An Empirical Study of Flight Control System Model Checking Integrated with FMEA

Xinyi Wang
School of Reliability and Systems Engineering, Beihang University
Beijing, China
XinyiWang@buaa.edu.com

Gaolei Yi
School of Reliability and Systems Engineering, Beihang University
Beijing, China
ygl666@buaa.edu.cn

Yichen Wang
School of Reliability and Systems engineering, Beihang University Science and Technology on Reliability and Environmental Engineering Laboratory, Beihang University Beijing, China
wangyichen@buaa.edu.cn

***Abstract*—Cyber-Physical Systems (CPS) is a multi-dimensional complex system that integrates computing, network and physical environment. Flight control system enables the aircraft to interact with the outside world and other components, which is a typical CPS. The Verification of safety has been a research hotspot. Model checking is a formal verification method with a high degree of automation, including three steps: model construction, properties specifications and execution. In general, Properties has been generated based on the subjective experience of the verifier, and there is a lack of a strict process. We focus on the process of establishing specifications using Failure mode and effect analysis (FMEA) And then use our improved method to do the requirement level model checking about the return process and collision prevention based on PX4 flight control system. The results prove that our method makes the model checking logic clearer and has good adaptability.**

Keywords—Model Checking, FMEA, Safety verification, PX4

I. INTRODUCTION

With the improvement of development technology and software requirements, the complexity and the scale of software projects has increased, making it more difficult to ensure the safety of software projects. Manual testing cannot ensure that all possible input combinations are used to test the software, so the safety verification of the software is developing in a more automated and logically strict direction.

Model checking is a collection of techniques that analyze the consistency between the abstract representation of a system and the properties to be verified[1].Model checking is divided into three steps: models construction, properties specifications and execution.. If it is not satisfied, a counterexample is automatically generated. The degree of automation is high, so it is now well used in safety-critical software.

The model checking at the requirement level is the first step to ensure the safety of the system, and the effect of the verification result is directly related to the proposed properties specification. Most of the current research on model checking focuses on the system modeling and tools. The property specification is often directly proposed based on the subjective experience of researchers, lacking of a complete and systematic method to generate the property specification. Our research focuses on properties specification process at the requirement level model checking .

The requirement description often lacks the discussion of possible faults, and the danger occurs under the premise that there are certain fault modes. Therefore, the analysis and discussion of faults is an important aspect of ensuring safety. FMEA is a bottom-up software reliability analysis method by identifying software failure modes, analyzing the consequences, researching and analyzing the causes of various failure modes, looking for ways to eliminate and reduce their harmful consequences[2]. Therefore, we propose a model checking technique that is both demand-oriented and failure-oriented. The property specification part is derived through the FMEA process, and a timed automata model is established according to the property specification, and then verified in the uppaal tool.

This paper is organized as follows: Section II introduces the related works. Section III puts forward the process of establishing the property specification based on FMEA and integrating it into the model checking. In Section IV, empirical verification was carried out with the case of PX4's collision prevention function during the return mode. Section V are the conclusion and future research directions.

II. RELATED WORKS

Cyber-physical system (CPS) is increasingly used in automotive systems and drone systems. CPS requires operations within properly defined boundaries and strict control of error margins. [3] The safety of CPS is related to the users, which has created an urgent need for CPS safety verification. The current research hotspot is the formal automatic verification using appropriate modeling and simulation methods [4,5,6].

Dario Amodei [7] defines accidents as unexpected and harmful behaviors that may occur due to poor design of real-world systems. The safety problems of many systems today can be solved through temporary repair or case rules.

When considering safety verification, the failure mode is particularly important. Existing studies have proposed the integration of many failure-focused technologies. Atifi Meriem [8] uses the FMEA method in the model-based test method to automatically generate a set of {test case, priority} pairs to prioritize the test cases. Chen, L.[9] proposes an improved formal failure analysis approach for safety-critical system based on MBSA. S.Wang[10] uses model checking technology to combine safety requirement analysis with software function design, and proposed a real-time system safety verification method based on extended fault tree. This method realizes the system safety verification by verifying whether the safety attributes extracted from the time series fault tree are satisfied in the system model established by AADL.

Model checking is a formal verification method with a high degree of automation, which has important applications in safety-critical systems. We hope to establish a model

978-1-7281-8915-4/20/$31.00 ©2020 IEEE
DOI 10.1109/QRS-C51114.2020.00086

checking method that is both demand-oriented and fault-oriented, combining FMEA with model checking.

III. Model Checking with FMEA

This section introduces the process of proposing properties specifications use FMEA, and the model checking process based on FMEA.

A. FMEA

Failure mode and effects analysis (FMEA) is regarded as one of the formal techniques for risk-based testing. It provides a structured method to identify system failure modes, analyze their impact and establish control measures to reduce risks and improve the quality of the system or product. Traditional.[11] FMEA mainly includes the following steps:

- Determine the analysis level and objects;
- Determine the failure mode of the analysis object;
- Analysis of possible causes of failure modes;
- Failure impact analysis;
- Determine the severity (S), occurrence (O) and detectability (D) factors;
- Calculate the RPN value of each failure mode (product of the severity (S), incidence (O) and detection (D) factors;
- Formulate improvement measures based on RPN level.

there are two methods for determining the RPN level. One based on the size of the calculated value and the other equivalent partition based on the matrix. When determining the RPN level based on the calculated value, S, O, and D are at equally important positions. But in reality, the severity is a more important factor, so we use the matrix-based RPN equivalent division method. The values in the matrix are the RPN levels determined based on the values of S, O, and D. The evaluation criteria for the values are shown in Fig 1.[8]

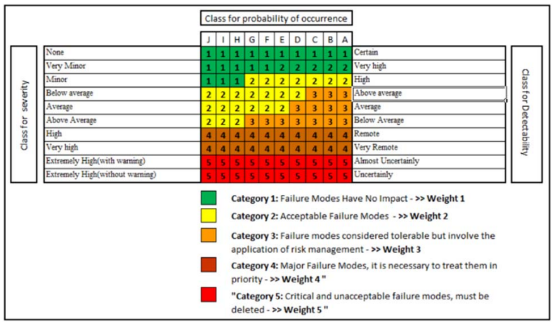

Fig. 1 Probability-severity-detectability Matrix

B. Improved FMEA for Model Checking

The FMEA we introduced is used as an analysis method of the property specification to serve the model checking process. The resulting property specification is considered to be the most important content of the system to be tested. In order to simplify the system, system behavior not related to the property specification should not be considered. Therefore, the following aspects of the FMEA process are improved.

- The first step of traditional model checking is modeling, but we propose that properties specification analysis by FMEA is carried out first, and then based on the specification to model system, which can simplify the model.
- Traditional FMEA determines the failure mode before failure cause analysis We propose conducting unsafe cause analysis first, and the failure mode is derived from the cause.
- Change "Development of improvement measures" to "Development of properties to be inspected". After obtaining the failure mode, what requirements does the system need to meet in order to avoid failure. Take this as the property to be detected.
- Combining RPN value with impact analysis, RPN value is also used as part of the impact, and the priority of the properties specification is sorted based on the RPN value.

C. Process of Model Checking with FMEA

The improved model checking process is shown in the Fig.2

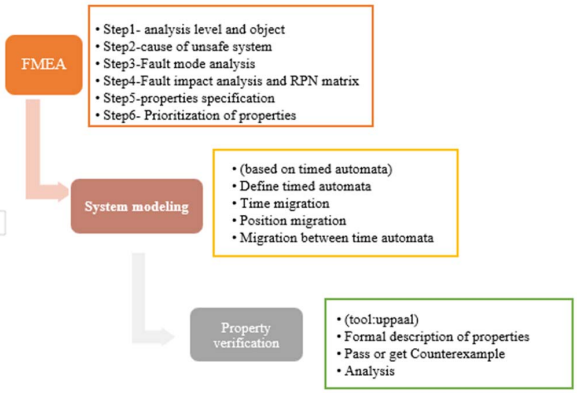

Fig.2 Model Checking process

1) Establish a property specification based on FMEA

- Step1- Determine the analysis level and object;
- Step2-Determine the cause of unsafe system;
- Step3-Fault mode analysis based on cause;
- Step4-Fault impact analysis and RPN matrix;
- Step5-Determine the properties specification of the test to avoid failure;
- Step6- Prioritization of properties based on RPN matrix.

Step1-3 is the basis of FMEA analysis, and Step4-6 can be directly obtained by establishing FMEA table based on Step1-3 analysis. In this article, the expression of establishing a table is adopted.

2) System modeling based on timed automata

It has been pointed out that the practical goal of verification of real-time systems is to verify simple logical properties, which does not need the whole power of model-checking. We shall only consider simple safety properties, which can be verified without constructing the whole reachability graph of a timed system.

i) Grammar of timed automata

A timed automata is a finite state machine with clock variables, usually defined as a six-tuple （ L,I_0,A,V,I,E） where:

L: finite position set; I_0: initial position set; A: finite action set; V: finite variable set;

ii) Time Semantic Network Semantics

Let M（L,I_0,A,V,I,E） be a timed automata which consists of a timed network. Channels are used to achieve synchronous communication. Suppose c is used to denote a channel, then c? represents an event received from another timed automata, and c! represents an event sent to another timea automata. There are three types of state transitions in the timed automata network:[12]

- Time migration: $(\bar{I},u)\rightarrow(\bar{I},u+d)$,which,$d\in R^{+}$,if $\forall d'\leq d$, then u and u+d satisfy $I(\bar{I})$;
- Position migration: $(\bar{I},u)\rightarrow(\bar{I},[I_i'/I_i],u')$, if $e(I_i,a,g,r,I_i')\in E,u'=r(u)$, then u satisfy g, u' satisfy $I(\bar{I})$;
- Migration between timed automata: $(\bar{I},u)\rightarrow\left(\bar{I},\left[I_i'/I_i,I_j'/I_j\right],u'\right)$, If there is an edge,$I_i\xrightarrow{a?,g,r,}I_i'$ and an edge $I_j\xrightarrow{a!,g,r,}I_j'$, then $u'=r_i\cup r_i(u)$, which u satisfy g, u' satisfy $I(\bar{I})$.

3) Property verification and result analysis

We use the model test tool uppaal to verify the timed automata based on the property specification. The description of the attribute formula uses a simplified version of the calculation tree logic (CTL) formula, which consists of two parts: the state formula and the path formula. The path formula of the tested properties is shown in TABLE 1.[13]

TABLE I. PATH FORMULA

Path Formula	Description
A [] p	For all states of all paths, p is satisfied
A <> p	For all paths, p finally meets
E [] p	There is a path with all states p satisfying
E <> p	There is a path, p finally meets
p imply q	If p satisfies, then q can also be satisfied

IV. CASE STUDY

A. FMEA Analysis of the Collision Prevention Feature of the Flight Control System PX4 during Return Mode

1) Requirement Statements

Return flight is an operation that occurs when the aircraft itself or external conditions change and the flight cannot continue as expected. It is to ensure safe operation under certain dangerous conditions and is a part closely related to safety during flight. The route at the time of return is not in the plan before the voyage. The biggest external disturbance during flight is collision, including collision with mountains or collision with birds. [14]PX4 uses the following two types of measures to prevent collision and achieve a safe return flight:

a) Set Return Altitude

The aircraft should be adjusted to a safe altitude before returning to the home to prevent collision with objects at altitude above the destination.

Use the parameters RTL_RETURN_ALT and RTL_CONE_ANG to define a half cone centered on the target position, as shown in picture 2, to configure the return height, and consider the conical area as a safe area. According to the difference between the initial position and the set value of the aircraft before returning to the home, we divide it into three set safety ranges according to the difference between the cone range and the starting position, divided into {ang=0, ang=90°, ang=(0, 90°)}. The combination of altitude control and collision prevention control during the return flight is that the safety ranges defined in the altitude control before returning to the flight are different. The three safety ranges are as follows[15]:

- If RTL_CONE_ANG is 90 degrees the vehicle will return at the greater of RTL_DESCEND_ALT and the current altitude.
- If RTL_CONE_ANG is 0 degrees there is no "cone": the vehicle returns at RTL_RETURN_ALT (or above)
- If the cone angle is between 0 and 90 degrees, the action corresponding to the starting position is to return home with the higher of the current altitude and RTL_RETURN_ALT

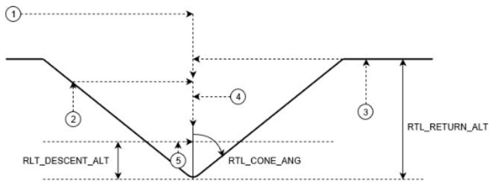

Fig. 2 Return height adjustment

The combination of altitude control and collision prevention control in the return process is that the safety range defined in altitude control before returning home is different, and the requirements for collision prevention are also high when the safety range is small, so we are instantiated in the aircraft template to meet the three safety ranges s plane. Assuming that RTL_TYPE=0 mode is used, the overall process is to ascend to a safe return height above any expected obstacle, fly to the starting position or meeting point through a direct path, and hover at the descending height. The aircraft can compensate for wind and other forces in the current direction and move parallel to the ground, as shown in Fig 3.[15] Therefore, when analyzing the return process, the situation of roll, pitch, and yaw is not considered. By default, after the return height adjustment, the aircraft flies horizontally parallel to the ground, and then descends until it reaches the destination, and finally hangs over the destination.

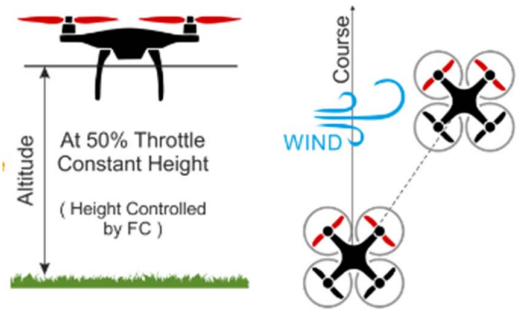

Fig. 3 Airplane flight attitude

b) Collision Prevetion

PX4 is designed with collision prevention function to avoid obstacles in the environment. Since the return altitude adjustment has made the aircraft enter a safe area without static obstacles, we use the collision prevention function to verify that when encountering obstacles with initial speed such as birds There are two ways to deal with the collision prevention function of objects:

- When the aircraft detects an obstacle, it starts to brake, and the speed gradually decreases toward the obstacle. The speed decreases to zero before reaching the minimum allowed approach distance.
- If the aircraft crosses the minimum allowed approach distance due to overshoot or external force, reverse thrust will be activated to keep the aircraft away from obstacles.

2) FMEA-based property specification establishment

a) Step 1: Determine the analysis level and objects.

The analysis level is the system requirement level, and the safety verification of the overall return process and collision prevention function after the return altitude adjustment is performed, and the aircraft speed, flight altitude, and obstacles are analyzed objects. Focus on the interaction between the three research objects.

b) Step 2: Determine the cause of system insafety

Our research object is the flight control software PX4, which is suitable for unmanned aerial vehicles. The return flight is a sub-mode in automata mode, and the flight control is based on the PID algorithm. We pay attention to the defects of the design and the interaction results with the environment. We do not pay attention to the learning process. We believe that the algorithm of the learning process achieves the expected goals. Consider the unsafe reasons from the following aspects:

- Negative side effects: If you only focus on a specific task in the environment, and other tasks do not change the task mode to respond in time after interacting with this specific task, it may have a negative impact.
- Parameter setting range: Collision prevention is the main focus of our safety analysis. Our goal is not to detect whether or not to achieve collision prevention under a specific input, but to verify the range of parameter values that can achieve collision prevention.

c) Step 3: Analysis of failure modes based on unsafe causes.

We pay attention to the impact of the following highly changed specific processes interacting with collision prevention tasks:

i) Negative side effects

- Failure Mode 1:collision prevention failure when RTL_CONE_ANG is 0 degrees.

  Analysis: When RTL_CONE_ANG is 0 degrees, it means that the range of the aircraft is more dangerous, there is no safe cone range, there are obstacles with altitude above the ground, if the aircraft encounters movable obstacles during the process of returning to RTL_DESCEND_ALT, if the stationary obstacles are not recognized in time Objects and activate the collision prevention function again, it is possible to hit the stationary obstacles around.

ii) Parameter range

- Failure Mode 2: The plane and the obstacle fly relatively, and the plane collides during deceleration.

  Analysis: When the collision prevention function is activated during flight, if an obstacle is flying relative to the aircraft, the aircraft needs to be decelerated in time. If the deceleration is not timely, the obstacle may be hit by the minimum allowed approach distance during the deceleration.

- Failure Mode 3：The aircraft was hit by an obstacle while accelerating in the reverse direction.

  Analysis: Reverse acceleration is an operation performed when the distance between the aircraft and the obstacle is too small or the aircraft's own speed is already too low. If the acceleration is not timely, the collision will occur when the obstacle speed is high

d) Step 4: FMEA Table

TABLE II. FMEA FORM OF COLLISION PREVENTION FUNCTION OF FLIGHT CONTROL SYSTEM PX4 DURING RETURN FLI

Unsafe Reason	Failure Mode	Failure Effects	Severity Factor(S)	Occurrence Factor(O)	Detectability factor(D)	RPN	Property to be verified	Category
Negative side effects	Collision prevention failure when RTL_CONE_ANG is 0 degrees	The surrounding is more dangerous and hits the surrounding stationary objects	Below Average	E	High	2	When RTL_CONE_ANG is 0 degrees, collision prevention can be achieved in all flight phases.	3
Parameter range	The plane and the obstacle fly relatively, and the plane collides during deceleration	The plane collided with an obstacle, but at this time the plane was slower	Average	C	Average	3	When the deceleration is 0, the distance between the aircraft and the obstacle is greater than the safe distance	2
	The aircraft was hit by an obstacle while accelerating in the reverse direction	The plane collided with an obstacle, and the speed of the plane was high	Above Average	D	Above Average	3	The plane can accelerate faster than the obstacle after reverse acceleration	1

Note: The RPN level is the same, the severity factor takes precedence

B. Timed automata model based on FMEA property specification

Through FMEA analysis, we have obtained the safety properties of the system, and the focus is on the following aspects: 1) According to the initial set cone angle, there are three corresponding initial safety ranges. 2) When it is in the most dangerous safety range, collision avoidance can be achieved in all flight conditions of the return flight. 3) When the aircraft and obstacles fly relative to each other, they need to go through two processes of deceleration and acceleration. 4) No collision during deceleration. 5) The speed of the aircraft after acceleration should not be less than the speed of the obstacle.

On the premise of satisfying the above five aspects, we establish the following three models:

- The Vehicle_Alt model, simulating the three initial safety ranges and also increases the state space, making the model test more meaningful.
- Vehicle_Speed model, simulating the speed of acceleration and deceleration of aircraft.
- The Obstacle model, simulating the appearance of obstacles and the process of approaching.

The communication between the models is represented by channels, and the transfer process is as described above. The communication channels included in the model we built are shown in TABLE 3:

1) Vehicle_Alt automata model

Take the unsafe state as the starting state, and the transition condition is StartAltitude! StartAltitude={ StartAltitude 1, StartAltitude 2, StartAltitude 3}, that is, the height adjustment before the return flight is performed in different safety ranges, assuming that the safe area is reached after the corresponding conversion is performed under the corresponding initial conditions, expressed as Safe_Area. Since the return flight is in Position Mode, the flight trajectory of the aircraft in Position Mode is parallel ground or vertical ground, regardless of the pitch, roll, and yaw conditions. In order to simplify the system, we stipulate that the flight path of the aircraft is to fly parallel to the ground first, to land on the vertical ground when reaching the destination, and to reach the hovering state when landing to RTL_DESCEND_ALT. Parallel flight is represented by the Horizontal_Fly state, and vertical ground flight is represented by the Vertical_Fly state. Suspend deactivated Loiter said. The template is shown in Fig 4. When an obstacle appears in a certain flight state, the obstacle_coming message is received, staying in the state at this time, when a collision is completed, and the current state is received when the Success_Vehicle_Speed model is received, the StartAltitude message is issued again. In order to show that you can continue to fly within the set safety range.

TABLE III. COMMUNICATION CHANNEL DESCRIPTION

Name	Function	Emitting end	Receiving end
StartAltitude	Represents three different initial safety ranges.	Vehicle_Alt	Vehicle_Speed
obstacle_coming	On behalf of the obstacles began to approach the aircraft, and start collision prevention.	Obstacle	Vehicle_Speed, Vehicle_Alt
success	On behalf of the completion of a collision prevention process.	Vehicle_Speed	Vehicle_Alt

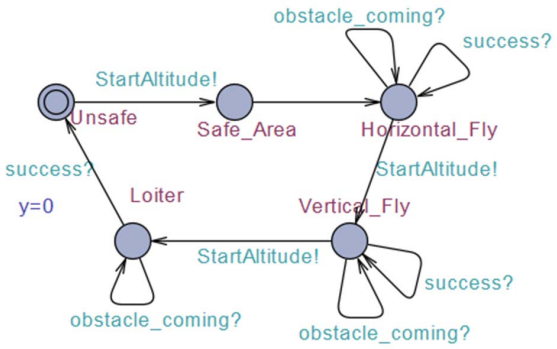

Fig. 4 Vehicle_Alt timed automata model

2) Vehicle_Speed timed automata model

Based on the nature specification that needs to be verified, we set the following collision avoidance scenarios and conditions: the obstacle moves towards the aircraft, assuming that the initial speed of the aircraft is 4m/s, when the sensor detects an obstacle, the collision avoidance function is started, assuming safety The distance is m, the obstacle moves at a constant speed, the aircraft decelerates under the action of the reverse acceleration a, and the direction of a reverses the initial speed of the aircraft until the deceleration is 0, and the distance is greater than the minimum allowed approach distance. Then it still accelerates in the opposite direction with the same acceleration until it reaches the same speed as the obstacle, and can fly in parallel and in the same direction as the obstacle. At this time, in order to achieve a collision avoidance, the process is shown in Fig 5.

Assume that only one Obstacle appears during the entire deceleration process. The initial state is Follow_preplanned_path, which indicates normal flight along the preset trajectory. StartAltitude indicates the safety range of the aircraft at this time. When an obstacle in the Obstacle model appears, the obstacle_coming message is received and deceleration begins. At this time, the system time Update to 0 to make each collision prevention process independent and set activated to true to prevent a new Obstacle from appearing. The initial distance between the aircraft and the obstacle is S, S1 is the distance traveled by the aircraft during the deceleration phase, S2 is the distance traveled by the aircraft and the obstacle during the deceleration phase, S3 is the distance traveled by the obstacle during the acceleration phase, and S4 is the distance traveled by the aircraft during the acceleration phase. According to the kinematics formula, the relationship that x, s1 and s2 need to meet is updated before entering safe_check. If the distance between the aircraft and the obstacle and the minimum allowed approach distance m is less than the initial distance, it proves that it is safe between the two when the deceleration is 0. When the above conditions are met, it can pass through Safe_check, enter Accelerate, and update according to the kinematics formula S3 and S4 movement status, S4+S-S2-S3>=m, it means that the aircraft is still safe when it accelerates to the same speed as the obstacle. When the above conditions are met, the transition from Accelerate to Move_Along_With_Obs is performed. At this point, an collision prevention is completed, a success message is sent, and the system time is updated, the acceleration is changed, and the next simulation is prepared.

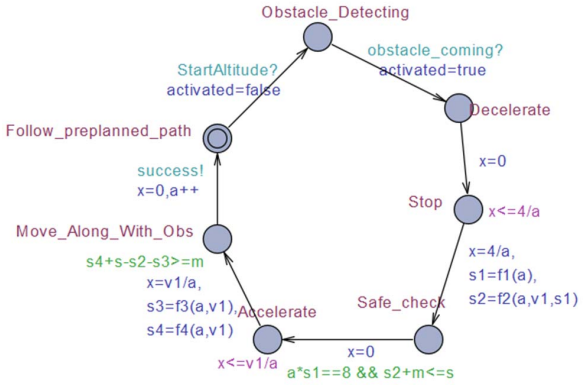

Fig. 5 Vehicle_Speed timed automata model

3) Obstacle timed automata model

This model is used to simulate the activities of obstacles, so that the appearance of obstacles can be signaled to notify Vehicle_Speed that there are obstacles, and then start the corresponding speed change process of Vehicle_Speed. The process starts at the position Start. When !activated is true, it moves to the Get_close position, and then triggers obstacle_coming! to notify Vehicle_Speed that an obstacle is approaching it, then returns to the Start position, and waits until !activeted is true to trigger again. The role of activated is to prevent multiple Obstacles from appearing in a collision avoidance process. The model is shown in Fig 6.

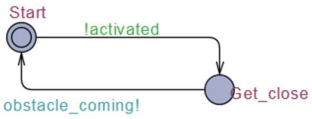

Fig. 6 Obstacle timed automata model

C. Property specification verification

We analyzed the unsafe causes of the system, the failure modes based on unsafe causes, and the impact of failures in the failure mode through FMEA, and obtained the safety properties that the system needs to meet. We modeled the automata around the nature, and the nature is also in the automata. A more specific description has been achieved. The properties based on FMEA and the performance in the automata are recorded in TABLE 4. In addition, in order to verify that our model is implemented correctly, we can perform exhaustion of the state space, so we add tests to the properties Whether the model is deadlocked, the verification results obtained after performing verification are shown in TABLE 4 and Fig 7.

```
A[] not deadlock                    (Verification/Kernel/Total)
验证费时/kernel费时/总费时: 34.844s / 0.047s / 34.929s. (Time)
常驻内存/虚拟内存的使用峰值: 174,380KB / 360,920KB. (Resource Usage)
满足该性质. (Property Satisfied)
A<>Vehicle_Speed1.Accelerate imply Vehicle_Speed1.Move_Along_With_Obs
验证费时/kernel费时/总费时: 0.593s / 0.016s / 0.608s.
常驻内存/虚拟内存的使用峰值: 174,296KB / 360,608KB.
满足该性质.
A<>Vehicle_Speed1.Decelerate imply Vehicle_Speed1.Accelerate
验证费时/kernel费时/总费时: 0s / 0s / 0.002s.
常驻内存/虚拟内存的使用峰值: 13,964KB / 37,020KB.
满足该性质.
A<>Vehicle_Alt2.Safe_Area imply Vehicle_Alt2.Unsafe
验证费时/kernel费时/总费时: 0s / 0s / 0.002s.
常驻内存/虚拟内存的使用峰值: 13,976KB / 37,020KB.
满足该性质.
```

Fig. 7 Property verification results

TABLE IV. STATISTICAL TABLE OF VERIFICATION RESULTS OF SPECIFICATION

Property specification based on FMEA	Representation of properties in automata	Formal description language	Verification results
None	The model created will not deadlock	A[] not deadlock	Satisfied
Collision Prevetion can be achieved in all flight phases when RTL_CONE _ANG is 0 degrees	Every flight phase in Vehicle_Alt2 can achieve collision avoidance, that is, any path from Safe_Area can eventually return to the initial state.	A <> Vehicle_Alt2. Safe _Area imply Vehicle_ Alt2. Unsafe	Satisfied
When the deceleration is 0, the distance between the aircraft and the obstacle is greater than the minimum allowed approach distance	Any path in Vehicle_Speed starting from Decelerate satisfies the conversion condition to Accelerate (the conversion condition is: S2+m<=S)	A<>Vehicle_ Speed1.Dec elerate imply Vehicle _Speed1 .Accelerate	Satisfied
The speed of the aircraft after reverse acceleration can be greater than the speed of the obstacle	Any path in Vehicle_Speed starting from Accelerate satisfies the conversion condition to Move_Along_Wit h_Obs (the guard condition is: s4+s-s2-s3>=m)	A<>Vehicle_ Speed1.Accel erate imply Vehicle_Spee d1. Move_Along _With_Obs	Satisfied

V. CONCLUSION AND FUTURE WORKS

After analyzing the process framework of model detection, we propose that the current research lacks exploration of the process of establishing the property specification, so we pay attention to the process of establishing the property specification. It is proposed to use FMEA technology to establish the property specification and improve it to make FMEA and model detection fusion. Through the steps of "Analysis of Unsafe Reasons", "Analysis of Failure Modes", "Analysis of Failure Effects", "Proposal of Nature Specification", "Determining Nature Priority", the nature specification based on FMEA was obtained. Then the timed automata model of the system is established based on the property specification, and this method can also simplify the system. Finally, perform verification in the tool. We verified the process we proposed to the collision prevention function during the return flight of the PX4 flight control system design. The results prove that our verification method has found a reasonable source for the nature specification, and the process is rigorous and highly applicable. Our future works are as follows

A. Add environment modeling

We analyzed the effects of the speed of the aircraft, the return process of the aircraft, and the interaction between obstacles, provided that the aircraft only paralleled the ground and lifted vertically, and lacked a description of the external environment. In the subsequent improvement, we consider designing the environment model to make the system have a more realistic simulation effect.

B. Increase analysis of smart components

The realization of aircraft collision prevention function also depends on the visual perception component. The visual perception component is needed to find obstacles and pass them to the control center. In subsequent studies, we consider adding analysis of visual perception components to make the system have intelligent behavior.

REFERENCES

[1] H .Ogawa , M .Ichii , F. Kumeno , “A Practical Study of Debugging Using Model Checking”. Software Engineering Conference. IEEE, 2014

[2] P.J.R. Torres, E.I.S. Mercado, L.A. Rifon, “Probabilistic Boolean network modeling and model checking as an approach for DFMEA for manufacturing systems”, Journal of Intelligent Manufacturing, 29 (2018) 1393-1413.

[3] H.-D. Tran, F. Cai, M.L. Diego, P. Musau, T.T. Johnson, X. Koutsoukos, “Safety Verification of Cyber-Physical Systems with Reinforcement Learning Control,” ACM Transactions on Embedded Computing Systems, 18 (2019) 1-22.

[4] C.Erkan Tuncali, J.Kapinski, H. Ito, and J.V.Deshmukh. “Reasoning about Safety of Learning-Enabled Components in Autonomous Cyber-physical Systems”. CoRR, abs/1804.03973, 2018.

[5] D. Tommaso , F.Daniel, G.Shromona, K. Edward, R. Hadi, V.Marcell ,S. Sanjit. (2019). “VERIFAI: A Toolkit for the Design and Analysis of Artificial Intelligence-Based Systems.”

[6] F. Mhenni, N. Nguyen, J.Y. Choley, “SafeSysE: A Safety Analysis Integration in Systems Engineering Approach,” Ieee Systems Journal, 12 (2018).

[7] D.Amodei, C.Olah, J.Steinhardt, P.Christiano, “Concrete Problems in AI Safety”a rXiv:1606.06565v2 [cs.AI] 25 Jul 2016

[8] A.Meriem and M.Abdelaziz, “Combining Model-Based Testing and Failure Modes and Effects Analysis for Test Case Prioritization: A Software Testing Approach”. Journal of Computer Science 2019, 15 (4): 435.449

[9] L. Chen, J. Jiao, Q.X. Wei, T.D. Zhao, “An improved formal failure analysis approach for safety-critical system based on MBSA”, Engineering Failure Analysis, 82 (2017) 713-725.

[10] S.Wang Siqi. “ Research on real-time system safety verification method based on sequential fault tree”. (Doctoral dissertation). (王思琪. (2016). 基于时序故障树的实时系统安全性验证方法研究. (Doctoral dissertation).)

[11] Abdelkaderbouti, D. Aitkadi, “A state-of-the-art review of FMEA/ FMECA”. International Journal of Reliability Quality and Safety Engineering, 2012, 01(4).

[12] Gu R , Enoiu E , Seceleanu C . TAMAA: UPPAAL-based mission planning for autonomous agents[C]// SAC '20: The 35th ACM/SIGAPP Symposium on Applied Computing. ACM, 2020.

[13] J .Bengtsson, K G .Larsen , F .Larsson , “UPPAAL - a Tool Suite for Automatic Verification of Real-Time Systems”, Hybrid Systems Iii: Verification & Control, Dimacs/sycon Workshop, October, Ruttgers University, New Brunswick, Nj, Usa. Springer-Verlag New York, Inc. 1995.

[14] H.Anand, Z.Chen , S.Bearman, “The OpenUAV Swarm Simulation Testbed: a Collaborative Design Studio for Field Robotics”. 2019.

[15] .PX4 UserGuide. https://docs.px4.cc/master/zh/index.html,2020.