Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 837259, 11 pages
http://dx.doi.org/10.1155/2015/837259

Research Article

Modeling and Analysis of the Obstacle-Avoidance Strategies for a Mobile Robot in a Dynamic Environment

Rui Wang, Ming Wang, Yong Guan, and Xiaojuan Li

College of Information Engineering, Capital Normal University, Beijing 100048, China

Correspondence should be addressed to Rui Wang; rwang04@cnu.edu.cn

Received 13 April 2015; Accepted 16 July 2015

Academic Editor: Krishnaiyan Thulasiraman

Obstacle avoidance is a key performance of mobile robots. However, its experimental verification is rather difficult, due to the probabilistic behaviors of both the robots and the obstacles. This paper presents the Markov Decision Process based probabilistic formal models for three obstacle-avoidance strategies of a mobile robot in an uncertain dynamic environment. The models are employed to make analyses in PRISM, and the correctness of the analysis results is verified by MATLAB simulations. Finally, the minimum time and the energy consumption are determined by further analyses in PRISM, which prove to be useful in finding the optimal strategy. The present work provides a foundation for the probabilistic formal verification of more complicated obstacle-avoidance strategies.

1. Introduction

Obstacle avoidance is a key technique for the design and applications of mobile robots, because static barriers and even dynamic obstacles frequently exist in their paths. When several robots move in the same region, they act in fact as obstacles to one another themselves [1, 2]. Therefore, studies on obstacle avoidance have become an active topic in the field of mobile robots, and a series of strategies have been proposed to realize the avoidance of static and/or dynamic obstacles [2–5]. In the above references, the dynamic environments of the mobile robots are known in advance, because the obstacles are assumed to have predefined moving behaviors. However, mobile robots occasionally have to work in uncertain circumstances, where the movements of the obstacles cannot be predicted accurately [6, 7]. Besides, the sensor and actuator noises, the time delay of network, and other factors may still lead to inaccurate operations of the robots themselves [8]. Due to these reasons, probability-based obstacle-avoidance algorithms have been put forward in recent years [9–11], which models both the movements of the obstacles and the operations of the robots as probabilistic events.

The correctness of obstacle-avoidance algorithm is very important. Testing has been the most frequently used method in the performance validation of robots, but it is by no means the best way [12]. Its deficiencies mainly lie in two aspects: (1) the testing results are incomplete, because only a subset of all the possible cases can be examined by physical experiments or software simulations; (2) the testing results are generally small-sample data that are unsuitable for probabilistic analysis.

Formal verification [13] is a useful alternative method to physical experiments and software simulations, because it is not only complete in logic but also adaptable for the description and analysis of probabilistic events [14–16]. For example, probabilistic model checking (PMC) is a probability-based formal verification method widely used in the performance validation of software and hardware [17, 18]. There are three main steps involved in PMC. First, a mathematical model is built to incorporate all the possible probabilistic behaviors; second, Probabilistic Computation Tree Logic (PCTL) formulae [19] are derived to describe the key logical requirements; third, a software tool such as PRISM [20] is employed to check whether the mathematical model satisfies the logical requirements or not. If all the requirements are fully satisfied, the probabilistic behaviors are validated. Otherwise, it implies that some errors may exist in the original model. In recent years, formal verification has

been widely used to verify the path planning of mobile robots. However, PMC has been scarcely applied in validating the obstacle-avoidance strategies.

The purpose of the present paper is to verify three avoidance algorithms that involve a mobile robot and a single dynamic obstacle [9]. The paths of the robot and obstacle cross each other, and their movements have the properties of Markov Decision Processes (MDPs) [21, 22]. The probabilistic models and the PCTL formulae are first built, and then the PRISM software is employed to make analyses on the three avoidance strategies. Both the time and the energy requirements for the robot to reach the destination have been obtained, and the time or energy based optimal condition is thereby applied to help find the optimal obstacle-avoidance strategy under diverse circumstances. In addition, simulations are also made in MATLAB to verify the correctness of the results obtained by PRISM. To this end, the time of obstacle-avoidance output by PRISM is taken as the input of MATLAB.

The rest of the paper is organized as follows. Section 2 introduces the necessary preliminaries. In Section 3, we formulate the obstacle-avoidance algorithms. Then, the MDPs are constructed in Section 4, and the verification on the algorithm is made in Section 5. Finally, conclusions are presented in Section 6.

2. Preliminaries

2.1. Markov Decision Process. A Markov Decision Process (MDP) is generally defined as a multituple [18]. In the present work, a five-tuple, $(S, S_0, \text{Acts}, \text{Steps}, \text{Cost})$, is adopted to describe the probabilistic model, where

(1) S represents a group of states,

(2) S_0 is the initial state and it belongs to S,

(3) Acts stand for a set of actions, each of which has its own probability,

(4) Steps are the state transition functions,

(5) Cost is a reward function that describes quantitative information of the model.

Figure 1 shows the schematic diagram of a simple MDP, where $p1$, $p2$, $p3$, $p4$, and $p5$ are the probabilities of the corresponding actions, $S = \{S_0, S_1, S_2, S_3\}$, $S_0 = S_0$, $\text{Acts}(S_0) = \{A\}$, $\text{Acts}(S_1) = \{B, C\}$, $\text{Acts}(S_2) = \{D\}$, $\text{Acts}(S_3) = \{E\}$, and the four state transition functions are

$$\begin{aligned}
&\text{Steps}(S_0) = \{(A, [S_1 \longrightarrow p1])\},\\
&\text{Steps}(S_1) = \{(B, [S_0 \longrightarrow p2, S_1 \longrightarrow (1-p2)]),\\
&\quad (C, [S_2 \longrightarrow p3, S_3 \longrightarrow (1-p3)])\},\\
&\text{Steps}(S_2) = \{(D, [S_2 \longrightarrow p4])\},\\
&\text{Steps}(S_3) = \{(E, [S_0 \longrightarrow p5, S_3 \longrightarrow (1-p5)])\}.
\end{aligned} \tag{1}$$

In addition, the reward function "Cost" only describes quantitative information that can be calculated somehow but is generally difficult to be shown in the schematic diagram.

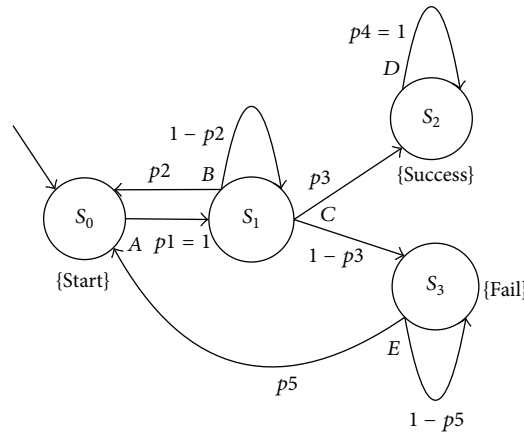


Figure 1: Schematic diagram of a simple MDP.

2.2. Probabilistic Computation Tree Logic. Probabilistic Computation Tree Logic (PCTL) is an extension of the nonprobabilistic Computation Tree Logic (CTL), which expresses the qualitative properties of the probabilistic models [19]. Generally, a PCTL is expressed as a formula that connects the properties of the MDP using the Boolean operators (such as $\neg$ (negation) and $\wedge$ (conjunction)), the temporal operators (such as X (next), $\mathcal{U}$ (until), F (future), and G (globally)), the probabilistic operator P, and a reward (or cost) operator R.

A PCTL formula can be used to calculate a certain quantity of the MDP such as the maximum probability and the minimum time (or energy) consumption. For example, using the following formula

$$P\max =? \left[F\left(\text{spec1} \wedge \text{spec2}\right)\right], \tag{2}$$

one can compute the maximum probability of a state that satisfies both the two specifications spec1 and spec2.

Again, the following formula can be used to find the minimum cost of energy when the specification spec3 is satisfied:

$$R\{\text{“energy”}\}\min =? \left[F\left(\text{spec3}\right)\right]. \tag{3}$$

Finally, consider another PCTL formula

$$P\max =? \left[\text{spec4} \wedge \left(\neg \text{spec5} \mathcal{U} \text{spec6}\right)\right]. \tag{4}$$

Obviously, it can be used to calculate the maximum probability of a state, which meets the two requirements spec4 and spec6 but does not satisfy spec5.

3. Obstacle-Avoidance Algorithms

3.1. Probabilistic Model of a Dynamic Obstacle. Divide the whole moving process of the dynamic obstacle into n steps, each of which has the same time interval ΔT. Assume that the dynamic obstacle has a stepwise uniform motion; that is, it has different velocity at different time step. In the whole process of movement, its minimum and maximum velocities are denoted by $V_{\min}$ and $V_{\max}$. Then, the probability density

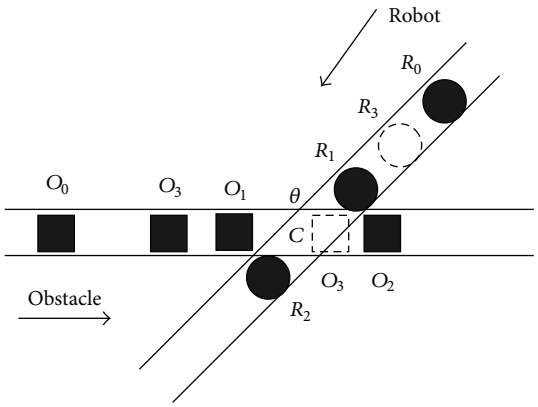


Figure 2: The paths of a mobile robot and a dynamic obstacle.

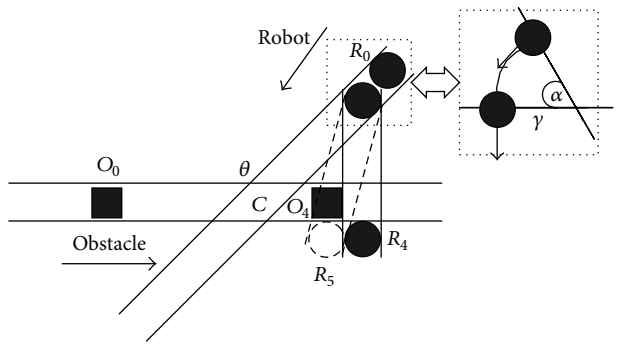


Figure 3: The detour strategy.

function $p(x;k)$ for the obstacle to reach the position x after k steps can be expressed as [9]

$$p(x;k) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}, \tag{5}$$

where

$$\begin{aligned} \sigma^2 &= \sigma_0^2 + k\sigma_{\text{step}}^2; \\ \mu &= x_0 + k\overline{V}\Delta T; \\ \sigma_{\text{step}}^2 &= \frac{(V_{\max} - V_{\min})^2}{12}. \end{aligned} \tag{6}$$

Here, x_0, σ_0^2, and $\overline{V}$ are the initial position, variance, and average velocity, respectively. Integrating $p(x;k)$ along the practical path of the obstacle, one can obtain the probability of reaching the position x.

3.2. Obstacle-Avoidance Strategies of a Mobile Robot. Here, three obstacle-avoidance strategies are given for a mobile robot with only a single dynamic obstacle. As shown in Figure 2, the former and the latter are represented by a circle and a square, respectively.

Assume that the paths of the robot and the obstacle intersect at region C with angle θ ($0^\circ < \theta < 180^\circ$) (see Figure 2). O_1 and R_1 represent their locations when they begin to enter region C, while O_2 and R_2 stand for those when they just leave such a region completely. R_0 is a reference position of the robot, which can be specified at an arbitrary point not over R_1. O_0 and O_3 are two reference positions of the obstacle to be determined in subsequent derivation. R_3 is an undetermined position of the robot.

Under this condition, we mainly consider three avoidance strategies: Acceleration, Deceleration, and Detour. In the Acceleration mode, the robot goes across region C earlier than the obstacle does by enhancing its velocity; in the Deceleration one, it passes region C later than the obstacle does by decreasing its velocity; in the Detour one, it realizes obstacle avoidance by changing its moving direction as shown in Figure 3, where α is the rotation angle and γ is the rotation radius. When α and γ are given, the robot may go along either R_0_R_4 or R_0_R_5, due to its probabilistic behavior. Here, R_0_R_4 represents the expected Detour path, while R_0_R_5 denotes the unexpected one. In Figure 3, O_4 represents the position where the obstacle begins to enter the path intersection region in the expected Detour mode.

However, how to choose one of these three strategies under a specific circumstance is still a problem that needs to be solved. Here, the concept of Dangerous Region is introduced to help explain the decision-making process of the robot. If the obstacle stays in such a region at the initial time, the robot has to choose a certain strategy to avoid collision.

To describe such a concept, one needs to make the following three parameters clear at first.

(a) Reference position O_0: when the two paths are given, the position of O_1 is fixed (see Figure 2). Therefore, the point O_0 can be determined by the distance D_{in}, which is defined as [9]

$$D_{\text{in}} = D_r \left\{ \sqrt{\frac{{V_r}^2 + {V_{\max}}^2 - V_r V_{\max}\cos\theta}{{V_r}^2}} + \frac{V_{\max}}{V_r} \right\}, \tag{7}$$

where V_r is the initial velocity of the robot and D_r is the distance between R_1 and R_2.

(b) Time t_r: $t_r = D_r/V_r$.

(c) Reference position O_3: it is defined in such a way that the obstacle can take less time than t_r to go from it to O_2 with the velocity $V_{\min}$.

Then, we name the region between O_0 and O_3 as the Dangerous Region. Note: the distance between O_0 and O_3 is less than that between O_0 and O_2.

4. Construction of MDPs

This section gives the MDP models for both the obstacle and the robot. The model symbols are shown in Symbol Description.

4.1. Formal Model of the Dynamic Obstacle. Assume that the dynamic obstacle moves along its path from the initial position O_S to the terminal one O_E. In this process, it passes

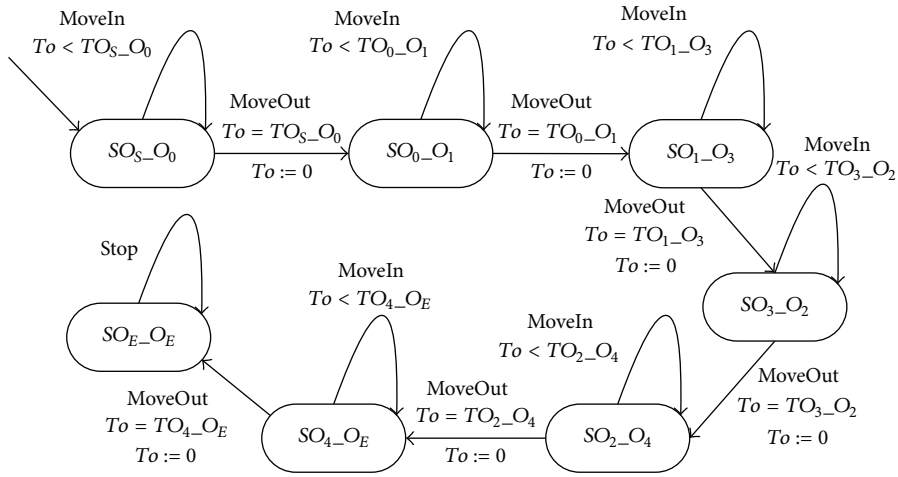


Figure 4: Formal model of the dynamic obstacle.

the locations O_0, O_1, O_3, O_2, and O_4 in turn. Under this condition, its movement is divided into seven states, and then its formal model is constructed as shown in Figure 4. According to the obstacle-avoidance strategies, $O_0\_O_3$ is the Dangerous Region. Therefore, if the obstacle is in the state $SO_0\_O_1$ or $SO_1\_O_3$, the robot should choose a strategy to avoid collision.

To help understand the model, three typical state transitions are explained as follows:

$$(1)\ \langle SO_S\_O_0 \xrightarrow{To<TO_S\_O_0} SO_S\_O_0\rangle.$$

When the timer To is less than $TO_S\_O_0$, the obstacle is in the state $SO_S\_O_0$; that is, the obstacle is moving in the region $O_S\_O_0$.

$$(2)\ \langle SO_S\_O_0 \xrightarrow{To=TO_S\_O_0;\ To:=0} SO_0\_O_1\rangle.$$

When To becomes equal to $TO_S\_O_0$, the obstacle switches to the state $SO_0\_O_1$ from the state $SO_S\_O_0$; that is, the obstacle moves into the region $O_0\_O_1$ from the region $O_S\_O_0$, and then the timer To is reset to 0.

$$(3)\ \langle SO_E\_O_E \rightarrow SO_E\_O_E\rangle.$$

The obstacle stays in the state $SO_E\_O_E$ and will not switch to other states; that is, the obstacle reaches its terminal position.

Dividing each region in Figure 4 into some equal steps in time, one can compute the probability for the obstacle to reach the position x by integrating the probability density function $p(x;k)$ in (5).

4.2. Formal Model of the Mobile Robot. This section provides the formal representation for the state transitions of the robot (see Figures 5–7). In the description, we assume that each avoidance strategy can be executed by the robot with a certain probability. Specifically, when the robot takes the "*Detour*" operation, it can rotate to the expected angle with probability $p1$, as a result of the wheel slippage and other factors; if the robot selects the action "*Acceleration*" or "*Deceleration,*" the probability of changing its velocity to the expected value is $p2$ or $p3$, because of the actuator error and other uncertainties.

4.2.1. Detour Strategy. The mobile robot moves from its initial position R_S to the reference one R_0 and then takes the detour strategy to avoid the obstacle. If the robot rotates to the expected angle, it can move into the desired region $R_0\_R_4$ successfully (see Figure 3). Otherwise, it may go into another arbitrary region such as $R_0\_R_5$ (see Figure 3). As stated above, it rotates to the expected angle with probability $p1$. Therefore, the robot can go into $R_0\_R_4$ with probability $p1$, while it will deviate into $R_0\_R_5$ with probability $1 - p1$. In the former case, the collision will be effectively avoided and the robot can go to its terminal R_E safely, because the detour operation is successful. However, in the latter case, whether the robot will collide with the obstacle or not is still uncertain. If the collision is avoided, the robot will also go to R_E successfully. Otherwise, the collision will make the detour operation fail.

To help understand the model, four typical state transitions are explained as follows:

$$(1)\ \langle SR_S\_R_0 \xrightarrow{p1,Tr=TR_S\_R_0\&\text{Safe}=0} SR_0\_R_4\rangle.$$

When the mobile robot is in the state $SR_S\_R_0$ and the value of Safe is equal to 0 (i.e., the obstacle is in the Dangerous Region), the robot will take the detour strategy to switch to the state $SR_0\_R_4$. The detour operation can be successfully made with probability $p1$.

$$(2)\ \langle SR_S\_R_0 \xrightarrow{1-p1,Tr=TR_S\_R_0\&\text{Safe}=0} SR_0\_R_5\rangle.$$

When the robot is in the state $SR_S\_R_0$ and the value of Safe is equal to 0, the robot will switch to the state $SE_0\_R_5$ with probability $(1 - p1)$.

$$(3)\ \langle SR_0\_R_5 \xrightarrow{\text{Conflict}=1;Tr:=0} \text{Collision}\rangle.$$

When the robot is in the state $SR_0\_R_5$ and the value of Conflict is equal to 1 (i.e., the robot collides with the obstacle), the robot will switch to the Collision state.

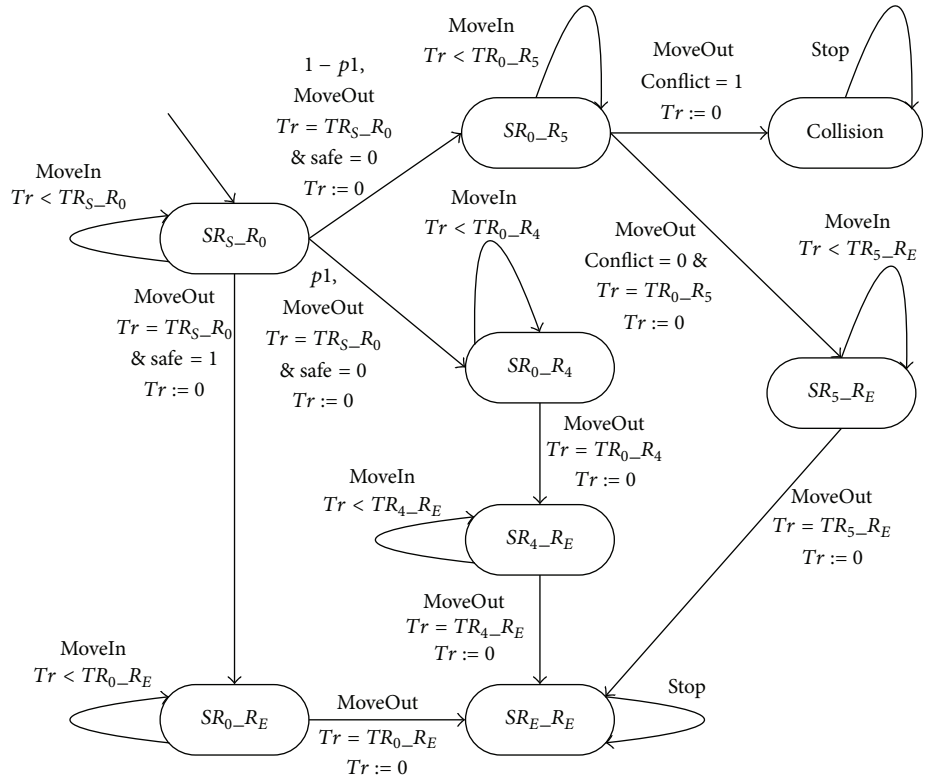


Figure 5: Formal model of the Detour strategy.

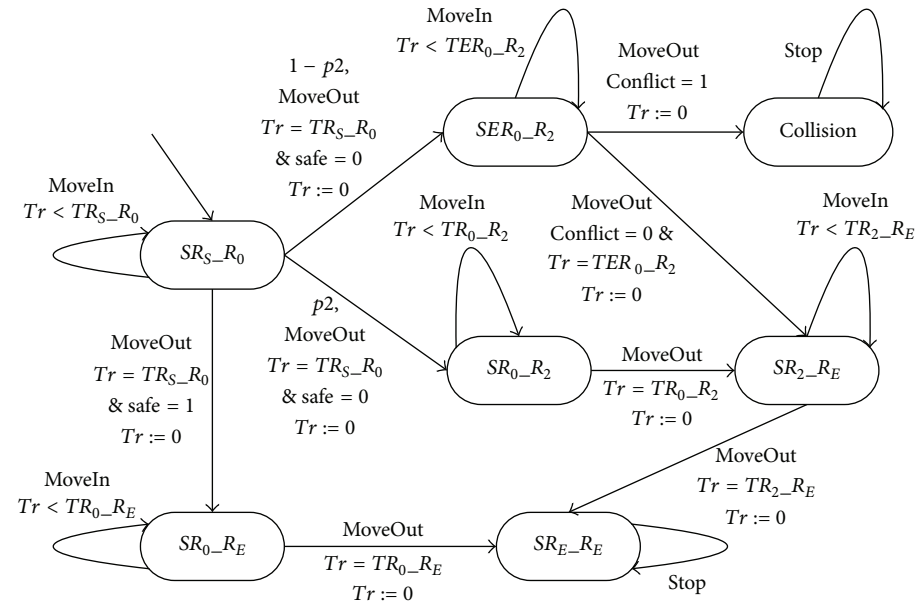


Figure 6: Formal model of the Acceleration strategy.

(4) ⟨Collision → Collision⟩.

The robot will stay in the Collision state and stop moving.

4.2.2. Acceleration Strategy. If the robot locating at R_0 takes the Acceleration strategy to avoid collision, it will move to R_2 by enhancing its velocity. As stated above, the probability of increasing its velocity to the expected value is $p2$. Therefore, it can reach R_2 with probability $p2$ and then go to the terminal R_E safely. In this strategy, it is easy to infer that the change of velocity will fail with probability $1 - p2$. When the desired velocity is not obtained, whether the robot will collide with

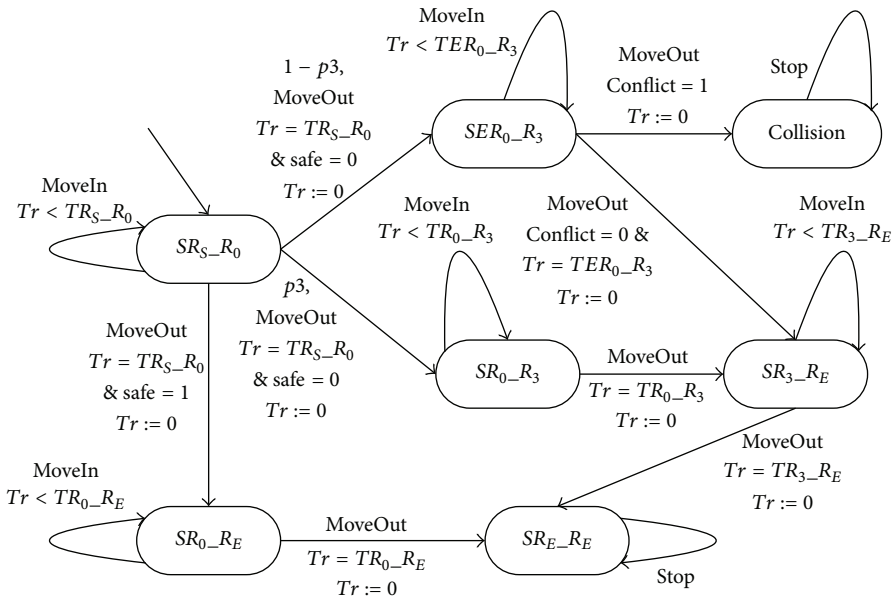


Figure 7: Formal model of the Deceleration strategy.

the obstacle or not is uncertain. If the collision is avoided, the robot will also go to R_E successfully.

Two typical state transitions are explained as follows:

$$(1)\ \langle SR_S\_R_0 \xrightarrow{p2, Tr=TR_S\_R_0 \& Safe=0} SR_0\_R_2 \rangle.$$

When the robot is in the state $SR_S\_R_0$ and the value of Safe is equal to 0, the robot chooses the Acceleration action. Its state switches from $SR_S\_R_0$ to $SR_0\_R_2$ with probability $p2$.

$$(2)\ \langle SR_S\_R_0 \xrightarrow{1-p2, Tr=TR_S\_R_0 \& Safe=0} SER_0\_R_2 \rangle.$$

When the robot is in the state $SR_S\_R_0$ and the value of Safe is equal to 0, the robot chooses the Acceleration action. Its state will change from $SR_S\_R_0$ to $SER_0\_R_2$ with probability $(1-p2)$, because the Acceleration action will fail with probability $(1 - p2)$.

4.2.3. Deceleration Strategy. If the robot locating at R_0 takes the Deceleration strategy to avoid collision, it will move to a certain position R_5 between R_0 and R_1 by reducing its velocity. As stated above, the probability of decreasing its velocity to the expected value is $p3$. Therefore, it can reach R_5 with probability $p3$ and then go to the terminal R_E safely. Obviously, the change of velocity will fail with probability $1-p3$. When the desired velocity is not obtained, whether the robot will collide with the obstacle or not is uncertain. If the collision is avoided, the robot will also go to R_E successfully.

Two typical state transitions are as follows:

$$(1)\ \langle SR_S\_R_0 \xrightarrow{p3, Tr=TSR_S\_R_0 \& Safe=0} SR_0\_R_3 \rangle.$$

When the robot is in the state $SR_S\_R_0$ and the value of Safe is equal to 0, the robot chooses the Deceleration action. Its state switches from $SR_S\_R_0$ to $SR_0\_R_3$ with probability $p3$.

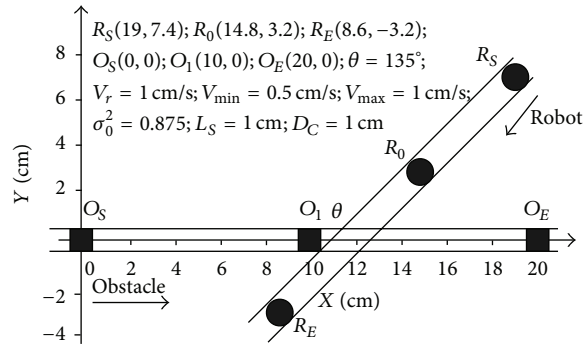


Figure 8: An example of obstacle avoidance.

$$(2)\ \langle SR_S\_R_0 \xrightarrow{1-p3, Tr=SR_S\_R_0 \& Safe=0} SER_0\_R_3 \rangle.$$

When the robot is in the state $SR_S\_R_0$ and the value of Safe is equal to 0, the robot chooses the Deceleration action. Its state will change from $SR_S\_R_0$ to $SER_0\_R_3$ with probability $(1-p3)$, because the Deceleration action will fail with probability $(1 - p3)$.

5. Probabilistic Analysis and Verification

In this section, we take an example to make probabilistic analysis and verification. The velocities and geometrical parameters of the example are shown in Figure 8, where D_C and L_S represent the diameter of the circle and the side length of the square, respectively.

Next, probabilistic analysis is performed in the PRISM software to get the probability-time curves of the three obstacle-avoidance strategies. Then, the corresponding computation is also carried out in MATLAB to verify the effectiveness of the analysis in PRISM. On this basis, the time

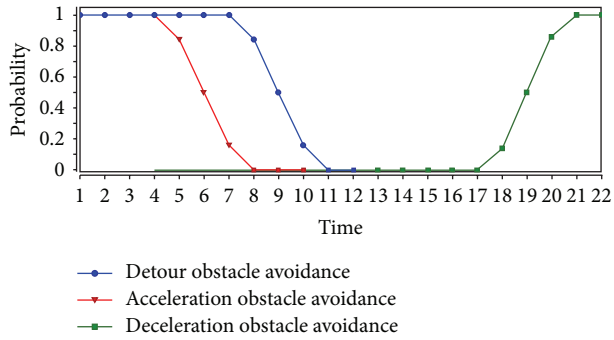


Figure 9: The probability-time curves of the three obstacle-avoidance strategies.

and energy consumption of the three strategies are calculated in PRISM to help find the optimal strategy.

5.1. Mobile Robot Transiting through the Expected States. Here, we consider the cases of the mobile robot moving through the states $SR_0\_R_3$ in Detour, $SR_0\_R_2$ in Acceleration, and $SR_0\_R_5$ in Deceleration. It is assumed that $p1 = p2 = p3 = 1$.

5.1.1. Probability-Time Curves Given by PRISM. To get the probability-time curves by analysis in PRISM, one has to construct the following PCTL formula at first:

$$P\max =? \left[F\left(\text{Success} = 1\right)\right], \tag{8}$$

where the value 1 of Success means that the robot succeeds in avoiding the obstacle.

The analysis results corresponding to the three avoidance strategies are shown in Figure 9.

(1) Detour Strategy. (The rotation radius γ of the robot is prescribed as 1.0 cm.) If the time for the robot to pass the region $R_0\_R_4$ is less than or equal to 7 s, the obstacle will be successfully avoided.

(2) Acceleration Strategy. If the time to go across the region $R_0\_R_2$ does not exceed 4 s, the robot is able to avoid collision.

(3) Deceleration Strategy. If the time to pass the region $R_0\_R_1$ is more than or equal to 21 seconds, the collision can be prevented.

5.1.2. Simulation Results by MATLAB. In this subsection, MATLAB is used to make simulation to verify the correctness of the results obtained by PRISM. To this end, the time of obstacle-avoidance output by PRISM is taken as the input of MATLAB.

Nevertheless, how to determine the input time is still a problem needing consideration. In the strategy of Detour or Acceleration, if the robot takes longer time to pass the region $R_0\_R_4$ or $R_0\_R_2$, the probability for it to avoid collision will become smaller. However, shorter time to pass $R_0\_R_1$ will give rise to smaller obstacle-avoidance probability in the Deceleration mode. Therefore, we choose the maximum time of obstacle avoidance in Detour (i.e., 7 s) or Acceleration (i.e., 4 s) but the minimum one in Deceleration (i.e., 21 s) to make verification in MATLAB.

The visualized results of the simulation are provided in Figure 10. It is seen that the collision is effectively avoided in each case, which verifies the analysis results in PRISM.

5.2. Mobile Robot Transiting through the Unexpected States. In this subsection, we consider the cases of the mobile robot moving through the states $SR_0\_R_5$ in Detour, $SER_0\_R_2$ in Acceleration, and $SER_0\_R_3$ in Deceleration.

5.2.1. Verification Results by PRISM. The PCTL formula is

$$P\max =? \left[F\left(\text{ErrorSuccess} = 1\right)\right], \tag{9}$$

where the value 1 of ErrorSuccess means that the robot succeeds in avoiding the obstacle finally, even if it goes through the unexpected state.

The probability-time curves of the three obstacle-avoidance strategies are shown in Figure 11.

(1) Detour Strategy. Assume that the expected rotation angle is 45°, but the actual one is only 15° or 30°. Under this condition, the robot will adjust its motion to avoid collisions. Figure 11(a) shows that when the actual rotation angle is 15° or 30° the maximum time to successfully avoid collision is 5 s or 6 s, respectively.

(2) Acceleration. Like in case A, if the time for the robot to reach R_2 is less than or equal to 4 s, the obstacle will be avoided (see Figure 11(b)).

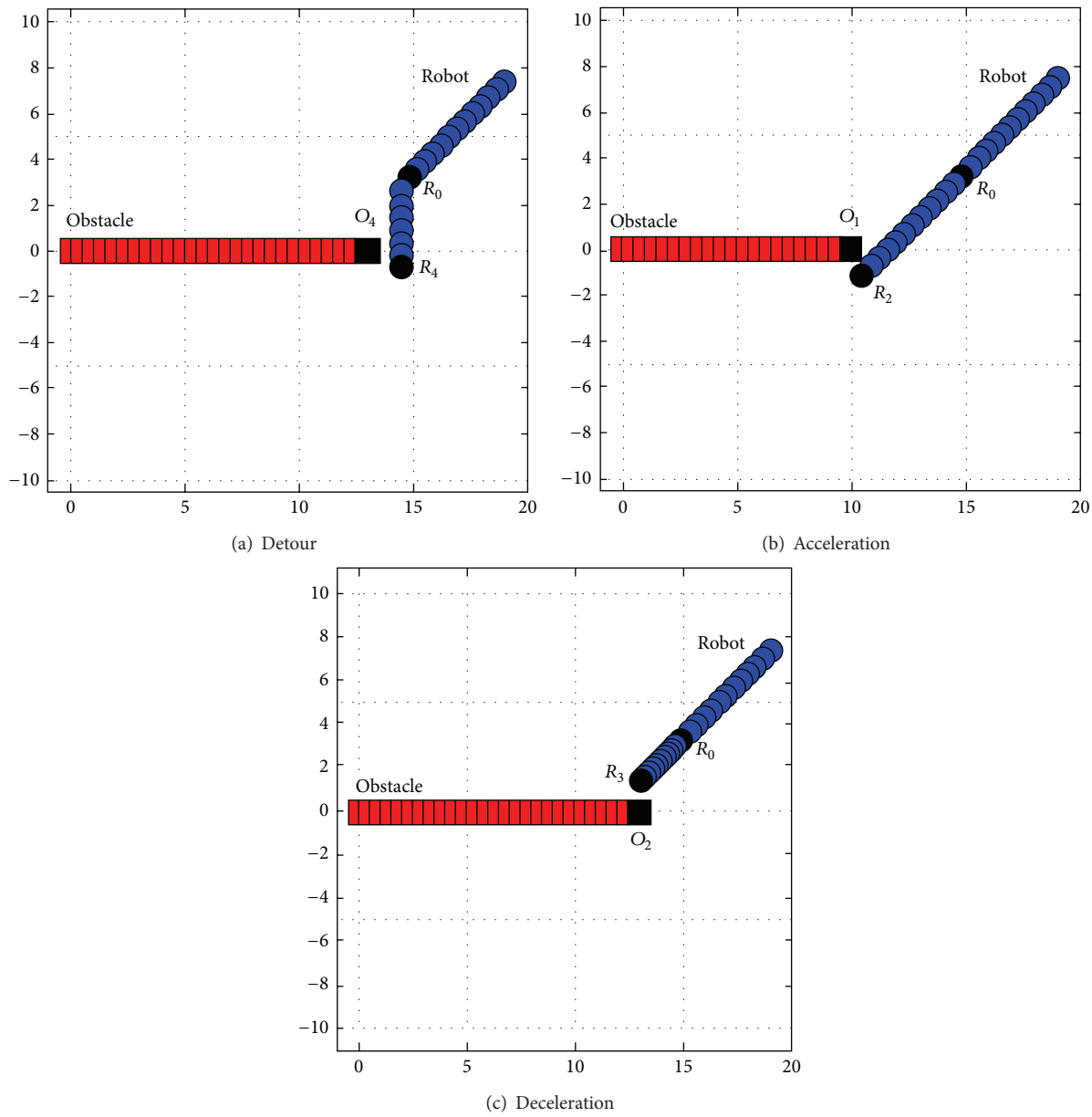
(3) Deceleration Strategy. If the time to arrive at R_1 is longer than 7 s, the robot can avoid the obstacle (see Figure 11(c)).

5.2.2. Simulation Results by MATLAB. In this subsection, simulations are carried out in MATLAB to verify the analysis results shown in Figure 11.

(1) Detour Strategy. Here, the time of obstacle-avoidance output by PRISM is taken as the input of MATLAB. Therefore, when the actual rotation angle is 15° or 30°, we take 5 s or 6 s to perform simulation. Figure 12 shows the visualized results, which obviously verify the results in Figure 11(a).

(2) Acceleration or Deceleration Strategy. For Acceleration or Deceleration, we take six different values of time and four different values of $p2$ or $p3$ to make simulations in MATLAB. In each case, 100 times of simulations are performed to calculate the probability of obstacle avoidance. The results are given in Tables 1 and 2, which are consistent with those shown in Figures 11(b) and 11(c). From this, we come to the conclusion that probabilistic model checking is an effective method to verify the obstacle-avoidance strategies of mobile robots.

5.3. Minimum Time and Energy Consumption. Here, the minimum time and energy consumption for the robot to go



(a) Detour (b) Acceleration

(c) Deceleration

Figure 10: Simulation results in MATLAB.

Table 1: Simulation results of the acceleration strategy.

Time (s)	Probability of obstacle avoidance			
	$p2 = 0$	$p2 = 0.3$	$p2 = 0.6$	$p2 = 0.9$
3	0.999	0.998	0.998	0.997
4	0.998	0.998	0.997	0.996
5	0.8500	0.890	0.940	0.988
6	0.5200	0.690	0.840	0.970
7	0.1800	0.420	0.674	0.926
8	0.0300	0.320	0.600	0.910

Table 2: Simulation results of the deceleration strategy.

Time (s)	Probability of obstacle avoidance			
	$p3 = 0$	$p3 = 0.3$	$p3 = 0.6$	$p3 = 0.9$
16	0.02	0.320	0.600	0.910
17	0.05	0.340	0.610	0.930
18	0.16	0.400	0.676	0.924
19	0.52	0.670	0.820	0.960
20	0.87	0.898	0.938	0.989
21	0.98	0.989	0.990	0.999

from its initial position to the destination are used to find the optimal obstacle-avoidance strategy. For this purpose, analysis is further performed in PRISM to determine these two parameters for different obstacle-avoidance paths.

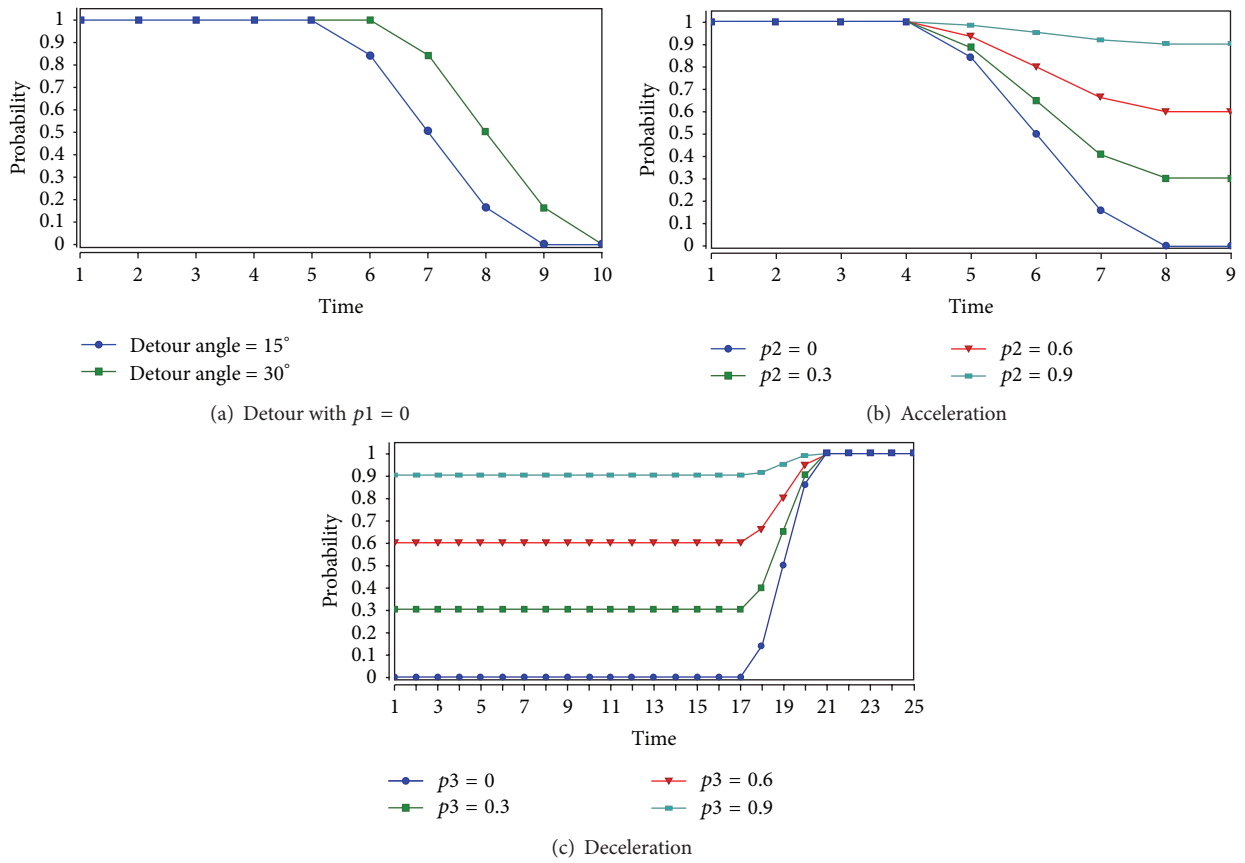


FIGURE 11: The probability-time curves of the three obstacle-avoidance strategies.

The PCTL formula corresponding to the minimum time is

$$P\max =? \left[F \leq T \left(\text{Done} = 1\right)\right], \tag{10}$$

where the value 1 of Done means the robot reaches the destination safely.

The PCTL formula associated with the minimum energy is

$$R\left\{\text{“Energy”}\right\}\min =? \left[F\left(\text{Done} = 1\right)\right]. \tag{11}$$

In addition, the kinetic energy of the robot can be calculated as follows:

$$\begin{aligned} E_1 &= \frac{\left(mv^2 + J\omega^2\right)}{2}, \quad \text{(rotation)}, \\ E_2 &= \frac{mv^2}{2}, \quad \text{(linear motion)}, \end{aligned} \tag{12}$$

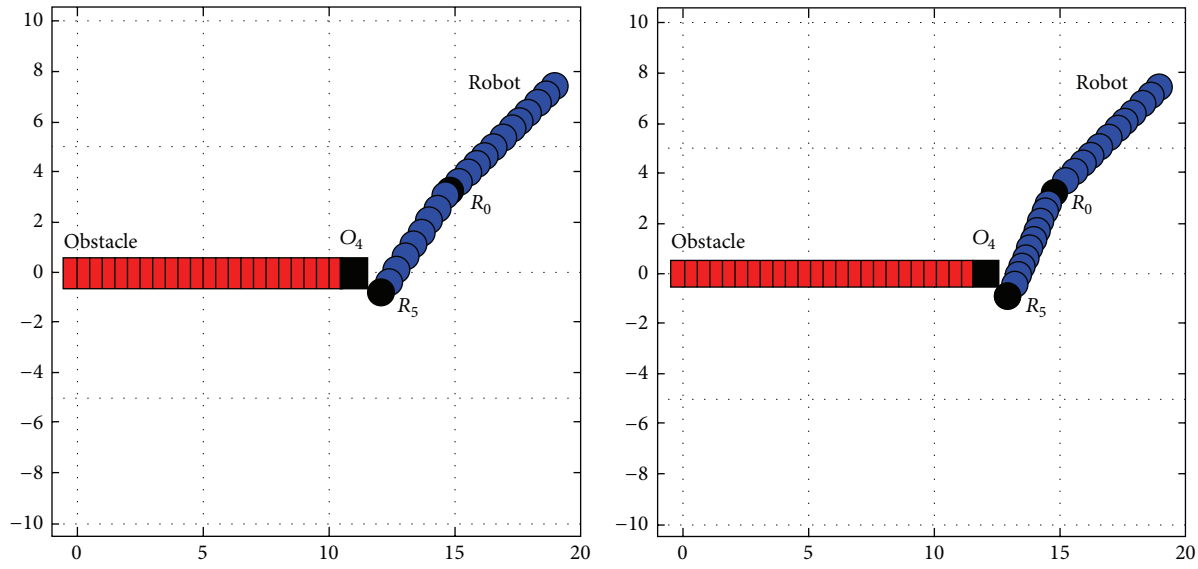
where m is the mass, J is the rotational inertia, v is the linear velocity, and ω is the angular velocity.

The analysis results of the minimum time and energy consumption are shown in Table 3. In the Detour mode, if actual rotation angle is 45°, the robot arrives at its destination by passing the expected state; otherwise, it reaches the terminal by going through the unexpected state. It is seen that the expected path costs the lowest “minimum energy” but the longest “minimum time,” while the unexpected trajectory (15°) consumes the shortest “minimum time” but the highest “minimum energy.” In the acceleration mode, the expected path ($TR_0\_R_2 = 4$) requires the shortest “minimum time” but the highest “minimum energy,” while the unexpected one ($TRE_0\_R_2 = 6$) needs the lowest “minimum energy” but the longest “minimum time.” In the Deceleration mode, the expected path ($TR_0\_R_3 = 21$) spends the longest “minimum time,” while all the paths expend the same “minimum energy.” Further comparison shows that the Acceleration strategy ($TR_0\_R_2 = 4$) is the least time-costing, while the Deceleration one is the least energy-consuming, among all the three kinds of strategies.

Finally, it deserves noting that the data in Table 3 can help us to find the optimal obstacle-avoidance strategy. For example, if the movement time of the robot is limited, we can choose the Acceleration strategy ($TR_0\_R_2 = 4$). If the power supply is not so sufficient, the Deceleration strategy can be selected instead.

6. Conclusions

The MDP-based probabilistic formal models are constructed for three different kinds of obstacle-avoidance strategies for a mobile robot in an uncertain dynamic environment.



(a) Detour with 15°

(b) Detour with 30°

Figure 12: Simulation results in MATLAB.

Table 3: Minimum time and energy consumption.

Methods	Time (s)	Minimum time (s)	Minimum energy (10^{-7} J)
Detour (45°)	$TR_0\_R_4 = 5$	24	2.2
Detour (30°)	$TR_0\_R_5 = 5$	23	2.4
Detour (15°)	$TR_0\_R_5 = 5$	22	2.6
Acceleration	$TR_0\_R_2 = 4$	19	3.5
Acceleration	$TER_0\_R_2 = 5$	20	3.4
Acceleration	$TER_0\_R_2 = 6$	21	3.3
Deceleration	$TR_0\_R_3 = 21$	40	1.5
Deceleration	$TER_0\_R_3 = 20$	39	1.5
Deceleration	$TER_0\_R_3 = 19$	38	1.5

The PCTL formulae are then built to perform analysis in the PRISM software, which yields the probability-time curves for the obstacle-avoidance paths. In addition, the time of obstacle-avoidance output by PRISM is taken as input to conduct simulations in MATLAB, which verify the correctness of the results obtained by PRISM. Further, both the minimum time and the energy consumption for the robot to reach its destination under each obstacle-avoidance strategy are obtained by analyses in PRISM. Finally, the time or energy based optimal condition is thereby applied to determine the optimal obstacle-avoidance strategy under diverse circumstances.

It deserves noting that the present paper only verifies the probabilistic obstacle-avoidance strategies that involve a mobile robot and a single dynamic obstacle. In practice, there may be more complex obstacle-avoidance conditions. For example, a single mobile robot may encounter several dynamic obstacles, or, even worse, a swarm of mobile robots may have to go across a terrain with multiple dynamic obstacles. These complex cases require more complicated probabilistic formal verifications. The present paper lays a good foundation for these future works.

Symbols

To, Tr: Two timers
A_B: It represents the region between A and B, where $A \in \{O_S, O_0, O_1, O_2, O_3, O_4, O_E\}$ and $B \in \{O_0, O_1, O_2, O_3, O_4, O_E\}$
SA_B: A state index of the dynamic obstacle, which means the obstacle locates at region A_B
TA_B: The time for the obstacle to pass region A_B
Safe: The security index. If the value of Safe is 0, it means that an obstacle exists in the Dangerous Region and the robot needs to avoid collisions
Conflict: The collision index. If the value of Conflict is 1, it means the robot collides with an obstacle
M_N: It represents the region between M and N, where $M \in \{R_S, R_0, R_1, R_2, R_3, R_4, R_5, R_E\}$ and $N \in \{R_0, R_1, R_2, R_3, R_4, R_5, R_E\}$
SM_N: A state index of the robot, which means the robot locates at region M_N
SEM_N: A state index of the robot. It means the robot locates at region M_N with operation errors
TM_N: The time for the robot to pass region M_N
TEM_N: The time for the robot to pass region M_N with operation errors.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported by the International Cooperation Program on Science and Technology (2010DFB10930 and 2011DFG13000), NSFC (61070049, 61170304, 61104035, 61373034, 61303014, and 61472468), NSFCB (4122017), TJSHG (201310028014), KM (201510028015), KZ (201210028036), and Scientific Research Base Development Program of the Beijing Municipal Commission of Education.

References

[1] S. Liu, D. Sun, and C. Zhu, "A dynamic priority based path planning for cooperation of multiple mobile robots in formation forming," *Robotics and Computer-Integrated Manufacturing*, vol. 30, no. 6, pp. 589–596, 2014.

[2] A. Babinec, F. Duchoň, M. Dekan, P. Pásztó, and M. Kelemen, "VFH*TDT (VFH* with Time Dependent Tree): a new laser rangefinder based obstacle avoidance method designed for environment with non-static obstacles," *Robotics and Autonomous Systems*, vol. 62, no. 8, pp. 1098–1115, 2014.

[3] A. Sgorbissa and R. Zaccaria, "Integrated obstacle avoidance and path following through a feedback control law," *Journal of Intelligent & Robotic Systems: Theory and Applications*, vol. 72, no. 3-4, pp. 409–428, 2013.

[4] A. Sgorbissa and R. Zaccaria, "Planning and obstacle avoidance in mobile robotics," *Robotics and Autonomous Systems*, vol. 60, no. 4, pp. 628–638, 2012.

[5] J. Hong and K. Park, "A new mobile robot navigation using a turning point searching algorithm with the consideration of obstacle avoidance," *International Journal of Advanced Manufacturing Technology*, vol. 52, no. 5–8, pp. 763–775, 2011.

[6] N. E. Du Toit and J. W. Burdick, "Robot motion planning in dynamic, uncertain environments," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 101–115, 2012.

[7] R. Vatcha and J. Xiao, "Detection of robustly collision-free trajectories in unpredictable environments in real-time," *Autonomous Robots*, vol. 37, no. 1, pp. 81–96, 2014.

[8] S. J. Yoo, "Adaptive neural tracking and obstacle avoidance of uncertain mobile robots with unknown skidding and slipping," *Information Sciences*, vol. 238, pp. 176–189, 2013.

[9] J. Miura and Y. Shirai, "Modeling motion uncertainty of moving obstacles for robot motion planning," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '00)*, pp. 2258–2263, San Francisco, Calif, USA, April 2000.

[10] H. Ishihara and E. Hashimoto, "Moving obstacle avoidance for the mobile robot using the probabilistic inference," in *Proceedings of the 10th IEEE International Conference on Mechatronics and Automation (ICMA '13)*, pp. 1771–1776, Takamatsu, Japan, August 2013.

[11] S. Loibl, D. Meyer-Delius, and P. Pfaff, "Probabilistic time-dependent models for mobile robot path planning in changing environments," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '13)*, pp. 5545–5550, Karlsruhe, Germany, May 2013.

[12] A. Tharakeshwar and A. Ghosal, "A three-wheeled mobile robot for traversing uneven terrain without slip: simulation and experiments," *Mechanics Based Design of Structures and Machines*, vol. 41, no. 1, pp. 60–78, 2013.

[13] E. M. Clarke and O. Grumberg, *Model Checking*, The MIT Press, 1999.

[14] Y. Jiang, H. Zhang, Z. Li et al., "Design and optimization of multiclocked embedded systems using formal techniques," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 2, pp. 1270–1278, 2015.

[15] T. Chen, V. Forejt, M. Kwiatkowska, D. Parker, and A. Simaitis, "Automatic verification of competitive stochastic systems," *Formal Methods in System Design*, vol. 43, no. 1, pp. 61–92, 2013.

[16] S. Konur, C. Dixon, and M. Fisher, "Analysing robot swarm behaviour via probabilistic model checking," *Robotics and Autonomous Systems*, vol. 60, no. 2, pp. 199–213, 2012.

[17] M. Kwiatkowska, G. Norman, and D. Parker, "Probabilistic verification of Herman's self-stabilisation algorithm," *Formal Aspects of Computing*, vol. 24, no. 4–6, pp. 661–670, 2012.

[18] T.-E. Lee and J.-H. Lee, "A two-phase approach for design of supervisory controllers for robot cells: model checking and Markov decision models," *Annals of Operations Research*, vol. 77, pp. 157–182, 1998.

[19] L. Song, L. Zhang, J. C. Godskesen, and F. Nielson, "Bisimulations meet PCTL equivalences for probabilistic automata," *Logical Methods in Computer Science*, vol. 9, no. 2, article 17, 2013.

[20] M. Kwiatkowska, G. Norman, and D. Parker, "Probabilistic symbolic model checking with PRISM: a hybrid approach," *International Journal on Software Tools for Technology Transfer*, vol. 6, no. 2, pp. 128–142, 2004.

[21] M. Lahijanian, S. B. Andersson, and C. Belta, "Temporal logic motion planning and control with probabilistic satisfaction guarantees," *IEEE Transactions on Robotics*, vol. 28, no. 2, pp. 396–409, 2012.

[22] L. Mikeev, M. R. Neuhäußer, D. Spieler, and V. Wolf, "On-the-fly verification and optimization of DTA-properties for large Markov chains," *Formal Methods in System Design*, vol. 43, no. 2, pp. 313–337, 2013.