles upon the robot's path, throughout this paper we will consider a simple and efficient obstacle avoiding algorithm for unknown environments—TangentBug [25, 26]. This algorithm uses information provided by a ranging sensor to find the shortest way to the goal and, like other bug algorithms [23] require two types of behavior: motion-to-goal when the robot moves on a straight line until the goal is reached or an obstacle is encountered; and boundary-following when the robot circumnavigates an obstacle.

3. Fictive-temporary obstacles

The obstacles introduced by our biomimetic method have two characteristics: *fictive* in the sense that they are simulated for cheating the obstacle avoiding mechanism to react even though they are not present in the real environment, and *ephemeral* in the sense that they are active in the robot's 'mind' only for a short-term period. Besides the fact that the obstacles' appearances cannot be predicted by external entities,

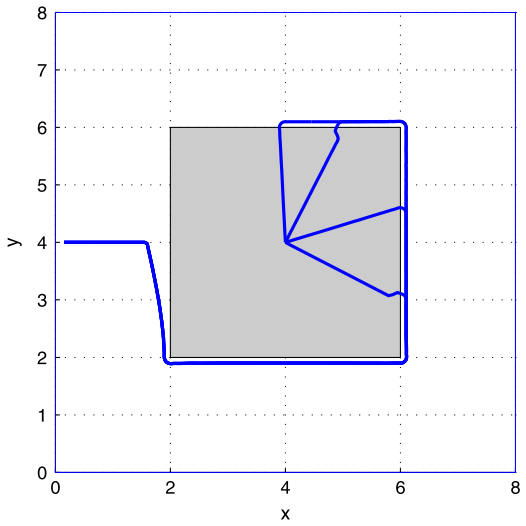

Figure 2. FTO is temporarily covering a waypoint.

the unpredictability is further enhanced by their ephemerality that brings a set of advantages to the generality of our method:

(a) Once appeared in a place, a fictive obstacle will not be encountered again if the robot will return to the same location;

(b) The shape of the robot's path is affected by the time fictive obstacle is active; Each fictive obstacle has a limited life period denoted by t_a. For obstacles having exactly the same shape and dimension, this limited period when they are active in the workspace may generate different paths. For example we considered that the robot's path starts from the point having the coordinates (0.1, 4) and is heading toward the goal position (7.4, 4). If a square fictive obstacle with vertices at (2, 2), (2, 6), (6, 2) and (6, 6) appears in front of the mobile robot (figure 1), the path will be affected by the life period t_a as follows: for $t_a < t_\varepsilon$ the robot didn't pass the entire obstacle when the obstacle disappears, while for $t_a \geqslant t_\varepsilon$ (most right path) the robot will pass the entire obstacle.

(c) The fictive obstacle can appear in any location of the robot's workspace, even to cover the waypoint that needs to be touched (figure 2) or to be superposed totally or partially with a RO (figure 3).

Figure 2 presents the situation in which the robot, starting from the location with coordinates (0.1, 4.0) and heading towards the goal point (4, 4), encounters a fictive obstacle (a square with vertices at (2, 2), (2, 6), (6, 2) and (6, 6)) that covers the position of the goal. Due to the fact that the obstacle is temporary, the obstacle will be circumnavigated by the robot until it disappears and after that, when the direct trajectory to the goal will be available, the robot will head towards the waypoint. Therefore, when the goal point is temporarily covered by a fictive obstacle, the path's shape is influenced by the time moment when this obstacle stops being active (figure 2).

As already mentioned, the fictive obstacle may appear in front of the robot anywhere in the workspace, sometimes being superposed totally (has no effect on the robot's path) or partially with a RO. This last situation is exemplified in figure 3, where the robot starting from (1.5, 0.5) is heading toward the goal point (1.5, 7). If the only obstacle encountered is the real one (the triangular obstacle) the path will follow the dashed line, while in the case that a fictive square obstacle is generated, it has to be circumnavigated, so the solid line will be pursued. In this case the robot has to pass a compound obstacle (CO) represented by the union of the two shapes:

$$\mathrm{CO} = O_i \bigcup F_j, \tag{1}$$

where O_i is a RO and F_j is a FTO.

4. Proposed methodology

Our methodology uses the obstacle avoidance mechanism to generate unpredictable paths for mobile robots. This mechanism is triggered either when a RO is encountered or by an algorithm that simulates a fictive obstacle. Figure 4 presents the block diagram for implementing the proposed method. Besides the two modules included in every mobile robot structure (obstacle sensing module and obstacle avoiding mechanism), our method introduces two other modules: a FTO generator and an interface block.

The FTO generator is the core part of our strategy, having the role to provide a simulated signal that forces the interaction between mobile robot and a set of carefully constructed fictive obstacles. This module will be thoroughly described and analyzed in the second part of the section.

The interface block combines the output signals provided by FTO generator and obstacle sensing modules in order to activate or deactivate the obstacle avoiding mechanism. From the functional point of view this module implements, in a broad-sense, a logic-OR operation forcing the obstacle avoiding mechanism to react when either a real or a FTO is encountered.

4.1. Generating fictive obstacles

The new module designed to simulate the fictive obstacles must effectively interrelate with other parts of the robotic system, especially with the obstacle avoidance mechanism. In order to describe a fictive obstacle we will start from a reasonable assumption that the obstacle avoiding mechanism reacts when the RO is in the robot's field of view within a given range $[d_{\min}, d_{\max}]$, and stops reacting when the obstacle leaves this zone. Accordingly, we will generate fictive obstacles characterized by two parameters (figure 5):

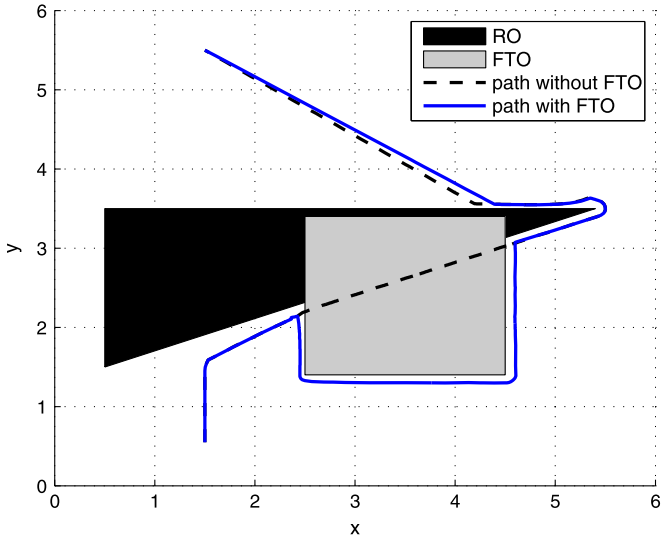

Figure 3. Partial superposition of a real and a fictive obstacles.

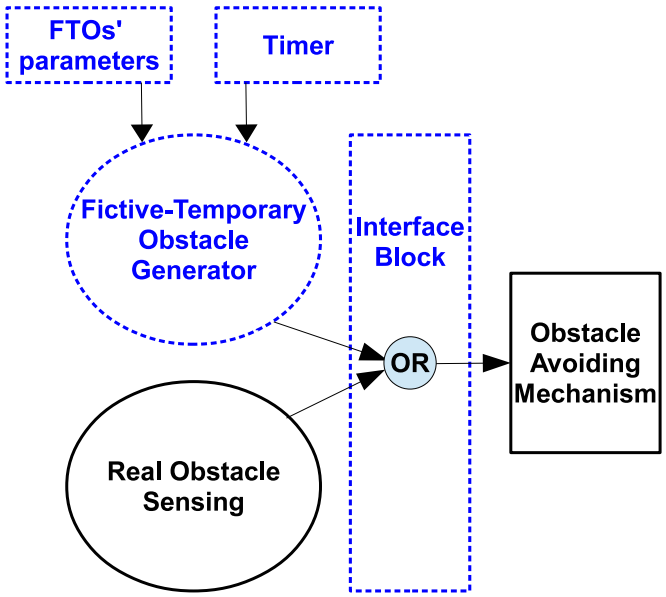

Figure 4. Functional block diagram of the proposed method.

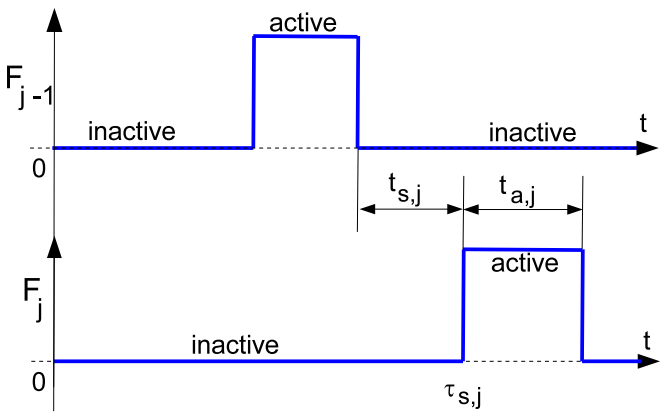

Figure 5. Time diagram of succession of FTOs.

(a) The moment in time when the fictive obstacle F_j appears in the field of view (appearance-time)—it is described by the parameter $t_{s,j}$ which represents the time interval from the moment the previous fictive obstacle F_{j-1} became inactive to the moment F_j becomes active;

(b) The time duration (active-time) $t_{a,j}$ when a fictive obstacle F_j is active inside the field of view;

As can be seen, the geometry of the obstacle is not essential for the proposed method, this attribute being eclipsed by the two mentioned parameters: the moments in time when the obstacle avoidance mechanism is activated or deactivated. Moreover, an explicit description of the FTO, which includes two spatially related components (location and geometry of the obstacle) and two temporal-related components (the moments in time when the obstacle appears and disappears from the workspace) is inappropriate because it forces the robot to memorize and process temporary maps of the environment, which requires high computational on-board power.

If we want to give a spatial description of the FTOs we generate, this description will be very much related with the robot's sensor type and the obstacle avoidance mechanism. For example if the robot, moving with the average speed v, uses a ranging sensor and the TangentBug obstacle avoidance algorithm, a fictive obstacle F_j may be envisioned as a very-thin rectangle with its center located on the direction of movement, and with the length being computed as $l_j = 2 \cdot v \cdot t_{a,j}$ to ensure that it may be overtaken after $t_{a,j}$ if the robot tries going either to the left or to the right.

Knowing the active-times and appearance-times for previously generated fictive obstacles, we can compute the appearance-time of each obstacle F_j relative to the robot's starting moment using the following formula:

$$\tau_{s,j} = t_{s,j} + \sum_{k=1}^{j-1} \left(t_{s,k} + t_{a,k} \right). \tag{2}$$

At each moment in time the workspace will be populated with n ROs and not more than one fictive obstacle. Thus, the total number $N(t)$ of obstacles at a given moment in time t will be:

$$N(t) = \begin{cases} n & \text{if } t \in \left[0, \tau_{s,1}\right) \cup \\ & \left(\tau_{s,1} + t_{a,1}, \tau_{s,2}\right) \cup \ldots \cup \\ & \left(\tau_{s,j} + t_{a,j}, \tau_{s,j+1}\right) \cup \ldots \\ n+1 & \text{otherwise} \end{cases} \tag{3}$$

Our fictive obstacles are generated in such a manner to appear random for external observers. Thus, we have to generate two sequences of random-looking variables ($t_{s,i} \in (t_{s\,\min}, t_{s\,\max})$ and $t_{a,i} \in (t_{a\,\min}, t_{a\,\max})$). In practice two ways can be followed to generate such sequences: to use a random number generator or a chaotic dynamic system.

Random numbers can be generated on numerical devices using two different approaches [27]:

(a) True random number generators which capture randomness from physical phenomena (e.g. atmospheric noise) and introduce it in the computer program; and

(b) Pseudo-random number generators , where the random feature characterized by non-determinism and aperiodicity is totally lost, providing instead deterministic and periodic sequences that have a certain distribution function (uniform or normal distributions are typically used).

Chaotic systems are by definition deterministic systems [28], where the random-looking sequences are obtained using mathematical models. In this case the source of unpredictability is considered to be their 'sensitivity to initial conditions' feature, which states that tiny differences in initial conditions can lead to huge differences between corresponding future states of the system.

There are two requirements that we imposed to bring a higher level of efficiency to our method. First of all we want the mechanism to be deterministic. This feature will ensure that ally entities (e.g. other robots constituting a team or the superior level observer), having full knowledge about the robot, can recompute the robot's path and use this information in their decision making process. On the other side we need a mechanism which can be efficiently implemented on resource-constrained devices (robot has to preserve its long-term autonomy) and moreover, that infinitesimal errors in knowing the parameters to lead to huge errors in predicting the robot's location. Combining the two criteria we decided to implement our method using chaotic system.

Our fictive obstacles are generated in such a manner to appear random for external observers using chaotic dynamics. Thus, we have to generate two sequences of random-looking variables ($t_{s,i} \in (t_{s\,\min}, t_{s\,\max})$ and $t_{a,i} \in (t_{a\,\min}, t_{a\,\max})$). The use of chaotic dynamics to generate the time parameters for the FTOs offers a vast range of options. In the following paragraphs we propose an efficient solution based on the dynamics of a hyperbolic toral automorphism—2D Arnold cat map [29]. This simple discrete map is given by the following transformation:

$$\begin{bmatrix} x_j \\ y_j \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x_{j-1} \\ y_{j-1} \end{bmatrix} \bmod 1, \tag{4}$$

where the use of modulo-1 operator (mod1) restricts the two state variables x_j and y_j to the interval [0, 1). The map (4) has an evident chaotic behavior in $\mathbb{R}^2 \setminus \mathbb{Z}^2$ [30], the Lyapunov exponents being $\lambda_1 = \log\left(0.5 \cdot \left(3 + \sqrt{5}\right)\right) > 0$ and $\lambda_2 = \log\left(0.5 \cdot \left(3 - \sqrt{5}\right)\right) < 0$. Since the determinant of its linear part is 1, the map (4) is invertible and area-preserving [31].

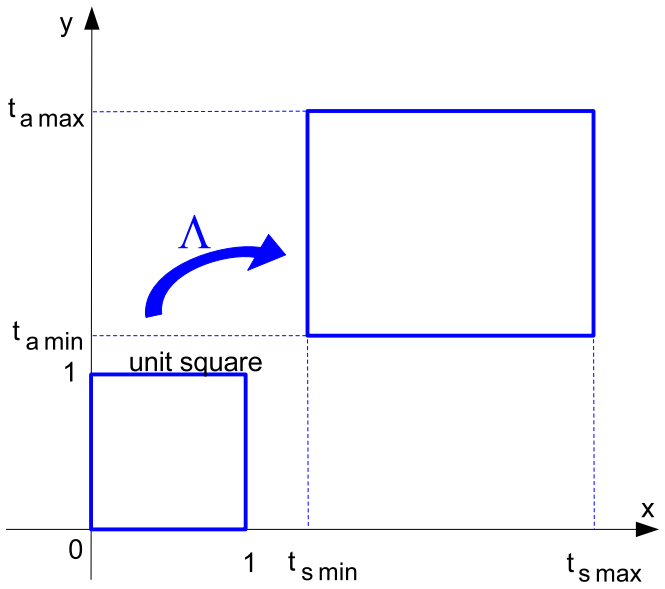

Figure 6. Affine transformation Λ from variables x_j, y_j to $t_{s,j}$ and $t_{a,j}$.

Starting with non-integer initial values for the state variables, we can generate two independent pseudo-random sequences x_j and y_j inside the interval (0, 1). To obtain the needed sequences $t_{s,j}$, $t_{a,j}$ that characterize the series of fictive obstacles F_j, we have to adapt each of the two pseudo-random variables x_j and y_j to fit into ($t_{s\ \min}$, $t_{s\ \max}$) and ($t_{a\ \min}$, $t_{a\ \max}$), respectively (figure 6). This is done using a combined affine transformation (scaling and translation) Λ, described by:

$$\begin{cases} t_{s,j} = \left(t_{s\ \max} - t_{s\ \min}\right) \cdot x_j + t_{s\ \min}, \\ t_{a,j} = \left(t_{a\ \max} - t_{a\ \min}\right) \cdot y_j + t_{a\ \min}. \end{cases} \quad (5)$$

Thus, at each iteration of the Arnold map (4), we will obtain the time-parameters $t_{s,j}$ and $t_{a,j}$ for a new FTO using (5).

For an efficient onboard implementation, four values must be memorized by the autonomous robot before starting its mission: $t_{s\ \min}$, $t_{a\ \min}$, ($t_{s\ \max} - t_{s\ \min}$) and ($t_{a\ \max} - t_{a\ \min}$). In this case the number of necessary operations is extremely low, only four additions and three multiplications being required to simulate the FTOs based on formulas (4) and (5).

4.2. Methodology implementation

Using the fictive obstacle generating method presented above, we can configure the robot motion algorithm as presented in the following pseudocode:

```
start moving;
while(1)
{
    WP = generate_new_waypoint();
    while (waypoint is not reached)
    {
        move_toward_waypoint(WP);
        (t_s, t_a) = generate_fictive_obstacle():
        start_timer();
        while(1)
        {
        if ((t_s< = timer< = t_a AND fictive_obstacle is still in the way) OR (real_obstacle is still in the way)
            obstacle_avoidance();
        if(timer > t_s + t_a)
            break;
        }
    }
}
```

Our algorithm considers a general case in which the robot's path is described by a sequence of waypoints computed by the *generate_new_waypoint*() function. The fictive obstacles, specified by the two mentioned time parameters (starting time t_s and active time t_a), are generated using *generate_fictive_obstacle*() when the robot starts moving or when the previous fictive obstacle becomes inactive. The robot is moving in a straight line toward the waypoint (function *move_toward_waypoint*(*WP*)) until it encounters either a RO or an active fictive obstacle. In the circumstance that an obstacle is blocking the line to the waypoint, the obstacle avoidance algorithm (*function obstacle_avoidance*()) is triggered. Immediately after the obstacle is passed, the robot rejoins the linear path towards the waypoint.

The method is exemplified by the simulation case study depicted in the following section.

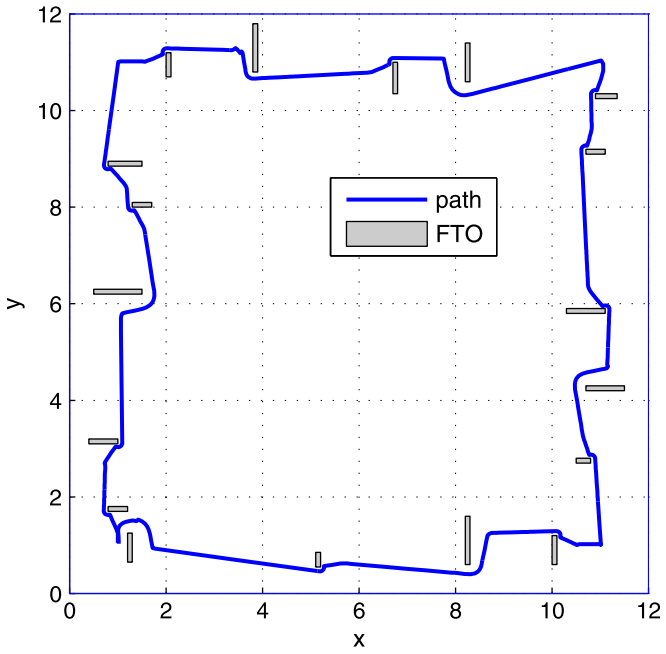

Figure 7. Impact of FTOs above robot's path (1 lap).

5. Case study

In order to assess the effectiveness of the proposed method an extensive simulation case study has been performed. For this, we configured a 144 square meters (12 m × 12 m) workspace where a mobile robot, equipped with a distance sensor having a 0.5 m range, is moving with a constant speed of 0.25 m s^{-1}. The robot pursues a square contour described by four waypoints placed in the vertices, having the following coordinates: (1, 1), (1, 11), (11, 11) and (11, 1). We also considered that the robot is using a suitable bug-like obstacle avoiding mechanism—TangentBug [25]. The first scenario presents a workspace free of ROs, while the second one considers some ROs that must be avoided.

5.1. Workspace without ROs

In adversarial conditions, pursuing the direct line between two successive waypoints offers the adversary enough time to attain and keep the sight alignment, practically endangering the robot's task accomplishment. If the trajectory includes sudden and random direction-changes, as the ones induced by our FTOs, opponent's mission becomes much more difficult.

Figure 7 presents the robot's path and the set of FTOs encountered by the robot on its first lap. These FTOs, used by the proposed method to impart unpredictability to robot's trajectory, were generated using the formulas (4) and (5) with initial values $x_0 = 0.18$ and $y_0 = 0.35$ and considering the appearance time (t_s) and the active time (t_a) being in the ranges 2.0–20.0 and 0.3–2.5 s, respectively.

When engaged in a patrol mission along the chosen square boundary, the robot will accomplish different paths on each loop. The chosen square contour (dotted line) and the trajectory of the robot for three consecutive laps are presented in figure 8.

In order to estimate the trajectory's unpredictability inserted by the method, we define two basic metrics that cover the temporal and spatial aspects:

(a) The mean time between FTO's, denoted by MTBFTO, measures the average time to a new sudden direction-change. In our case, when the two temporal variables are generated using Arnold's cat map which produces uniform pseudorandom sequences [32], this metric can be computed using the following formula:

$$\mathrm{MTBFTO} = \frac{\left(t_{s\,\max} - t_{s\,\min}\right) + \left(t_{a\,\max} - t_{a\,\min}\right)}{2}. \tag{6}$$

(b) The root mean square deviation (RMSD) computed in a given point $P(x_P, y_P)$ of the normal path (path without FTOs), characterizes the spreading of the path's laps in that specific point and is described by the formula:

$$\mathrm{RMSD}(P) = \sqrt{\frac{1}{n}\sum_{i=1}^{n} d\left(P_i, P\right)^2}, \tag{7}$$

where $d(P_i, P)$ is the distance between the point of a path's lap and the point P, in cross-section. We can choose any arbitrary point on the square contour described by the four waypoints to calculate this metric, except the waypoints where RMSD(P) will be

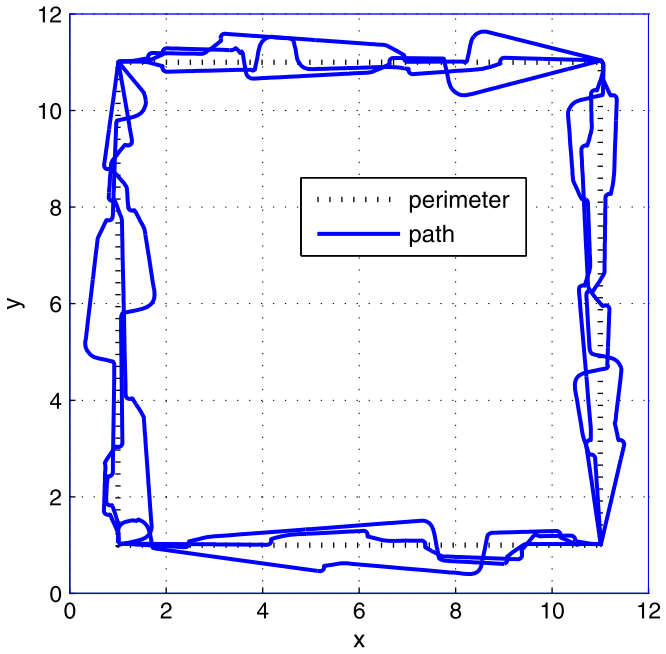

Figure 8. Impact of FTOs above robot's path (3 laps).

Table 1. Chosen temporal FTOs' parameters and corresponding metrics' values.

Case no.	$t_{s\,\min}$ (s)	$t_{s\,\max}$ (s)	$t_{a\,\min}$ (s)	$t_{a\,\max}$ (s)	MTBFTO (s)	RMSD(P(4, 1)) (m)
1	2	20	0.3	2.5	10.1	0.65
2	0	15	0.3	2.5	8.6	0.69
3	0	10	0.3	2.5	6.1	0.72
4	0	7	0.3	2.5	4.6	0.74
5	2	20	0.3	5	11.35	0.81
6	0	15	0.3	5	9.85	0.85
7	0	10	0.3	5	7.35	0.88
8	0	7	0.3	5	5.85	0.90

equal to zero (at each lap, the path must touch the waypoints).

If we choose the boundary point P, in which the metric will be estimated, along one of the two horizontal edges of the contour (the cross-section being characterized by $x_i = x_p$, for every index i), then (7) can be rewritten in a simplified form as:

$$\mathrm{RMSD}(P) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - y_p\right)^2}. \tag{8}$$

Table 1, presents these two metrics for different ranges of appearance time ($t_s \in (t_{s\,\max} - t_{s\,\min})$) and active time ($t_a \in (t_{a\,\max} - t_{a\,\min})$), RMSD($P$) being computed based on 40 consecutive laps in the point P having the coordinates $x_P = 4$ and $y_P = 1$.

5.2. Workspace with ROs

If the workspace contains ROs, the robot's path can also be transformed into a misleading trajectory using our proposed method. In this situation, the obstacle avoidance algorithm is activated either by FTOs or ROs, a common situation being the one already presented in section 3 (figure 3) where a FTO can be superposed, totally or partially, with a RO. Figure 9 describes such a path showing the encountered FTOs, too. To perceive the trajectory's unpredictability, in figure 10 we present three consecutive laps of the mobile robot operating in this environment.

6. Conclusions

The paper described a novel bio-inspired solution to obtain misleading and unpredictable paths for mobile robots evolving in environments with adversaries while preserving their predefined tasks. The kernel of our method lies in the new concept of FTOs that randomly appear in the robot's field of view. Such simulated obstacles can be generated in an efficient manner using the Arnold's cat map with onboard existing resources making the method feasible for autonomous robots. By simulating an infinite series of FTOs the robot's trajectory will be altered, the path seen by an adversary being similar to the one of an individual pray animal under threat.

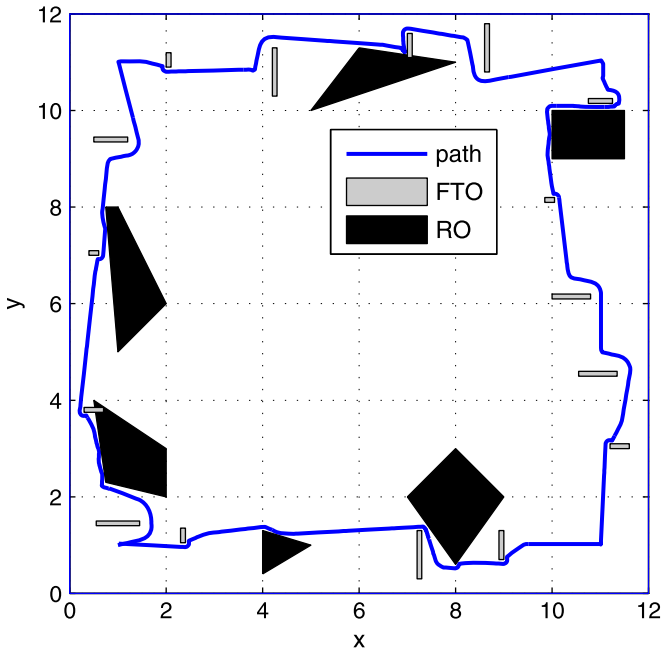

Figure 9. Impact of FTOs above robot's path in a workspace with real obstacles (1 lap).

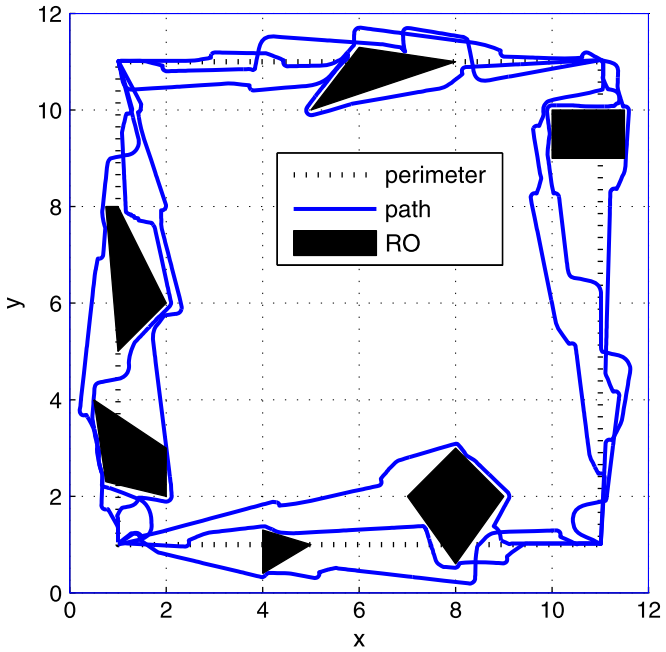

Figure 10. Impact of FTOs above robot's path in a workspace with real obstacles (3 laps).

References

[1] Humphries D A and Driver P M 1970 Protean defence by prey animals *Oecologia* **5** 285–302

[2] Shahaf E and Eilam D 2004 Protean behavior under barn-owl attack: voles alternate between freezing and fleeing and spiny mice flee in alternating patterns *Behav. Brain Res.* **155** 207–16

[3] Vulinec K and Miller M C 1989 Aggregation and predator avoidance in whirligig beetles (Coleoptera: Gyrinidae) *J. New York Entomological Soc.* **97** 438–47

[4] Fitzgibbon C D 1990 Anti-predator strategies of immature Thomson's gazelles: hiding and the prone response *Anim. Behav.* **40** 846–55

[5] Bateman P W and Fleming P A 2014 Switching to Plan B: changes in the escape tactics of two grasshopper species (Acrididae: Orthoptera) in response to repeated predatory approaches *Behav. Ecol. Sociobiol.* **68** 457–65

[6] Araiza-Illan D and Dodd T J 2012 Bio-inspired autonomous navigation and escape from pursuers with potential functions *Advances in Autonomous Robotics* (Berlin: Springer) pp 84–95

[7] Araiza-Illan D and Dodd T J 2010 Biologically inspired controller for the autonomous navigation of a mobile robot in an evasion task *World Acad. Sci., Eng. Technol.* **4** 1344–9
[8] Basilico N, Gatti N and Amigoni F 2012 Patrolling security games: definition and algorithms for solving large instances with single patroller and single intruder *Artif. Intell.* **184** 78–123
[9] Basilico N, Gatti N and Amigoni F 2009 Leader-follower strategies for robotic patrolling in environments with arbitrary topologies *Proc. 8th Int. Conf. on Autonomous Agents and Multiagent Systems* (*Budapest, Hungary*) pp 57–64
[10] LaValle S M and Hutchinson S A 1993 Game theory as a unifying structure for a variety of robot tasks *Proc. 1993 IEEE Int. Symp. on Intelligent Control* (*Chicago, IL, USA*) pp 429–34
[11] Hernández E, Barrientos A and del Cerro J 2014 Selective smooth fictitious play: an approach based on game theory for patrolling infrastructures with a multi-robot system *Expert Syst. Appl.* **41** 2897–913
[12] Agmon N, Kaminka G and Kraus S 2011 Multi-robot adversarial patrolling: facing a full-knowledge opponent *J. Artif. Intell. Res.*(*JAIR*) **42** 887–916
[13] Agmon N, Kraus S and Kaminka G 2008 Multi-robot perimeter patrol in adversarial settings *IEEE Int. Conf. on Robotics and Automation,* (*ICRA'08*) (*Pasadena, CA, USA*) pp 2339–45
[14] Amigoni F, Basilico N and Gatti N 2009 Finding the optimal strategies for robotic patrolling with adversaries in topologically—represented environments *IEEE Int. Conf. on Robotics and Automation* (*ICRA'09*) (*Kobe, Japan*) pp 819–24
[15] Martins-Filho L S and Macau E E N 2007 Patrol mobile robots and chaotic trajectories *Math. Probl. Eng.* **2007** 61543
[16] Martins-Filho L S and Macau E E N 2007 Trajectory planning for surveillance missions of mobile robots *Autonomous Robots and Agents* ed S C Mukhopadhyay and G S Gupta (Berlin: Springer) pp 109–17
[17] Volos C K, Kyprianidis I M and Stouboulos I N 2013 Experimental investigation on coverage performance of a chaotic autonomous mobile robot *Robot Auton. Syst.* **61** 1314–22
[18] Volos C K, Kyprianidis I M and Stouboulos I N 2012 A chaotic path planning generator for autonomous mobile robots *Robot Auton. Syst.* **60** 651–6
[19] Li C, Wang F, Zhao L, Li Y and Song Y 2013 An improved chaotic motion path planner for autonomous mobile robots based on a logistic map *Int. J. Adv. Rob. Syst.* **10** 273
[20] Curiac D I and Volosencu C 2014 A 2D chaotic path planning for mobile robots accomplishing boundary surveillance missions in adversarial conditions *Commun. Nonlinear Sci.* **19** 3617–27
[21] Curiac D I and Volosencu C 2009 Developing 2D trajectories for monitoring an area with two points of interest *Proc. 10th WSEAS Int. Conf. Automation and Information* (*Prague, Czech Republic*) pp 366–9
[22] Curiac D I and Volosencu C 2012 Chaotic trajectory design for monitoring an arbitrary number of specified locations using points of interest *Math. Probl. Eng.* **2012** 940276
[23] Ng J and Bräunl T 2007 Performance comparison of bug navigation algorithms *J. Intell. Robot. Syst.* **50** 73–84
[24] LaValle S M 2006 *Planning Algorithms* (Cambridge: Cambridge University Press)
[25] Kamon I, Rimon E and Rivlin E 1998 Tangentbug: a range-sensor-based navigation algorithm *Int. J. Robot. Res.* **17** 934–53
[26] Choset H M (ed) 2005 *Principles of Robot Motion: Theory, Algorithms, and Implementation* (Cambridge, MA: MIT)
[27] Kerrigan B and Yu C 2012 *A Study of Entropy Sources in Cloud Computers: Random Number Generation on Cloud Hosts* (Berlin: Springer)
[28] Kellert S H 1994 *In the Wake of Chaos: Unpredictable Order in Dynamical Systems* (Chicago IL: University of Chicago Press)
[29] Arnold V I and Avez A 1968 *Ergodic Problems of Classical Mechanics* (New York: Benjamin)
[30] Chen G, Mao Y and Chui C K 2004 A symmetric image encryption scheme based on 3D chaotic cat maps *Chaos Solitons Fractals* **21** 749–61
[31] Weigert S 1993 Quantum chaos in the configurational quantum cat map *Phys. Rev.* A **48** 1780–98
[32] Barash L and Shchur L N 2006 Periodic orbits of the ensemble of Sinai-Arnold cat maps and pseudorandom number generation *Phys. Rev.* E **73** 036701