IFAC-PapersOnLine, Volume 49, Issue 23, 2016, Pages 127-132
DOI:10.1016/j.ifacol.2016.10.332.

Verifying Collision Avoidance Behaviours for Unmanned Surface Vehicles using Probabilistic Model Checking

Yu Lu, Hanlin Niu*, Al Savvaris, Antonios Tsourdos

School of Aerospace, Transport and Manufacturing, Cranfield University, Cranfield, United Kingdom
* *Email: h.niu@cranfield.ac.uk*

Abstract: Collision avoidance is an essential safety requirement for unmanned surface vehicles (USVs). Normally, its practical verification is non-trivial, due to the stochastic behaviours of both the USVs and the intruders. This paper presents the probabilistic timed automata (PTAs) based formalism for three collision avoidance behaviours of USVs in uncertain dynamic environments, which are associated with the crossing situation in COLREGs. Steering right, acceleration, and deceleration are considered potential evasive manoeuvres. The state-of-the-art PRISM model checker is applied to analyse the underlying models. This work provides a framework and practical application of the probabilistic model checking for decision making in collision avoidance for USVs.

Keywords: unmanned surface vehicles; collision avoidance; probabilistic timed automata; formal methods; probabilistic model checking

1. INTRODUCTION

The development of unmanned surface vehicles spans several decades. The original radio-controlled vessels were designed for damage assessment and dangerous mine clearance operations. Over the past two decades, the development of more advanced sensors and the increased capabilities of computational power and communication technology coupled with a reduction in cost have motivated the use of USVs in novel applications and more complex missions such as minesweeping, environmental data collection and monitoring, water survey, anti-surface, and submarine warfare.

Collision avoidance (Savvaris et al. (2014)) is a central component for the design and development of USVs, due to that static obstacle or even dynamic intruders frequently exist in their paths. When several USVs or other vessels move in the same region, they act in fact as intruders to one another themselves. Thus, research on collision avoidance have become an active topic in the area of autonomous vehicles, and numerous algorithms have been proposed to realise the avoidance of static obstacles or dynamic intruders. In the past, the dynamic environments of USVs may be known in advance, since the intruders are assumed to have predefined or predicted moving behaviours. However, today's USVs commonly have to work in uncertain circumstances, where the movements of the intruders are not easy to be predicted accurately. Consequently, a number of probabilistic collision avoidance algorithms have been proposed in recent years, which models both the movements of the intruders and the operations of the vessels as probabilistic events.

The correctness of collision avoidance algorithms for USVs is very crucial. Simulation and testing have been the most frequently used analysis approach for verifying USVs' behaviours. However, either of them is by no means the best solution. Their weaknesses mainly lie in two aspects: (1) the results are incomplete, due to that only a subset of all the possible cases can be examined by physical system testing or software simulations; (2) the results are generally small sample data that are unsuitable for complex probabilistic analysis.

Formal verification now becomes a very useful alternative approach to traditional analysis approaches such as simulation and testing, because it is not only complete in logic and rigorous in mathematics but also adaptable for the description and analysis of probabilistic events. For example, probabilistic model checking is a quantitative verification approach widely applied in the reliability, safety, and performance analysis of both hardware and software systems. In general, there are three main phases involved in probabilisitic model checking: (1) a high level mathematical model is built to incorporate all the possible probabilistic behaviours; (2) formal logical formulae are derived to describe the key logical requirements; (3) an automatic tool such as PRISM is applied to check whether the mathematical model satisfies the logical requirements. If all the requirements are fully satisfied, the probabilistic behaviours are verified. Otherwise, it implies that some errors may exist in the original model. In recent years, formal verification has already been used to verify the path planning problem for autonomous vehicles (Quottrup et al. (2004); Fainekos et al. (2005)). However, there is little work that applied in verifying the collision avoidance problem in the same domain.

The aim of the paper is to formally verify three avoidance algorithms (steering, acceleration, and deceleration)

that involve an USV and a single dynamic intruder. The paths of the USV and intruder cross each other, and their movements have both probabilistic and real-time properties, which is very suitable for using probabilistic timed automata. Thus, the probabilistic models and the logical formulae are first built, and then the PRISM model checker is applied to verify the underlying three avoidance strategies.

The rest of the paper is organised as follows. In Section 2, we present the collision avoidance algorithms for USVs. Section 3 introduces the necessary preliminaries of probabilistic timed automata and probabilistic model checking. Then, the PTAs are constructed in Sections 4 and 5 respectively. Finally, we conclude the paper in Section 6.

2. COLLISION AVOIDANCE FOR USVS

We have made some assumptions on the motion of the intruder vessel: (1) the whole moving process of the intruder can be divided into n steps; (2) the intruder has a stepwise uniform motion, that is, it has different velocity at different time step, and the minimum and maximum velocities are denoted by v_{min} and v_{max}; (3) the intruder changes the velocity at every time interval $\triangle T$; (4) the velocity for a time interval is constant, and independently and randomly selected within the range of $[v_{min}, v_{max}]$.

Based on the work of Miura and Shirai (2000), the probability distribution $p(x;n)$ for the intruder to reach the position x after n steps can be expressed as:

$$p(x;n) = \frac{1}{\sqrt{2\pi\sigma_i^2}} e^{-(\frac{(x-\bar{x}_i)^2}{2\sigma_i^2})^\gamma}, t \geq 0, \gamma, \alpha > 0 \qquad (1)$$

where $\sigma^2 = \sigma_0^2 + n\sigma_{step}^2$, $\bar{x}_i = x_0 + n\bar{v}\triangle T$, and $\sigma_{step}^2 = (v_{max} - v_{min})^2/12$. Here, x_0, σ, and $\bar{v}$ are the initial position, variance, average velocity, respectively. Integrating $p(x;n)$ along the practical path of the intruder, one can obtain the probability of reaching the position x.

In this paper, three collision avoidance algorithms are given for a USV with only a single dynamic intruder. As shown in Fig. 1, the former and the latter are represented by a black-yellow USV and a white-blue ship, respectively. Assume that the paths of the USV and the intruder intersect at region C with angle θ ($0° < \theta < 180°$) (see Fig. 1). I_2 and U_2 represent their positions when they begin to enter the collision region C, while I_3 and U_4 stand for those when they just leave such a region completely. U_0 is a reference position of the USV, which can be specified at an arbitrary point not over U_2. I_0 and I_1 are two reference positions of the intruder. U_1 is an undetermined position of the USV.

Under this condition, we mainly consider three avoidance behaviours: acceleration, deceleration, and steering. For the acceleration behaviour, the USV goes across region C earlier than the obstacle does by increasing its velocity; for the deceleration case, it passes region C later than the obstacle does by decreasing its velocity; for the steering one, it realises collision avoidance by changing its moving direction as shown in Fig. 2, where α is the heading angle and γ is the turning radius. When α and γ are given,

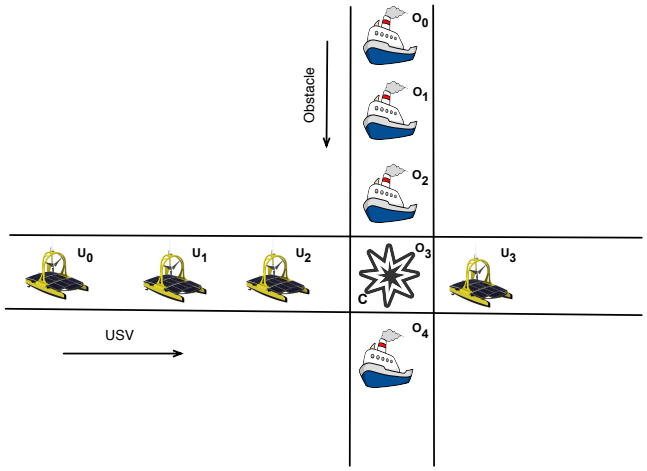

Fig. 1. The paths of an USV and an intruder: the USV is the stand-on vessel with respect to the International Regulations for Preventing Collisions at Sea 1972 (COLREGS).

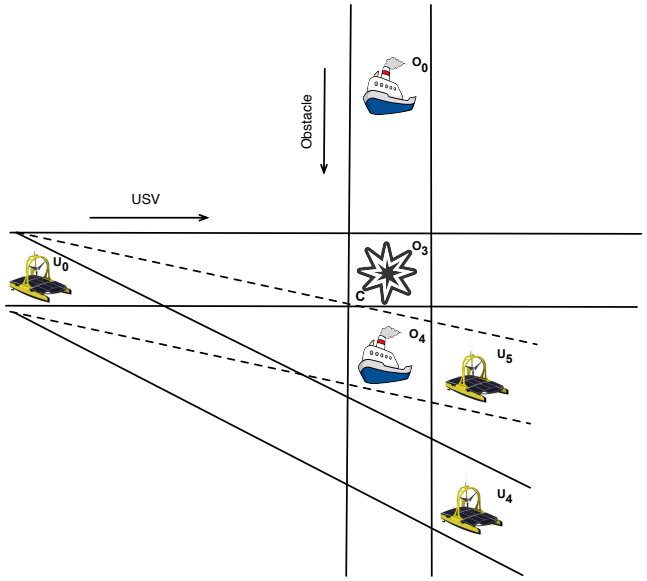

Fig. 2. The steering behaviour of the USV: the USV is the stand-on vessel with respect to the COLREGS.

according to its probabilistic behaviours, the USV may choose either U_0-U_4 or U_0-U_5. We represent U_0-U_4 as the expected path, while U_0-U_4 as the unexpected path. In Fig. 2, $U4$ represents the position where the obstacle begins to enter the path intersection region in the expected steering behaviour.

3. PROBABILISTIC MODEL CHECKING

3.1 Probabilistic Timed Automata

Full details about probabilistic timed automata (PTAs) can be found (Kwiatkowska et al. (2006); Norman et al. (2013)). We outline the important aspects in this section.

PTAs allow us to use the real-valued clocks of timed automata, together with the discrete probabilistic choice of MDPs. PTAs have real-valued clocks and, like MDPs

and therefore allow us to model systems with a range of different characteristics (non-determinism, probability, and real time).

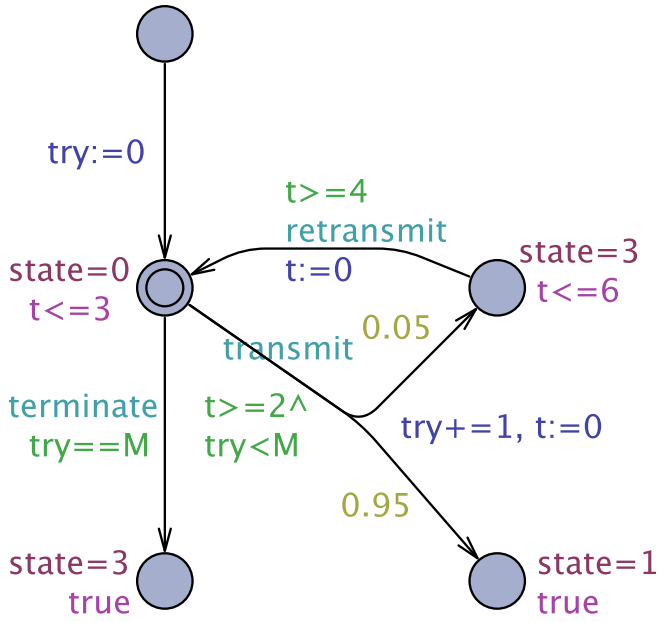

Fig. 3. An example of a probabilistic timed automaton, with clock x and integer variable tries, modelling attempted message transmission over an unreliable channel.

In this section, we illustrate a number of basic PTA concepts using the example in Fig. 3. The Figure illustrates a probabilistic timed automaton, with clock t and integer variable try, modelling a simple probabilistic communication protocol. In the protocol, a sender repeatedly attempts to transmit a message over an unreliable channel. The probability that the sender's transmission fails due to that the channel is unreliable is 0.05, and the sender successfully transmit the message to the receiver with probability 0.95. If message data from the sender is lost, the sender suspends its activity, and there is a delay (of between 4 and 6 time units) before the sender tries to resend its message (up to $M-1$ times).

The control states of the automaton model, "$state = 0$", "$state = 1$", "$state = 2$", "$state = 3$", have the meaning of "transmit", "wait", "quit", and "finish"respectively. They are depicted as the nodes (circles) of the underlying graph, and the available transmissions between these control states are indicated as the edges (with arrow) of the graph. In the initial state, "$state = 0$", shown as the extra border, a communication is being initialised by the sender along the transmission channel. After between 2 and 3 time units, the sender attempts to send the message, and with probability 0.95 message is sent correctly, meantime with probability 0.05 message is lost. In "$s = 1$", when 4 to 6 time units have elapsed from whenever the message is lost, the sender tries to re-transmit the message.

Definition 1. A probabilistic timed automaton PTA is a tuple of the form
$(L, l_0, \Sigma, inv, prob)$ where:

- $L = \{l_0, l_1, l_2, ..., l_n\}$ is a finite set of positions;
- $l_0 \in L$ is the initial position;
- $\chi = \{x, y, z, ...\}$ is a finite set of clocks;
- $\Sigma = \{a, b, c, ...\}$ is a finite set of events, of which $\Sigma_u \subseteq \Sigma$ are declared as being urgent;
- the function $inv : L \rightarrow CC(\chi)$ is the invariant condition;
- the finite set $prob \subseteq L \times CC(\chi) \times \Sigma \times Dist(2^\chi \times L)$ is the probabilistic edge relation.

Note that clocks are real-valued. The values of the clocks synchronise and increase together over time. Transitions and states may have guards and invariants over clock variables and other variables which indicate when transitions can occur and how long can be spent in a state. In our example, the transition between states $state = 0$ and $state = 1$ (or $state = 2$) has the clock guard $t \geq 2$. The state $state = 0$ and $state = 3$ have the invariant t < 3 and t < 6 respectively.

The semantics of PTAs are formally defined as an infinite state MDP. As clocks are real-valued the MDP will have an infinite state-space (both in terms of set of states, and the set of transitions). Since model-checking algorithms are designed to work on finite state spaces, the analysis of PTAs requires some form of abstraction, to a finite state representation. PTAs have been used to verify a variety of protocols, e.g. the CSMA/CD back-off protocol (Duflot et al. (2005)), the FireWire root contention protocol (Kwiatkowska et al. (2003)), and the IPv4 Zeroconf protocol (Kwiatkowska et al. (2006)).

3.2 The Probabilistic Model Checker PRISM

In this paper, we use Continuous Stochastic Logic (CSL) (Baier et al. (1999); Aziz et al. (2000)) to specify requirements as properties. There are two types of formulae in CSL: state formulae, which are true or false in a specific state, and path formulae, which are true or false along a specific path.

Definition 2. Let $a \in AP$ be an atomic proposition, $p \in [0, 1]$ be a real number, $\bowtie \in \{\leq, <, >, \geq\}$ be a comparison operator, and $I \subseteq \mathbb{R}_{\geq 0}$ be a non-empty interval. The syntax of CSL formulas over the set of atomic propositions AP is defined inductively as follows:

- $true$ is a state-formula.
- Each $a \in AP$ is a state formula.
- If Φ and Ψ are state formulas, then so are $\neg\Phi$ and $\Phi \wedge \Psi$.
- If Φ is state formula, then so is $\mathcal{S}_{\bowtie p}(\Phi)$.
- If φ is a path formula, then $\mathcal{P}_{\bowtie p}(\varphi)$.
- If Φ and Ψ are state formulas, then $\mathcal{X}_I\Phi$ and $\Phi\mathcal{U}_I\Psi$ are path formulas.

In this paper, we use the PRISM probabilistic model checker developed by the work of Kwiatkowska et al. (2009). It supports the analysis of several types of probabilistic models: discrete-time Markov chains (DTMCs), continuous-time Markov chains (CTMCs) (Peng et al. (2014)), Markov decision processes (MDPs) (Lu et al. (2015)), probabilistic automata (PAs), and also probabilistic timed automata (PTAs), with optional extensions of costs and rewards. Moreover, PRISM allows us to verify properties specified in the temporal logics PCTL for DTMCs and MDPs and CSL for CTMCs. Models are described using the PRISM language, a simple, state-based language.

4. FORMAL MODELLING OF USVS WITH RESPECT TO A STATIC OBSTACLE

In this section, we present an illustrative case study of a single USV with respect to a static obstacle, analysed using PTAs and probabilistic model checking.

+1.2 7	+0.1 8	+0.2 9	 10
+1.3 4	+0.9 5		+0.2 6
+0.2 0	+0.4 1	+0.7 2	+1 3

Fig. 4. Fuel consumption in different area.

We use a case study to illustrate the role of PTAs for the analysis of USV if there is a static obstacle. In this case, the area is divided into 12 positions, from 0 to 10 (as shown in Figure 4. One of the areas is shaded, which denotes the position of a static obstacle expressing this position cannot be passed by the USV.

Assuming a USV starts to travel from position 0 to position 10, during this process, the USV operators need to evaluate the condition and environment of the positions, and decide the optimal path to the position 10. Because of the uncertainty of the navigation, guidance, and control, the USV does not always move to the intended direction. There is a probability of 20% that the USV will move to the wrong direction or remain in the same position. Furthermore, there is a probability of 10% of the deviation to the left position and similarly to the right position.

When the USV does not move according to the planned path, it will have to re-select the path of the route. For example, when the USV should move ahead to position 4 from position 0, it has 80% probability of completing the mission, 10% probability of remaining in position 0, and 10% probability of moving to position 1.

In this case, there are two other parameters: path following time and path following fuel cost. The time the USV spends in each position moving to each direction varies. The energy the USV uses in each position also varies. Figure 4 shows the fuel cost in each position. Table 1 shows the time spent by the USV to move between positions.

4.1 Real-time verification

Model checking using PRISM shows that the probability of the USV completing the path within 30 minutes is 0.84134. The USV will have 5.184% moving to the area 10 within 15 minutes, and 84.134% within 30 minutes. Figure 7

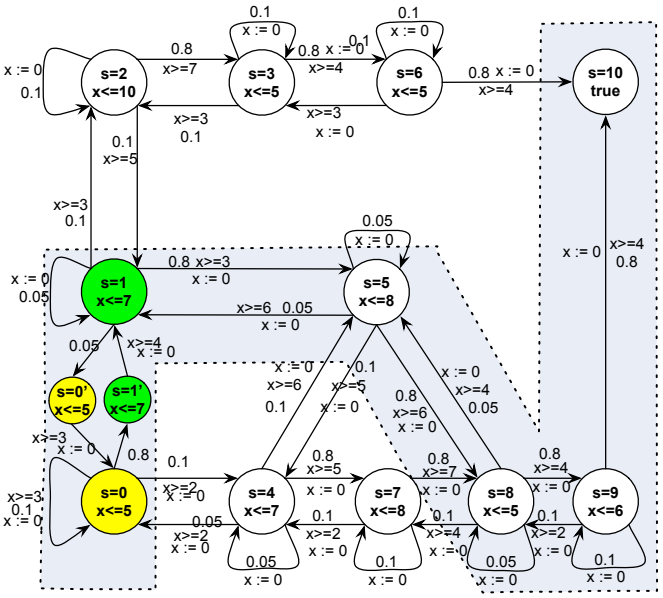

Fig. 5. Probabilistic timed automaton (PTA) w.r.t. minimum fuel consumption.

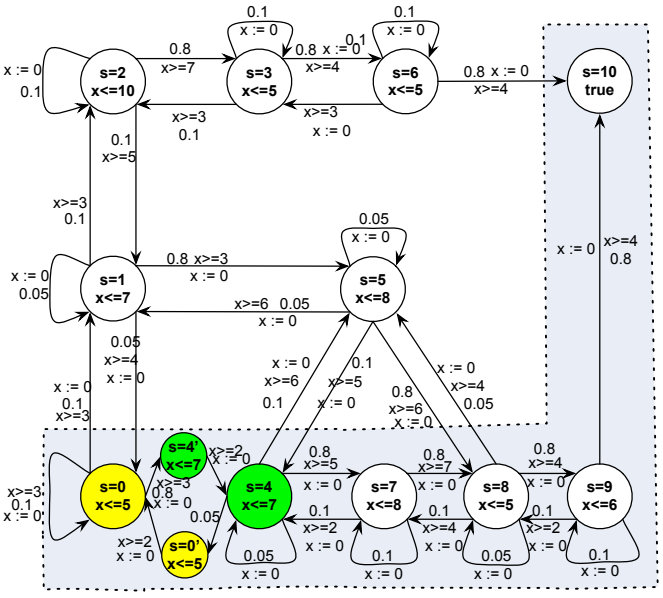

Fig. 6. Probabilistic timed automaton (PTA) w.r.t. minimum fuel path following time.

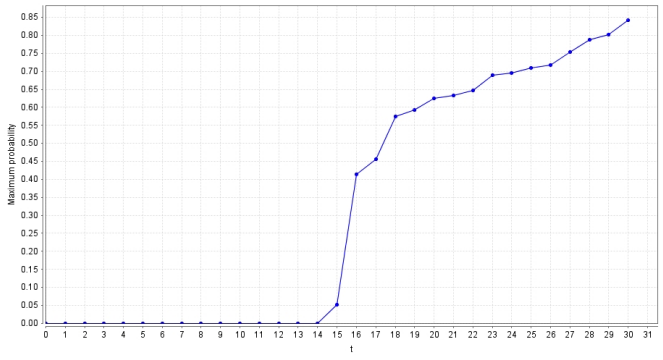

Fig. 7. The probability curve w.r.t. travel time of the USV.

Table 1. Time consumption of the USV moves to each position.

Direction	0	1	2	3	4	5	6	7	8	9
East	$3 \sim 5$	$5 \sim 6$	$7 \sim 8$	$3 \sim 5$	$6 \sim 7$	–	–	$7 \sim 8$	$5 \sim 5$	$2 \sim 6$
South	–	–	–	–	$4 \sim 7$	$6 \sim 8$	$3 \sim 5$	$5 \sim 8$	$4 \sim 8$	–
West	–	$4 \sim 6$	$7 \sim 8$	–	–	$5 \sim 8$	–	–	$4 \sim 5$	$4 \sim 6$
North	$2 \sim 5$	$3 \sim 6$	–	$4 \sim 5$	$5 \sim 7$	$6 \sim 8$	$4 \sim 5$	–	–	–

illustrates the relationship between the probability and time to complete the whole path.

We models a simple single USV scheduling in case of the Wenchuan earthquake. The USV starts in the initial position 0; after between 3 and 5 time units, the USV attempts to move to position 4:

- with a probability of 0.8, the USV starts to move to position 4;
- with a probability of 0.1, the USV stays in position 0;
- with a probability of 0.1, the USV starts to move to the position 1 instead.

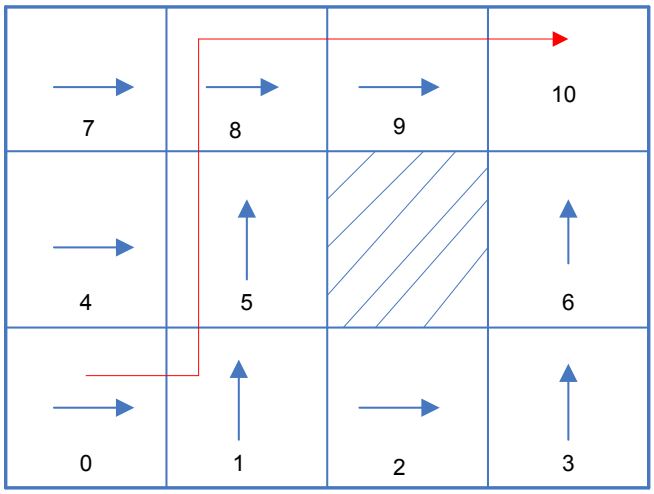

Fig. 8. Result of the USV to reach a given target: minimum energy consumption.

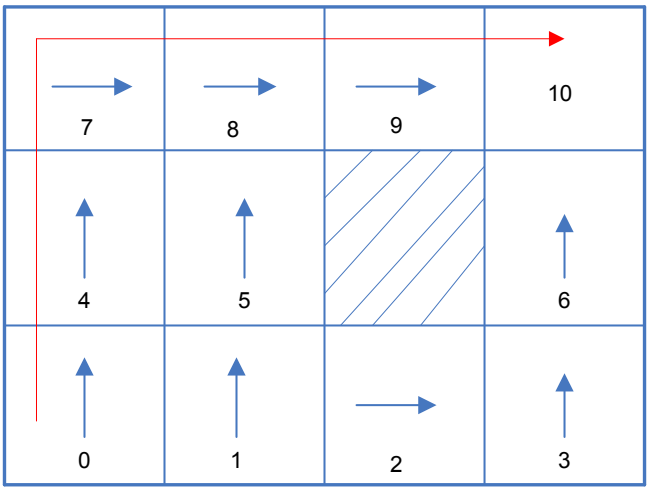

Fig. 9. Result of the USV to reach a given target: minimum time.

The minimum energy consumption requirement of the USV moving from position 0 to position 10 is 13.3. The route of the USV is as shown in Figure 8 for minimum energy consumption. The best route is: $0 \rightarrow 1 \rightarrow 5 \rightarrow 8 \rightarrow 9 \rightarrow 10$. If the direction of USV deviates, the USV should take the best route of each area as shown in Figure 8, for which the energy consumption is the least.

Similarly, the time the USV spends transferring from position 0 to position 10 is at least 22.8 minutes. The shortest path is shown in Figure 9. The best route is: $0 \rightarrow 4 \rightarrow 7 \rightarrow 8 \rightarrow 9 \rightarrow 10$. There is one best direction from each area. Once the USV enters into an area, the USV will take the least time if the USV takes the given path.

5. FORMAL MODELLING OF USVS WITH RESPECT TO A DYNAMIC OBSTACLE

In this section, we give the probabilistic timed automata (PTAs) models for both the obstacle and the USV with respect of three collision avoidance behaviours.

5.1 PTA Model of the Dynamic Obstacle

Assume that the dynamic obstacle moves along its path from the initial position O_S to the terminal position O_E. During the process of movement, it passes the positions 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 model can be constructed as shown in Fig. 10. According to the collision avoidance behaviours, we consider O_0-O_3 is the dangerous region. As a result, if the obstacle is in the state $i01$ or $i13$, the USV should make decision on a specific behaviour to avoid collision.

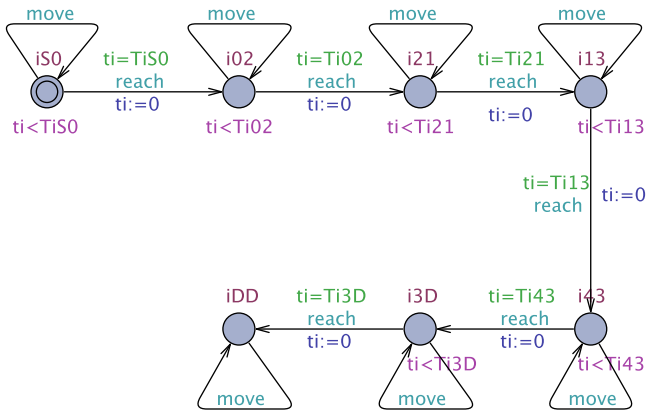

Fig. 10. The PTA model of the intruder.

5.2 The PTA model of the USV

In this subsection, we provide the formal representation for the state transitions of the USV (see Fig. 11, Fig. 12, and

Fig. 13). In the description, we assume that each collision avoidance behaviour can be performed by the USV with a certain probability.

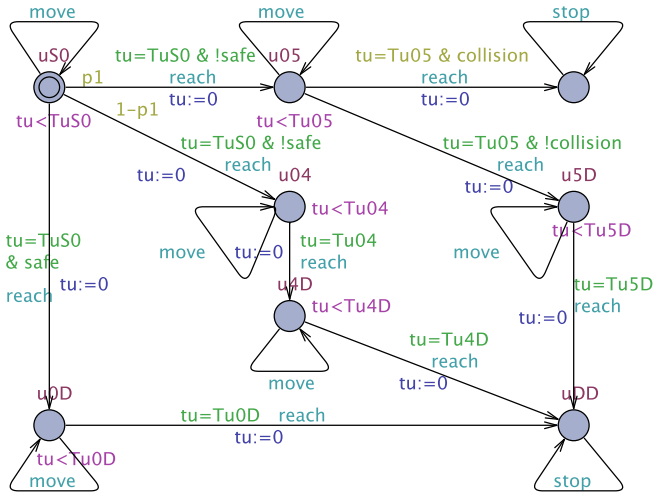

Fig. 11. The PTA model of the steering behaviour.

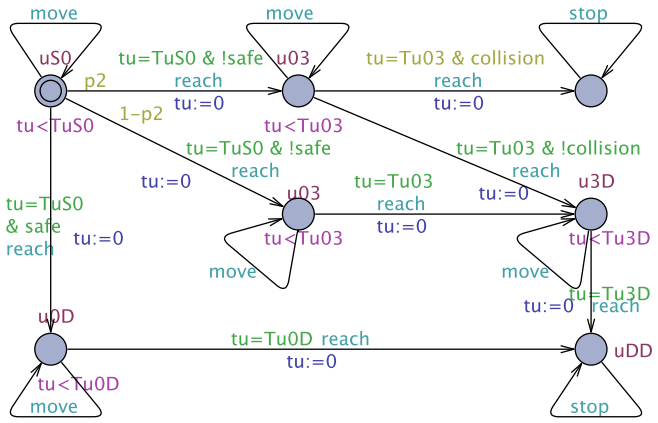

Fig. 12. The PTA model of the acceleration behaviour.

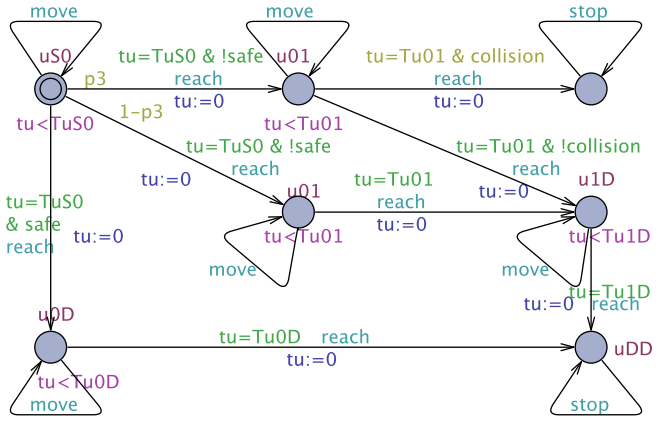

Fig. 13. PTA model of the deceleration behaviour.

6. CONCLUSIONS

In this paper, we have successfully used a case study to demonstrate how to use probabilistic model checking technique for verifying three collision avoidance behaviours of a USV in an uncertain dynamic environment. We first build a probabilistic timed automata based models of the underlying behaviours. Then, we apply the PRISM model checker to analyse the PTA models. We believe that this work can provide a foundation for the verification of complicated decision makings behaviours such as collision avoidance for USVs.

REFERENCES

Aziz, A., Sanwal, K., Singhal, V., and Brayton, R. (2000). Model-Checking Continuous-Time Markov Chains. *ACM Transactions on Computational Logic*, 1(1), 162–170.

Baier, C., Katoen, J.P., and Hermanns, H. (1999). Approximative Symbolic Model Checking of Continuous-Time Markov Chains. In *Proceedings of the 10th International Conference on Concurrency Theory (CONCUR 1999)*, LNCS, 146–161. Springer.

Duflot, M., Fribourg, L., Herault, T., Lassaigne, R., Magniette, F., Messika, S., Peyronnet, S., and Picaronny, C. (2005). Probabilistic Model Checking of the CSMA/CD Protocol Using PRISM and APMC. *Electronic Notes in Theoretical Computer Science*, 128(6), 195–214.

Fainekos, G.E., Kress-Gazit, H., and Pappas, G.J. (2005). Temporal Logic Motion Planning for Mobile Robots. In *Proceedings of ICRA 2005*, 2020–2025. IEEE.

Kwiatkowska, M., Norman, G., and Parker, D. (2009). PRISM: Probabilistic model checking for performance and reliability analysis. *ACM SIGMETRICS Performance Evaluation Review*, 36(4), 40–45.

Kwiatkowska, M., Norman, G., Parker, D., and Sproston, J. (2006). Performance analysis of probabilistic timed automata using digital clocks. *Formal Methods in System Design*, 29(1), 33–78.

Kwiatkowska, M., Norman, G., and Sproston, J. (2003). Probabilistic Model Checking of Deadline Properties in the IEEE 1394 FireWire Root Contention Protocol. *Formal Aspects of Computing*, 14(3), 295–318.

Lu, Y., Peng, Z., Miller, A., Zhao, T., and Johnson, C. (2015). How reliable is satellite navigation for aviation? checking availability properties with probabilistic verification. *Reliability Engineering & System Safety*, 144, 95–116.

Miura, J. and Shirai, Y. (2000). Modeling Motion Uncertainty of Moving Obstacles for Robot Motion Planning. In *Proceedings of ICRA 2000*, 2258–2263. IEEE.

Norman, G., Parker, D., and Sproston, J. (2013). Model checking for probabilistic timed automata. *Formal Methods in System Design*, 43(2), 164–190.

Peng, Z., Lu, Y., Miller, A., Johnson, C., and Zhao, T. (2014). Formal Specification and Quantitative Analysis of a Constellation of Navigation Satellites. *Quality and Reliability Engineering International*, 32(2), 345–361.

Quottrup, M.M., Bak, T., and Izadi-Zamanabadi, R. (2004). Multi-Robot Planning: A Timed Automata Approach. In *Proceedings of ICRA 2014*, volume 5, 4417–4422. IEEE.

Savvaris, A., Niu, H., Oh, H., and Tsourdos, A. (2014). Development of Collision Avoidance Algorithms for the C-Enduro USV. In *Proceedings of the 19th IFAC World Congress (IFAC 2014)*, 12174–12181. IFAC.