Parametric Statistical Model Checking of UAV Flight Plan*

Ran Bao[1,2], Christian Attiogbe[2][0000−0002−7815−1752], Benoît Delahaye[2][0000−0002−9104−4361], Paulin Fournier[2], and Didier Lime[3]

[1] PIXIEL GROUP, Nantes, France https://www.pixiel-group.com/
[2] Université de Nantes - LS2N UMR CNRS 6004, Nantes, France
[3] Centrale Nantes - LS2N UMR CNRS 6004, Nantes, France

Abstract. Unmanned Aerial Vehicles (UAV) are now widespread in our society and are often used in a context where they can put people at risk. Studying their reliability, in particular in the context of flight above a crowd, thus becomes a necessity. In this paper, we study the modeling and analysis of UAV in the context of their flight plan. To this purpose, we build a parametric probabilistic model of the UAV and use it, as well as a given flight plan, in order to model its trajectory. This model takes into account parameters such as potential filter or sensor (like GPS) failure as well as wind force and direction. Because of the nature and complexity of the successive obtained models, their exact verification using tools such as PRISM or PARAM is impossible. We therefore develop a new approximation method, called Parametric Statistical Model Checking, in order to compute failure probabilities. This method has been implemented in a prototype tool, which we use to resolve complex issues in a practical case study.

Keywords: UAV · Formal Model · Markov Chain · Parametric statistical model checking

1 Introduction

Unmanned Aerial Vehicles (UAV) are more and more present in our lives through entertainment or industrial activities. They can be dangerous for their environment, for instance in case of a failure when an UAV (aka a drone) is flying above a crowd. Unfortunately until today, there does not exist any kind of UAV regulation around the world. Only some recommendations are used; for instance in order to avoid accidents in case of malfunctioning, a drone should never fly above a crowd.

In this context, we are working with PIXIEL group to build a reliable UAV control system. PIXIEL group is a company expert in safety drones and public performances including UAVs. For example, PIXIEL is in particular known for developing a public performance in the French entertainment park called "Puy

* Supported by PIXIEL and Association Nationale Recherche Technologie (ANRT)

du Fou" that includes both human actors and drones. The company is strongly attached to the safety of the public. Therefore, ensuring that its UAV systems are secure for humans during the performances is a priority. As for the current practices, the performances including UAVs are only allowed to occur when the weather is sunny and when the area above which the UAVs fly is unauthorized for actors and public. However, there is no certification proving that the UAVs always follow their intended flight plan.

The management of performances indeed requires to pay close attention to the drone trajectory computation as well as to the accuracy of the measurements concerning its immediate position in space and its movements. However, a rigorous study is necessary to ensure reliability of the drone control system, for instance by decreasing the risks of failure using the appropriate tuning of the drone flying parameters which impact the computation of its trajectory. Accordingly, the questions are *how to prove that the UAV failure probability is low and which parameters have to be taken into account to ensure human safety during performances including UAVs.*

High-quality aircrafts such as Hexarotors can easily avoid the majority of minor failures related to hardware because they can fly with only five motors and the probability of concurrent failure of more than two motors is in general negligible. In the same way, in case of battery failure, the UAV is able to land down on a specified area without any safety issue for the environment as long as it is situated in a safe zone where humans are not endangered. However, software failure may be a lot more problematic and complex to study. In this case, the UAV behavior might become unpredictable. One critical issue in this context is the potential inaccuracy of position estimation in drone systems, either as a result of inaccurate sensor measurements or of misinterpretation of data coming from those sensors. Besides aircraft system failure consideration, there is also a far more critical aspect to take into account: the weather environment. Therefore, a general approach to improve UAV safety is to study the impact of inaccuracy in position measurements on the resulting flight path compared to a given, fixed, flight plan while taking into account weather conditions.

There are many works dedicated to the UAV domain. In [20] Koppány Máthé and Lucian Buşoniu basically explain the functioning of a drone. UAV movement recognition is studied in [10]. Automatic landing on target is described in [17] and monitoring and conservation are dealt with in [11]. Some works also try to detect breakdowns and malfunctions that can impact drones. We can mention *inter alia*, the detection of communication errors in a multi-drone framework studied in [13] or the development of a basic diagnosis model for solving system issues in [9]. Our work is closer to this second category of topics. However, to the best of our knowledge, there are no existing works on the parametric study of the impact of component inaccuracy on UAV trajectory. In [5,24] the authors study through the *secure estimation problems* how to estimate the true states of an UAV system when the measurements from sensors are corrupted, for instance by attackers. In their work, these authors reformulate the estimation problem into the error correction problem and then they use the successively observed

measurement anomalies to reconstruct the correct states of the system. While the used techniques are completely different, the objectives of avoiding bad states is similar to ours, in avoiding the states reaching bad security zones.

The purpose of our work is therefore to provide means to study the reliability of UAVs in the context of a given flight plan. In order to do that, we have to build a formal (mathematical) model which will allow us (1) to analyze the drone system and detect the most important parameters, and (2) to tune those parameters in order to reduce the system failure probability. To this intent, we thoroughly study the UAV system, formalize it and analyze it with using parametric probabilistic methods. Among the components of a drone system, we particularly focus on the *Flight Control System* (FCS), which is responsible for computing estimations of the UAV position during its flight in order to adapt its trajectory to a given predefined flight plan. We therefore build a formal model of the flight controler in terms of parametric probabilistic models that takes into account the potential inaccuracy of the position estimation. Since UAVs are particularly sensitive to the weather environment (and in particular to wind conditions), we also enhance our model in order to take into account potential wind perturbations. Since wind force can drastically vary from one point of a given flight plan to another, we also use parameters to encode the wind force and allow our model to adapt to particular weather conditions.

The contributions of this paper are:

- a method to build a parametric model of UAV systems; the parameters can then be finely tuned until reaching values that ensure defined safety thresholds;
- a parametric statistical model checking technique; this enables us to formally analyze the parametric models build for the drones. Indeed because of the complexity of the built models, tools such as PRISM [15,16] and PARAM [12] were limited for their analysis.
- an illustration of the use of our method on a complex industrial case study.

The paper is organized as follows. In Section 2 we provide the essential background to understand UAV functioning and then we build a formal model that support their behaviours. Section 3 is an introduction to parametric Markov chains and Statistical Model Checking. Implementations of the models and experimentations are presented in Section 4; finally Section 5 draws conclusions and further work.

2 Building a Formal Model of UAV

In this section, we present our method to build the UAV model. Recall that we are interested in studying UAV safety, i.e. studying the probability that a UAV encounters dangerous situations. These situations are of two kinds: either the UAV can stop flying and fall, or it can enter a "forbidden" zone were it endangers humans. As explained earlier, professional UAV can handle the falling risk through material redundancy. Moreover, as long as a UAV stays in a "safe"

zone, it will not endanger human even in case of falling. The aim of our model is therefore to evaluate the probability for a UAV to enter a "forbidden" zone.

We start by explaining how the zones are computed with respect to the given flight plan. We then show how the UAV software can be decomposed into components and focus on the most important ones. Finally, we detail how the formal (mathematical) models for the important components are built and present the resulting global model.

2.1 Safety zones

In the context of software, considerations in airborne systems and equipment certification (named DO-178C) defined five levels of safety zones, the most secure being Zone 1 and the most dangerous being Zone 5. These zones are characterized by their distance from the intended flight plan, as shown in Figure 1.

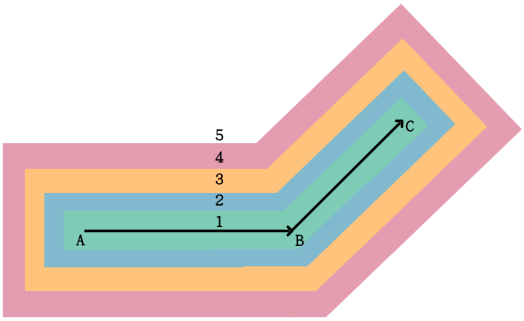

Fig. 1: Safety zones

The size of each safety zone is not definitely fixed; it can be defined for a specific requirement or for a given application. In practice the safety zones are specifically defined for a flight environment and for a given flight plan. The main principle is that no human should be present in Zones 1 to 3, while a few people can be present in Zone 4 and most people can be present in Zone 5. As a consequence, the probability that the UAV endangers humans is directly proportional to the probability that it enters Zones 4 or 5. In the following of the paper, our target will therefore be to compute this probability.

2.2 Drone components

We now move to the decomposition of the UAV hardware and UAV software into components and introduce the most important component in the UAV system: the flight controller (FC). The FC is responsible for collecting data from various sensors, using this data to compute the precise *position* and *attitude* of the drone and adjust the attitude in order to follow the given flight plan to the best of its ability.

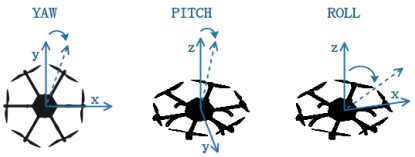

Fig. 2: Attitude coordinates

Notice the difference between position and attitude: while the position of the UAV is defined by 3-dimensional coordinates x, y and z, its attitude is the collection of *yaw*, *pitch* and *roll* measurements for the UAV compared to the vertical (see Figure 2). The attitude allows to control the movement

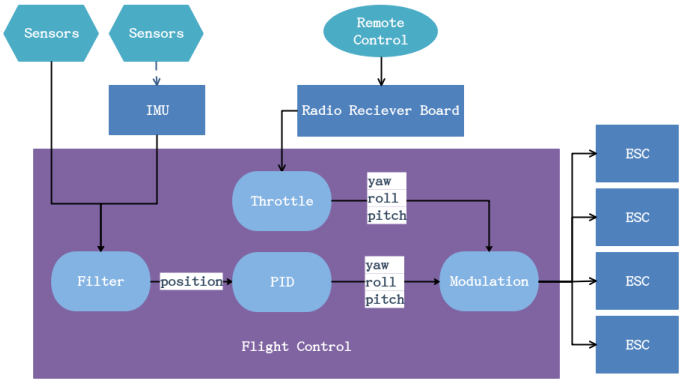

Fig. 3: Flight Control Overview

of the UAV: by controlling the speed of each motor, one can control which motor will be the highest, and hence control the direction the UAV will fly to.

Flight Controller As explained above, the FC is the central component in any UAV as it is responsible for collecting data from sensors and translating them to the UAV attitude. An overview of the FC of an UAV is given in Figure 3. Remark that the FC can be linked to components responsible for communicating with a remote control. While these components are necessary in order to allow a pilot to take over when the automatic flight mode of the UAV fails, we will consider in the following that this is not the case and that the UAV we study are always in automatic flight mode.

As one can see from Figure 3, the intuitive behavior of the FC is as follows. The filter uses sensors measurements in order to compute the current drone position and attitude. Since the data can be noisy and inaccurate, the filter uses complex algorithms in order to clean the noises in the measurements and compute a realistic position and attitude. Remark that in some cases, the filter can itself introduce inaccuracy in the computed position and attitude, which can be problematic. Once the estimated current position and attitude are computed, the Proportional Integral Derivative (PID) uses this information to compute the local trajectory that the drone has to follow in order to be as close as possible to its intended flight plan. This local trajectory is then transformed into a new value for the attitude of the drone. Finally, Modulation transforms this attitude into signal to the Electronic Speed Controller (ESC) which is responsible for controlling each motor's speed.

Recall that we are interested in computing the probability that a UAV enters a forbidden zone while following its flight plan. By construction, as long as the position and attitude measurements are perfect, there is no reason why the UAV should deviate from its intended trajectory, and therefore the probability that it enters a forbidden zone is null. However, as explained above, the data gathered

from sensors can be noisy and inaccuracy can sometimes be introduced through filtering. In this case, the estimated position and attitude of the UAV can be faulty, resulting in a deviation from the intended flight plan and potentially leading to a forbidden zone. It is therefore of paramount importance to study how the filters work and to take into account in our formal model the potential inaccuracy of position and attitude measurement.

Filter The role of the filter is to use sensors measurements in order to compute the UAV position and attitude with the highest possible precision. However, the high precision comes with a cost in terms of complexity: in order to gain precision, filters have to run complex algorithms which takes time. As a consequence, the most precise filters are also the slowest, which implies that the position can be estimated less often, which itself results in inaccuracy.

There exists a large amount of filters in UAV industry, among which one can find Extended Kalman Filter (EKF) [22], Explicit Complement Filter [8], Gradient Descent [19], Conjugate Gradient, and a more accurate but slower filter: Unscented Kalman Filter (UKF) [6], etc. Usually, researchers use EKF as a fundamental to compare to other kinds of filters and explain precision and speed differences. All filters improve their accuracy during the flight through training, in particular by recording recurrent noises and correcting them. However, this training is only valid through a single flight and is lost as soon as the UAV lands.

Since the accuracy of the estimated position and attitude is of paramount importance for computing the probability of entering a forbidden zone, and since the choice of filter has a direct impact on this measurement, we chose to implement this accuracy as a *parameter* of our model. This will be explained in more details in Section 2.4.

2.3 Formal Model of the UAV in its Environment

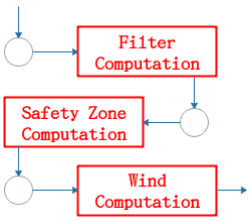

Fig. 4: A flow diagram of the formalization steps

We use a flow diagram to present our global approach for formalizing the UAV functioning (See Figure 4). After a step where the filter computation reflects the precision of position and attitude estimation, we consider the computations of the probabilities to reach the given safety zones in the next time-step; accordingly, the idea is to adapt the next attitude according to the original flight plan in order to be more secure. The last step allows to incorporate wind perturbations and compute the next UAV position.

As explained above, the filter is one of the most crucial components and its ability to estimate the UAV position precisely has a huge impact on the probability of reaching a forbidden zone. For this reason, we choose to represent the accuracy of the estimated position of the UAV (therefore including both sensor measurements and filter correction) as a parameter of our model. In the following, we show how the next

position of the UAV is computed according to the current estimated position, and how errors in the estimation can lead to the drone entering forbidden zones.

Computation of the next position We now explain how the next position is computed according to the estimated current position. In particular, we show that inaccuracy in the estimation can lead the UAV to entering a forbidden zone.

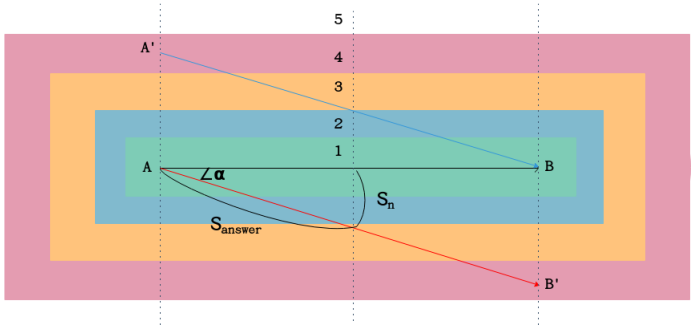

Fig. 5: Issue on drone location and misleading positions

For the sake of simplicity, we assume here that the UAV moves in 2 dimensions only and that inaccuracy only occurs on one of them. Figure 5 illustrates the situation. Assume that the intended flight plan consists in going from point A to point B. Assume also that the current position of the UAV is exactly on A but that the estimated position (taking into account sensors and filter inaccuracy) is on A'. As a consequence, the PID will try to correct the current deviation by changing the angle of the UAV in order to lead it back to B. However, since the UAV is really on A, the correction will instead lead the UAV to a position B', in the forbidden zone. Fortunately, the position estimation takes place several times between A and B, according to the filter frequency f. Therefore, a new position will be estimated before reaching B', hopefully with a better accuracy, which will allow the PID to again correct the trajectory. We should also take into account that the speed of the UAV is also computed according to the flight plan, which precises the remaining time and distance before the next checkpoint. We now show how we can compute the safety zone where the UAV ends before the position is estimated again. In Figure 5, this zone is represented by the distance S_n.

Let S_{answer} be the distance that the UAV covers before a new estimation of the position. Let V be the velocity of the UAV, which is computed by the PID in order to reach B on time, i.e. in precisely T time units. We therefore have $V = A'B/T$, and $S_{answer} = V/f = (A'B)/(T * f)$. Finally, $AA'/A'B = S_n/S_{answer}$, and therefore

$$S_n = \frac{AA'}{T * f}.$$

Remark that the resulting deviation is directly proportional to AA'/f, hence the necessity to take into account the trade-off between accuracy and filter speed in order to optimize the probability of never entering any dangerous zone.

Taking into account wind perturbations follows a similar computation than the one presented above. This allows us to incorporate wind parameters as well in our model.

2.4 Resulting Global Model

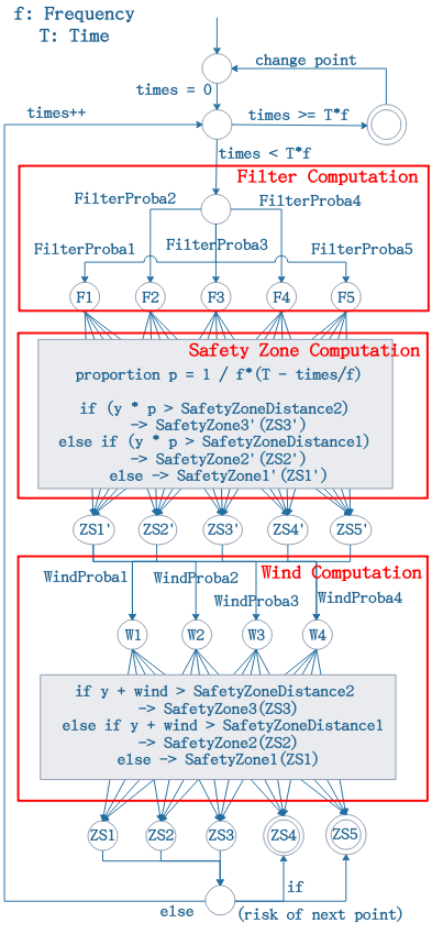

Fig. 6: Global behaviour of the FCS

The global model of the UAV flight control system is depicted in Figure 6. The purpose of this model is to represent the computations taking place in the FCS in order to adapt the UAV trajectory to the intended flight plan according to inaccurate position and attitude estimations as well as wind perturbations. In this model, the exact position of the UAV is encoded using 3-d coordinates. These coordinates are then compared to the intended flight plan in order to decide to which safety zone they belong. As soon as the UAV reaches one of the forbidden zones (4 or 5), the computation stops.

The model uses several probabilistic parameters. Parameters FilterProba1, FilterProba2, FilterProba3, FilterProba4 and FilterProba5 represent the accuracy of the position and attitude estimation by both the filter and the sensors. The resulting probabilistic choice depicted in the box labelled **Filter Computation** therefore dictates the distance between the exact and estimated position of the UAV. This choice is followed by a computation in the box labelled **Safety Zone Computation** that computes the exact coordinates of the next position of the drone and allows to decide the safety zone to which this position belongs. When the wind is not taken into account, the result of this computation is enough to decide whether the model should pursue its execution. When the wind is taken into account, another step follows, depicted in the box labelled **Wind Computation**, where other probabilistic parameters are used in order to decide the wind strength (we assume

that the direction is constant) and a new position taking into account these perturbations is computed. Finally, the zone to which this last position belongs is computed and, depending on whether this zone is safe, the model goes on to another position estimation.

Remark that the filter frequency and the position and distance of checkpoints in the flight plan are given as inputs to the model. The position of checkpoints in the flight plan allows to compute the required UAV speed, while the frequency of the filter allows to fix the number of position estimations that will happen in a given flight plan (i.e. the number of loops the model goes through, at most).

3 Parametric Statistical Model Checking

As explained above, we have developed a parametric probabilistic model in order to represent the behaviour of our UAV according to a given flight plan. We now introduce the necessary theory to formally compute the probabilities of a given UAV entering a forbidden zone in the context of its flight plan. We start by recalling a classical verification technique called Statistical Model Checking (SMC), then introduce the modeling formalism we use: parametric Markov Chains (pMCs) and finally show how SMC can be adapted to this formalism.

3.1 Standard Statistical Model Checking

Recall that a Markov Chain (MC) is a purely probabilistic model $\mathcal{M} = (S, s_0, P)$, where S is a set of states, $s_0 \in S$ is the initial state, and $P : S \times S \to [0, 1]$ is a probabilistic transition function that, given a pair of states (s_1, s_2), yields the probability of moving from s_1 to s_2.

Given a MC $\mathcal{M}$, one can define a probability measure on the infinite executions of $\mathcal{M}$ using a standard construction based on the σ-algebra of cylinders.

A run of a MC is a sequence of states $s_0, s_1 \ldots$ such that for all i, $P(s_i, s_{i+1}) > 0$. Given a finite run $\rho = s_0 s_1 \ldots s_l$, its length, written $|\rho|$ represents the number of transitions it goes through (including repetitions). Here $|\rho| = l$. We write $\Gamma_{\mathcal{M}}(l)$ (or simply $\Gamma(l)$ when $\mathcal{M}$ is clear from the context) for the set of all finite runs of length l, and $\Gamma_{\mathcal{M}}$ for all finite runs *i.e.* $\Gamma_{\mathcal{M}} = \cup_{l \in \mathbb{N}} \Gamma_{\mathcal{M}}(l)$. As usual we define the probability measure, written $\mathbb{P}_{\mathcal{M}}$ on runs based on the sigma-algebra of cylinders (see e.g. [2]). This gives us that for any finite run $\rho = s_0 s_1 \ldots s_n$, $\mathbb{P}_{\mathcal{M}}(\rho) = \prod_{i=1}^{n} P(s_{i-1}, s_i)$. In the rest of the section, we only consider $finite$ runs. Given a reward function $r : \Gamma(l) \to \mathbb{R}$, we write $\mathbb{E}^l_{\mathcal{M}}(r)$ for the expected value of r on the runs of length l of a given MC $\mathcal{M}$.

Statistical Model Checking [23] is an approximation technique that allows to compute an estimation of the probability that a purely probabilistic systems satisfies a given property[4]. In particular, the Monte Carlo technique uses samples of the runs of length l, $\Gamma(l)$, of a given Markoc chain $\mathcal{M}$ in order to estimate the

[4] Particular SMC techniques also allow to estimate the satisfaction of qualitative properties [18].

probability that $\mathcal{M}$ satisfies a given bounded linear property. It can also be used for approximating the expected value of a given reward function r on the runs $\Gamma(l)$ of $\mathcal{M}$. In order to provide some intuition, we briefly recall how standard Monte Carlo analysis works in the context of statistical model checking of MC. In this context, a set of n samples of the runs of the MC. These runs are generated at random using the probability distribution define through the Markov chain. Each of these samples is evaluated, yielding a reward value according to the reward function r. According to the law of large numbers (see e.g. [21]), the mean value of the samples provides a good estimator for the expected value of the reward function r on the runs of the given MC. Moreover, the central limit theorem provides a confidence interval that only depends on the number of samples (provided this number is large enough).

3.2 Parametric Markov Chains(pMC)

Markov Chains are inadequate in the context of drone flight plan analysis. Indeed, the models we develop in this context are subject to uncertainties that we model using parameters, such as precision of the position and attitude estimations and wind strength. The resulting models are therefore not purely probabilistic since they contain parameters. As a consequence, we need to use a more expressive type of model that allows to take into account probabilistic parameters, such as Parametric Markov Chains (see e.g. [1]).

A pMC is a tuple $\mathcal{M} = (S, s_0, P, \mathbb{X})$ such that S is a finite set of states, $s_0 \in S$ is the initial state, $\mathbb{X}$ is a finite set of parameters, and $P : S \times S \to Poly(\mathbb{X})$ is a parametric transition probability function, expressed as a polynomial on $\mathbb{X}$. A parameter valuation is a function $v : \mathbb{X} \to [0, 1]$ that assigns values to parameters. A parameter valuation v is valid w.r.t. a given pMC $\mathcal{M}$ if, when replacing parameters with their assigned values, the resulting object is a MC (i.e. the outgoing probabilities of all states sum up to 1). If v is a valid parameter valuation with respect to $\mathcal{M}$, the resulting Markov chain is written $\mathcal{M}^{\sqsubseteq}$.

Given a pMC $\mathcal{M}$, a run ρ of $\mathcal{M}$ is a sequence of states $s_0 s_1 \dots$ such that for all $i \geq 0$, $P(s_i, s_{i+1}) \neq 0$ (*i.e.* the probability is either a strictly positive real constant or a function of the parameters). As for MCs, we write $\Gamma_{\mathcal{M}}(l)$ for the set of all finite runs of length l and $\Gamma_{\mathcal{M}}$ for the set of all finite runs.

Observe that for any valid parameter valuation v, $\Gamma_{\mathcal{M}^v}(l) \subseteq \Gamma_{\mathcal{M}}(l)$ since v may assign 0 to some transition probabilities.

3.3 Parametric SMC

As it is, standard SMC cannot be used in the context of pMC because of their parametric nature. Indeed, we cannot produce samples according to the parametric transition probabilities. Luckily, the underlying theory used in SMC can be extended in order to take into account parameters. The method we propose in the following is in line with a technique called *importance sampling* (see [21] for a description). The purpose of this technique is to sample a stochastic system using a chosen probability distribution (which is not the original distribution

present in this system) and "compensate" the results using a *likelihood ratio* in order to estimate a measure according to the original distribution. In the context of SMC, importance sampling has mainly been used in order to estimate the probability of rare events [3] and/or to reduce the number of required samples in order to obtain a given level of guarantee [14]. It has also been used in the context of parametric continuous-time Markov chains in order to estimate the value of a given objective function on the whole parameter space while using a reduced number of samples [4]. However, to the best of our knowledge, importance sampling has never been used in order to produce symbolic functions of the parameters as we do here.

The intuition of the method we propose here is to fix the transition probabilities to an arbitrary function f, which we call *normalization function*, and to use these transition probabilities in order to produce samples of the pMC $\mathcal{M}$. However, instead of evaluating the obtained runs by directly using the desired reward function r, we define a new (parametric) reward function r' that takes into account the parametric transition probabilities. We show that, under any parameter valuation v, the evaluation of the mean value of r' on the set of samples is a good estimator for the expected value of the reward r on $\mathcal{M}^v$. The central limit theorem (see e.g. [21]) also allows to produce parametric confidence intervals, but we do not go into details here (see [7] for more details on this topic).

Remark. The choice of the normalization function is crucial. In particular, the results presented below require that the graph structure of the MC obtained with this normalization function is identical to the graph structure of the MC obtained using the chosen parameter valuation. This is discussed in more details in [7]. In the following, we only consider parameter valuations that assign non-zero probability to parameterized transitions. Since we use the uniform normalization function, the graph structures of the obtained MCs are indeed identical, which ensures that the results presented below hold as expected.

Let $Pa : \Gamma_{\mathcal{M}} \to Poly(\mathbb{X})$ be a parametric reward function. For any valid valuation v and any run $\rho \in \Gamma_{\mathcal{M}^v}$ we have $\mathbb{P}_{\mathcal{M}^v}(\rho) = Pa(\rho)(v)$.

Given any valid normalization function f and any run $\rho \in \Gamma_{\mathcal{M}}$, let parametric reward function r' be $r'(\rho) = \frac{Pa(\rho)}{\mathbb{P}_{\mathcal{M}^f}(\rho)} r(\rho)$.

We now prove that the expected values are equal. Let $\rho \in \Gamma_{\mathcal{M}^f}(l)$ be a random sample of $\mathcal{M}^f$ and let Y be the random variable defined as follows $Y = r'(\rho)$. The following computation shows that, under any valid parameter valuation v such that $\mathcal{M}^f$ and $\mathcal{M}^v$ have the same structure, we have $\mathbb{E}(Y)(v) = \mathbb{E}^l_{\mathcal{M}^v}(r)$.

$$\begin{aligned}
\mathbb{E}(Y)(v) &= \Big(\sum_{\rho \in \Gamma_{\mathcal{M}^f}(l)} \mathbb{P}_{\mathcal{M}^f}(\rho) y(\rho) \Big)(v) \\
&= \Big(\sum_{\rho \in \Gamma_{\mathcal{M}^f}(l)} \mathbb{P}_{\mathcal{M}^f}(\rho) \frac{Pa(\rho)}{\mathbb{P}_{\mathcal{M}^f}(\rho)} r(\rho) \Big)(v) \\
&= \sum_{\rho \in \Gamma_{\mathcal{M}^f}(l)} Pa(\rho)(v) r(\rho) \\
&= \sum_{\rho \in \Gamma_{\mathcal{M}^f}(l)} \mathbb{P}_{\mathcal{M}^f}(\rho) r(\rho) \\
&= \sum_{\rho \in \Gamma_{\mathcal{M}^v}(l)} \mathbb{P}_{\mathcal{M}^f}(\rho) r(\rho) \\
&= \mathbb{E}^l_{\mathcal{M}^v}(r)
\end{aligned}$$

Our adaptation of the Monte Carlo technique for pMC is thus to estimate the expected value of Y in order to obtain a good estimator for the expectation of r. Let $\rho_1, \ldots, \rho_n$ be a set of n runs of length l of $\mathcal{M}^f$. Let Y_i be the random variable with values in $Poly(\mathbb{X})$ such that $Y_i = r'(\rho_i)$. Notice that the Y_i are independent copies of the random variable Y. Y_i are therefore independent and identically distributed. Let γ be the parametric function giving their mean value.By the results above, for all valid parameter valuation v such that $\mathcal{M}^v$ and $\mathcal{M}^f$ have the same structure, $\mathbb{E}^l_{\mathcal{M}^v}(r) = \mathbb{E}(Y)(v) = \mathbb{E}(\sum_{i=1}^n Y_i/n)(v) = \gamma(v)$. Our parametric approximation of the expected value is therefore:

$$\widehat{\gamma} = \textstyle\sum_{i=1}^n Y_i/n.$$

In the sequel we will this use Parametric Statistical Model Checking (PSMC) to check the formal model we will implement for the UAV.

4 Implementation, Experimentations and Results

While our complete formal model has been introduced in Section 2 in the form of an automata, we now explain how we successively implemented and improved the model by considering different formalisms and model checking tools. At each step, we show the limitations of the related model which leads to the next step of the implementation. The different steps of the model implementations are depicted in Figure 7.

To start, a first partial version of the formal model of Section 2.3 was implemented as a PRISM model using the PRISM tool [16], without parameters.

This first version, as depicted in 7a, corresponds to a very simple UAV flight plan, going in a straight line from point A to point B in T time units. In this context, the intermediate positions are estimated $T * f$ times, where f is the frequency of the filter. The sizes used for the five security zones are respectively 20m, 40m, 60m, 80m and 100m.

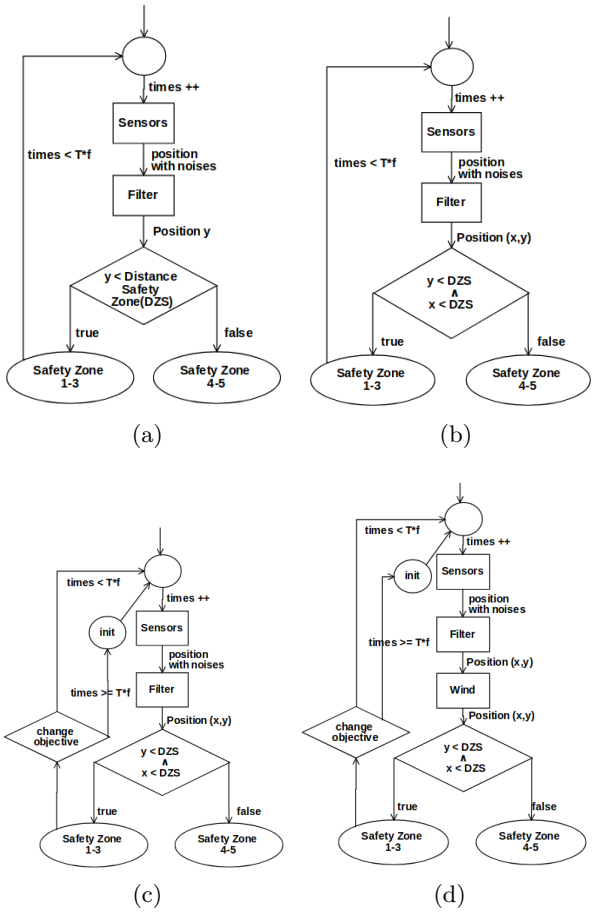

Fig. 7: Incremental development of the SMC model

As explained in Section 2, the filter removes the noise corrupting data coming from sensors. In this first version, we only consider potential deviations along the y-axis. At each computation step, the inaccurate position given by the filter is computed using the accuracy of the filter and sensors (as a single real-valued variable), and compared to the intended position as given by the flight plan. The safety zone is deduced from the distance between the estimated position and the intended position. If the UAV enters Zones 4 or 5, the computation stops.

In this first model, the accuracy of the filter is probabilistic but not parametric, i.e. probability values have been encoded directly in the model. These values are the results of a set of experiments performed by using a flight controller plugged on a production line with a predefined path with a loop. We launched several runs of the device on the production line path and measured the outputs of the EKF filter. These measures then allowed us to compute the estimated position, which can then be compared to the exact position on the production line. We consequently obtained probabilities for the accuracy of the position estimation using an EKF filter and sensors coming from an industrial UAV. However, the major drawback of these experimentations is that they did not reflect a realistic UAV environment. In particular, since the experiment was conducted indoor using a fixed production line, the precision of some of the sensors (GPS for instance) is not representative of the precision one could obtain in a realistic flight environment. Although we were able to verify this model using PRISM, the results are not representative and can only be considered as a proof-of-concept. Since our aim is to study the same problem for

different accuracy probabilities, we changed the exact probability values to parameters and submitted this new model to the PRISM Model Checker. However, because of the real-valued variables used in the model and of the numerous intermediate computations, PRISM was not able to handle this model and timed-out after 2 hours of unsuccessful computations.

Facing these shortcomings with the PRISM tool, we considered the implementation of our model with the PARAM tool [12] which is a model checker for parametric discrete-time Markov chains. PARAM is efficient and allows to compute the probability of satisfying given properties as polynomials or rational functions of the parameters. As PRISM, PARAM also failed to model check our current version of the model. At this stage, since both PARAM and PRISM failed to verify our simplest model because of its complexity, we considered using a different approach based on Parametric Statistical Model Checking. For this purpose, we developed a prototype tool[5]. In this context, our model was expressed as a python program using real-valued variables both for the position of the UAV and for the probabilistic parameters. It appears that PSMC is particularly efficient in this context, and was able to verify our model (by performing more than 20k simulations) in less than 1 minute. We therefore chose to pursue our experimentation using this prototype tool and refined versions of our model.

In the second version of the model, depicted in Figure 7b, we allowed deviations to also occur along the x-axis. This is not problematic when considering a straight line flight plan, but could become important as soon as the flight plan is curved (as the one in Figure 1). Indeed, in this context, deviations along the x-axis (for example if the drone is "late") could result in the PID deciding to cut the trajectory, i.e. going straight to point C before reaching point B, therefore promoting a trajectory that might colide with the forbidden safety zones. Again, our tool managed to verify this model in a very short time.

For the third version of the model as depicted in Figure 7c, we add a third target point to the flight plan, which is not aligned with the first to points, i.e. like in Figure 1.In this third version, the inacurracy of the position estimation along the x-axis also allows the UAV to be "late" and decide to cut the flight plan as explained above.

Finally, the last version, as depicted in Figure 7d, takes into account wind perturbations. We assumed here that the wind direction was constant but that the wind force was again parametric. This will allow us to study the right tradeoff between filter capacity and frequency *depending on the weather conditions*. This last version is the most complex we studied, and therefore took more time to verify than the previous ones. With our prototype tool, it took 190 seconds to perform the verification using 10k simulations in this context while the same amount of simulations only took 28 seconds for the previous model (without wind parameters).[6]

[5] available at `https://github.com/paulinfournier/MCpMC`

[6] We do not share the exact models used in our prototype tool for confidentiality reasons, but the models used in PRISM and PARAM can be found here: `https://github.com/br4444/modelPrism/tree/master`

The outputs of our prototype tool are multivariate polynomials on the parameters of our model. Given the number of parameters, the size of the model and the length of the considered simulations, these polynomials are quite complex and therefore difficult to report in this paper. As an example, below is the output polynomial representing the probability that a UAV enters Zones 4 or 5 using our last version of the model:

$$\begin{aligned} &0.43 * ProbaFilter_3 * ProbaWind_1 \ + \ 0.16 * ProbaFilter_3 * ProbaWind_2 \\ &+ \ 0.17 * ProbaFilter_3 * ProbaWind_3 \ + \ 0.28 * ProbaFilter_3 * ProbaWind_4 \\ &+ \ 0.85 * ProbaFilter_4 * ProbaWind_1 \ + \ \ldots \end{aligned}$$

Instead of showing the resulting polynomials, we will only present the evaluation of these polynomials using realistic values for the parameters. We defined two scenarios (Scenario 1, Scenario 2) with one set of values of parameters for each scenario. For these two scenarios, ProbaF0 (resp F1, F2, F3, F4) models the probability that the estimated position is from 0 to $2m$ (resp. $2-4m$, $4-6m$, $6-8m$, $8-10m$) from the real position. In the first (resp. second) scenario, we have set these values to 0.15/0.3/0.4/0.1/0.05 (resp. 0.1/0.25/0.35/0.2/0.1). According to experiments done at PIXIEL, the first scenario is more realistic than the second one. Similarly, the wind parameters correspond to the probability of having a wind force of $0-20km/h$,$20-30km/h$, $30-50km/h$ and $50-70km/h$ respectively and have been set to 0.55/0.43/0.01/0.01 (which corresponds to typical weather conditions in Nantes, France) for the numerical evaluation. In both scenarios, Zone 4 (resp. 5) is situated $8m$ (resp. $50m$) from the flight plan.

In Table 1, we gather the results for running the simulation for the two considered scenarios; the simulation with PSMC is performed with 10k, 20k and 50k samples. Each time, a polynomial is computed and then evaluated using the parameter values given above. In order to illustrate the stability of our results despite their statistical nature, each complete scenario was performed two times (labelled V1 and V2 in the table). The value reported in the table represents the probability of the UAV eventually reaching Zones 4 or 5 during its flight. Experiments were performed using the formal models presented in Figure 7c (without wind) and Figure 7d (including wind perturbations) on a flight plan resembling the one shown in Figure 1, with a total flight duration of $5s$ and a filter frequency of $1Hz$. We considered two versions of the model from Figure 7d: 7d(np) where wind strength is directly input as a constant probability in the model (resulting in a polynomial where the only variables represent the precision of position estimation), and 7d(p) where wind strength is input as parameter variables in the model (allowing to evaluate/optimize the resulting polynomial according to any wind strength). Remark that the results in the first case are more precise because there are less variables in the polynomial, and obtained in a more efficient manner. Depending on whether we are interested in specific or generic information concerning the weather environment, we can chose to use the first of the second version. Remark that the probabilities of entering the forbidden zones are quite high. This is not surprising as Zone 4 is situated $8m$ from the intended trajectory and the precision of position estimation can be up

Table 1: Results of the experiments

	Model	**10k**		**20k**		**50k**	
		V1	**V2**	**V1**	**V2**	**V1**	**V2**
Running time	7c	28s		51-54s		142-143s	
Scenario 1	7c	4.99%	5.09%	4.74%	5.10%	4.91%	4.98%
Scenario 2	7c	10.38%	10.04%	9.82%	10.05%	9.95%	9.81%
Running time	7d(np)	28s		53-54s		149-155s	
Scenario 1	7d(np)	5.43%	5.31%	5.61%	5.21%	5.59%	5.47%
Scenario 2	7d(np)	10.8%	10.9%	10.8%	10.8%	10.9%	10.7%
Running time	7d(p)	185-190s		311-314s		612-621s	
Scenario 1	7d(p)	4.95%	5.97%	5.28%	6.62%	4.16%	5.61%
Scenario 2	7d(p)	9.55%	9.87%	10.3%	11.3%	9.57%	10.7%

to $10m$. These values have been made deliberately high for the purpose of this study but can be chosen more realistically when verifying the real model.

5 Conclusion and Future Work

In this paper, we have presented a formal model to study the safety of a UAV in automatic flight following a predefined flight plan. This formal model consists in a parametric Markov Chain and takes that takes into account the precision of position and attitude estimation using sensors and filters as well as potential wind perturbations. We have also proposed a new verification technique for parametric probabilistic model: parametric Statistical Model Checking. This new technique has been implemented in a prototype tool. While state of the art tools such as PRISM and PARAM have timed out on the verification of the simplest version of our formal model, our prototype tool has been able to successfully verify the most complex version in less than 12 minutes.

In the future, we plan to keep enhancing our model in order to include filter frequency to be used as a parameter in the model. Using these parameters will allow us to obtain the parametric probability to enter dangerous zones depending on both the filter frequency and the precision probabilities. Studying/optimizing this parametric probability will allow PIXIEL to work on the trade-off between frequency and precision in order to choose their components wisely depending on their intended flight plan.

References

1. Alur, R., Henzinger, T.A., Vardi, M.Y.: Parametric real-time reasoning. In: Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of

Computing, May 16-18, 1993, San Diego, CA, USA. pp. 592–601 (1993). https://doi.org/10.1145/167088.167242, https://doi.org/10.1145/167088.167242
2. Baier, C., Katoen, J.: Principles of model checking. MIT Press (2008)
3. Barbot, B., Haddad, S., Picaronny, C.: Coupling and importance sampling for statistical model checking. In: International Conference on Tools and Algorithms for the Construction and Analysis of Systems. pp. 331–346. Springer (2012)
4. Bortolussi, L., Milios, D., Sanguinetti, G.: Smoothed model checking for uncertain continuous-time markov chains. Information and Computation **247**, 235–253 (2016)
5. Chang, Y.H., Hu, Q., Tomlin, C.J.: Secure estimation based Kalman Filter for cyber-physical systems against sensor attacks. Automatica **95**, 399–412 (2018). https://doi.org/10.1016/j.automatica.2018.06.010, https://doi.org/10.1016/j.automatica.2018.06.010
6. de Marina, H.G., Pereda, F.J., Giron-Sierra, J.M., Espinosa, F.: UAV Attitude Estimation Using Unscented Kalman Filter and TRIAD. IEEE Transactions on Industrial Electronics **59**(11), 4465–4474 (Nov 2012). https://doi.org/10.1109/TIE.2011.2163913
7. Delahaye, B., Fournier, P., Lime, D.: Statistical model checking for parameterized models (Feb 2019), https://hal.archives-ouvertes.fr/hal-02021064, working paper or preprint
8. Euston, M., Coote, P., Mahony, R., Kim, J., Hamel, T.: A complementary filter for attitude estimation of a fixed-wing UAV. In: 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 340–345 (Sep 2008). https://doi.org/10.1109/IROS.2008.4650766
9. Freddi, A., Longhi, S., Monteri, A.: A model-based fault diagnosis system for unmanned aerial vehicles. IFAC Proceedings Volumes **42**(8), 71 – 76 (2009). https://doi.org/https://doi.org/10.3182/20090630-4-ES-2003.00012, http://www.sciencedirect.com/science/article/pii/S147466701635755X, 7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes
10. Gasior, P., Bondyra, A., Gardecki, S.: Development of Vertical Movement Controller for Multirotor UAVs. In: Szewczyk, R., Zielinski, C., Kaliczynska, M. (eds.) Automation 2017 - Innovations in Automation, Robotics and Measurement Techniques, Proceedings of AUTOMATION 2017, March 15-17, 2017, Warsaw, Poland. Advances in Intelligent Systems and Computing, vol. 550, pp. 339–348. Springer (2017). https://doi.org/10.1007/978-3-319-54042-9_31, https://doi.org/10.1007/978-3-319-54042-9_31
11. Gonzalez, L.F., Montes, G.A., Puig, E., Johnson, S., Mengersen, K.L., Gaston, K.J.: Unmanned aerial vehicles (uavs) and artificial intelligence revolutionizing wildlife monitoring and conservation. Sensors **16**(1), 97 (2016). https://doi.org/10.3390/s16010097, https://doi.org/10.3390/s16010097
12. Hahn, E.M., Hermanns, H., Wachter, B., Zhang, L.: PARAM: A Model Checker for Parametric Markov Models. In: Touili, T., Cook, B., Jackson, P. (eds.) Computer Aided Verification. pp. 660–664. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)
13. Heredia, G., Caballero, F., Maza, I., Merino, L., Viguria, A., Ollero, A.: Multi-Unmanned Aerial Vehicle (UAV) Cooperative Fault Detection Employing Differential Global Positioning (DGPS), Inertial and Vision Sensors. Sensors **9**(9), 7566–7579 (2009). https://doi.org/10.3390/s90907566, https://doi.org/10.3390/s90907566

14. Jegourel, C., Legay, A., Sedwards, S.: Cross-entropy optimisation of importance sampling parameters for statistical model checking. In: International Conference on Computer Aided Verification. pp. 327–342. Springer (2012)
15. Kwiatkowska, M., Norman, G., Parker, D.: PRISM: Probabilistic Model Checking for Performance and Reliability Analysis. ACM SIGMETRICS Performance Evaluation Review **36**(4), 40–45 (2009)
16. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: Verification of probabilistic real-time systems. In: Gopalakrishnan, G., Qadeer, S. (eds.) Proc. 23rd International Conference on Computer Aided Verification (CAV'11). LNCS, vol. 6806, pp. 585–591. Springer (2011)
17. Kyristsis, S., Antonopoulos, A., Chanialakis, T., Stefanakis, E., Linardos, C., Tripolitsiotis, A., Partsinevelos, P.: Towards Autonomous Modular UAV Missions: The Detection, Geo-Location and Landing Paradigm. Sensors **16**(11), 1844 (2016). https://doi.org/10.3390/s16111844, https://doi.org/10.3390/s16111844
18. Legay, A., Delahaye, B., Bensalem, S.: Statistical model checking: An overview. In: Proc. Runtime Verification - First International Conference, RV 2010, St. Julians, Malta, November 1-4, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6418, pp. 122–135. Springer (2010)
19. Madgwick, S.O.H.: An efficient orientation filter for inertial and inertial / magnetic sensor arrays (2010)
20. Máthé, K., Busoniu, L.: Vision and Control for UAVs: A Survey of General Methods and of Inexpensive Platforms for Infrastructure Inspection. Sensors **15**(7), 14887–14916 (2015). https://doi.org/10.3390/s150714887, https://doi.org/10.3390/s150714887
21. Rubinstein, R.Y., Kroese, D.P.: Simulation and the Monte Carlo method, vol. 10. John Wiley & Sons (2016)
22. Sabatelli, S., Galgani, M., Fanucci, L., Rocchi, A.: A Double-Stage Kalman Filter for Orientation Tracking With an Integrated Processor in 9-D IMU. IEEE Transactions on Instrumentation and Measurement **62**(3), 590–598 (March 2013). https://doi.org/10.1109/TIM.2012.2218692
23. Sen, K., Viswanathan, M., Agha, G.: On statistical model checking of stochastic systems. In: International Conference on Computer Aided Verification. pp. 266–280. Springer (2005)
24. Zhou, Z., Ding, J., Huang, H., Takei, R., Tomlin, C.: Efficient path planning algorithms in reach-avoid problems. Automatica **89**, 28–36 (2018). https://doi.org/10.1016/j.automatica.2017.11.035, https://doi.org/10.1016/j.automatica.2017.11.035