Towards Integrating Formal Verification of Autonomous Robots with Battery Prognostics and Health Management

Towards Integrating Formal Verification of Autonomous Robots with Battery Prognostics and Health Management*

Xingyu Zhao[1], Matt Osborne[1], Jenny Lantair[1], Valentin Robu[1], David Flynn[1], Xiaowei Huang[2], Michael Fisher[2], Fabio Papacchini[2], and Angelo Ferrando[2]

[1] School of Engineering and Physical Sciences,
Heriot-Watt University, Edinburgh EH14 4AS, U.K.
{xingyu.zhao,mho1,jl153,v.robu,d.flynn}@hw.ac.uk
[2] Department of Computer Science,
University of Liverpool, Liverpool L69 3BX, U.K.
{xiaowei.huang,mfisher,fabio.papacchini,angelo.ferrando}@liverpool.ac.uk

Abstract. The battery is a key component of autonomous robots. Its performance limits the robot's safety and reliability. Unlike liquid-fuel, a battery, as a chemical device, exhibits complicated features, including (i) capacity fade over successive recharges and (ii) increasing discharge rate as the state of charge (SOC) goes down for a given power demand. Existing formal verification studies of autonomous robots, when considering energy constraints, formalise the energy component in a generic manner such that the battery features are overlooked. In this paper, we model an unmanned aerial vehicle (UAV) inspection mission on a wind farm and via probabilistic model checking in PRISM show (i) how the battery features may affect the verification results significantly in practical cases; and (ii) how the battery features, together with dynamic environments and battery safety strategies, jointly affect the verification results. Potential solutions to explicitly integrate battery prognostics and health management (PHM) with formal verification of autonomous robots are also discussed to motivate future work.

Keywords: Formal verification · Probabilistic model checking · PRISM · Autonomous systems · Unmanned aerial vehicle · Battery PHM.

1 Introduction

Autonomous robots, such as unmanned aerial vehicles (UAV) (commonly termed drones[3]), unmanned underwater vehicles (UUV), self-driving cars and legged-robots, obtain increasingly widespread applications in many domains [14]. Ex-

* Supported by the UK EPSRC through the Offshore Robotics for Certification of Assets (ORCA) [EP/R026173/1], Robotics and Artificial Intelligence for Nuclear (RAIN) [EP/R026084] and Science of Sensor System Software (S4) [EP/N007565].

[3] We have used the word "drone" interchangeably with the abbreviation UAV as a less formal naming convention throughout the paper.

treme environments – a term used by UK EPSRC[4] to denote environments that are remote and hazardous for humans – are the most promising domains in which autonomous robots can be deployed to carry out a task, such as exploration, inspection of oil/gas equipment on the seabed, maintenance of offshore wind turbines, and monitoring of nuclear plants in high radiation conditions [26].

However, autonomy poses a great challenge to the assurance of safety and reliability of robots, whose failures may cause both a detriment to human health and well-being and huge financial losses. Thus, there are increasing demands on regulation of autonomous robots to build public trust in their use, whilst the development, verification and certification of autonomous robots is inherently difficult due to the sheer complexity of the system design and inevitable uncertainties in their operation [9, 6, 8, 34]. For instance, [21] shows the infeasibility of demonstrating the safety of self-driving cars from road testing alone, and both [23] and [21] argue the need for alternative verification methods to supplement testing. Formal techniques, e.g. model checking and theorem proving, offer a substantial opportunity in this direction [12]. Indeed, formal methods for autonomous robots have received great attention [6, 28], both in controller synthesis, see e.g. [12, 30], and in verifying safety and reliability when the control policy is given, see e.g. [42, 18, 11].

The battery as the power source of autonomous robots plays a key role in real-life missions [41]. However to the best of our knowledge, most existing formal verification studies of autonomous robots, when considering energy constraints, formalise the energy component in a generic and simplified manner such that some battery features are overlooked:

- **Capacity fading**: Over successive recharges, unlike a liquid-fuelled system whose tank volume normally remains unchanged, the charge storage capacity of a battery will diminish over time.
- **Increasing discharge rate**: In one discharge cycle, since the voltage drops as the battery is being discharged, for a constant power output (a product of the voltage and the current), the current increases meaning an increased discharge rate occurs. This is different to a liquid-fuelled system in which a constant power output typically means a constant rate of fuel consumption.

Thus, usual assumptions, like (i) a fixed battery capacity regardless the number of recharges and (ii) constant energy consumption for a given action regardless the stage in a discharge cycle, become potentially problematic.

On the other hand, the battery prognostics and health management (PHM) community has been developing techniques to accurately forecast the battery behaviour in both a life-cycle and a discharge-cycle. We believe such battery PHM results should be integrated into formal studies (either controller synthesis or verification) of robots to refine the analysis. To take a step forward in this direction, in this paper, our main work is as follows:

- We formalise a UAV inspection mission on an offshore wind farm, in which the mission scenario and choice of model parameters are based on a real

[4] https:/epsrc.ukri.org/files/funding/calls/2017/raihubs

industry survey project. The UAV takes a sequence of actions and follows a fixed inspection route on a 6×6 wind farm. It autonomously decides when to return to the base for recharges based on the health/states of the battery. Uncertainties come from the dynamic environment which causes different levels of power demand.
- We explicitly consider the two battery features in our modelling and show (i) how different battery safety strategies, dynamic environments (i.e. different levels of power demand) and the battery chemical features jointly affect the formal verification results; and (ii) the verification results could be either dangerously optimistic or too pessimistic in practical cases, without the modelling of the battery features.
- We discuss important future work on explicitly integrating battery PHM with formal verification, given the trend that advanced PHM algorithms are mostly based on real-time readings from sensors deployed on the battery.

The organisation of the paper is as follows. In the next section, we present preliminaries on probabilistic model checking and battery PHM. The running example is described in Sec. 3. We show our probabilistic model and verification results in Sec. 4 and 5, respectively. Sec. 6 summarises the related work. Future work and contributions are concluded in Sec. 7.

2 Background

2.1 Probabilistic Model Checking

Probabilistic model checking (PMC) [25] has been successfully used to analyse quantitative properties of systems across a variety of application domains, including robotics [28]. This involves the construction of a probabilistic model, commonly using Discrete Time Markov Chain (DTMC), Continuous Time Markov Chain (CTMC) or Markov Decision Process (MDP), that formally represent the behaviour of a system over time. The properties of interest are usually specified with e.g., Linear Temporal Logic (LTL) or Probabilistic Computational Tree Logic (PCTL), and then systematic exploration and analysis is performed to check if a claimed property holds. In this paper, we adopt DTMC and PCTL whose definitions are as follows.

Definition 1. A DTMC is a tuple $(S, s_1, \mathbf{P}, L)$, where:

- S is a (finite) set of states; and $s_1 \in S$ is an initial state;
- $\mathbf{P} : S \times S \to [0, 1]$ is a probabilistic transition matrix such that $\sum_{s' \in S} \mathbf{P}(s, s') = 1$ for all $s \in S$;
- $L : S \to 2^{AP}$ is a labelling function assigning to each state a set of atomic propositions from a set AP.

Definition 2. AP is a set of atomic propositions and $ap \in AP, p \in [0, 1], t \in \mathbb{N}$ and $\bowtie \in \{<, \leq, >, \geq\}$. The syntax of PCTL is defined by *state formulae* Φ and

path formulae Ψ.

$$\Phi ::= true \mid ap \mid \Phi \wedge \Phi \mid \neg\Phi \mid \mathcal{P}_{\bowtie p}(\Psi)$$
$$\Psi ::= X\,\Phi \mid \Phi\, U^{\leq t}\, \Phi \mid \Phi\, U\, \Phi$$

where the temporal operator X is called "next", $U^{\leq t}$ is called "bounded until" and U is called "until". Also, $F\,\Phi$ is normally defined as $true\, U\, \Phi$ which is called "eventually". State formulae Φ is evaluated to be either true or false in each state. Satisfaction relations for a state s are defined:

$$s \models true \quad , \quad s \models ap \quad \text{iff} \quad ap \in L(s)$$
$$s \models \neg\Phi \quad \text{iff} \quad s \not\models \Phi$$
$$s \models \Phi_1 \wedge \Phi_2 \quad \text{iff} \quad s \models \Phi_1 \text{ and } s \models \Phi_2$$
$$s \models \mathcal{P}_{\bowtie p}(\Psi) \quad \text{iff} \quad Pr(s \models \Psi) \bowtie p$$

$Pr(s \models \Psi) \bowtie p$ is the probability of the set of paths starting in s and satisfying Ψ. Given a path ψ, if denote its i-th state as $\psi[i]$ and $\psi[0]$ is the initial state. Then the satisfaction relation for a path formula for a path ψ is defined as:

$$\psi \models X\Phi \quad \text{iff} \quad \psi[1] \models \Phi$$
$$\psi \models \Phi_1 U^{\leq t} \Phi_2 \quad \text{iff} \quad \exists 0 \leq j \leq t$$
$$(\psi[j] \models \Phi_2 \wedge (\forall 0 \leq k < j\ \psi[k] \models \Phi_1))$$

It is worthwhile mentioning that both DTMC and PCTL can be augmented with rewards/costs [7], which can be used to model, e.g. the energy consumption of robots in a mission. Indeed, this is the typical way used in existing studies, and differs from our modelling of battery in this study.

After formalising the system and its requirements in DTMC and PCTL, respectively, the verification focus shifts to the checking of *reachability* in a DTMC. In other words, PCTL expresses the constraints that must be satisfied, concerning the probability of, starting from the initial state, reaching some states labelled as, e.g. unsafe, success, etc. Automated tools have been developed to solve the reachability problem. We use PRISM [24] which employs a symbolic model checking algorithm to calculate the probability that a path formulae is satisfied. More often, it is of interest to know the actual probability that a path formula is satisfied, rather than just whether or not the probability meets a required bound. So the PCTL definition can be extended to allow *numerical queries* by the form $\mathcal{P}_{=?}(\Psi)$ [25].

In general, a PRISM module contains a number of local variables which constitute the state of the module. The transition behaviour of the states in a module is described by a set of commands which take the form of:

$$[Action]\, Guard \rightarrow Prob_1 : Update_1 + ... + Prob_n : Update_n;$$

As described by the PRISM manual[5], the guard is a predicate over all the variables (including those belonging to other modules. Thus, together with the action labels, it allows modules to synchronise). Each update describes a transition

[5] https://www.prismmodelchecker.org/manual/

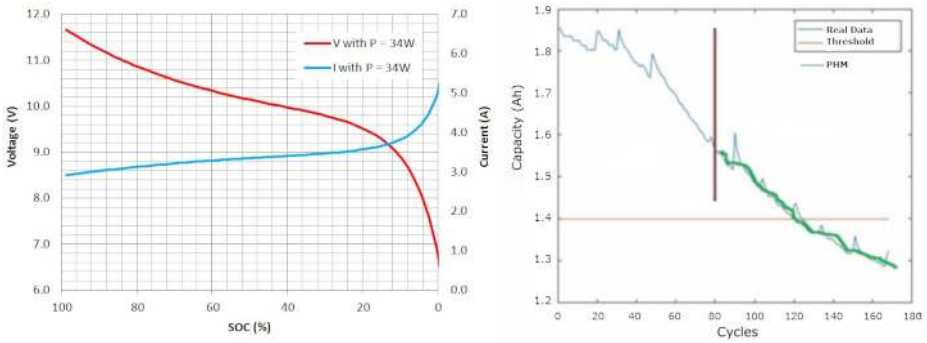

Fig. 1. (Left) The non-linear dynamics of voltage and current vs SOC for a constant power demand from [38]. (Right) Cited from [1], the "real data" curve showing a Lithium-ion battery capacity fade and its PHM predictions (thick green line).

which the module can make if the guard is true. A transition is specified by giving the new values of the variables in the module, possibly as a function of other variables. Each update is also assigned a probability (in our DTMC case) which will be assigned to the corresponding transition.

2.2 Battery Modelling and PHM

Electric batteries exhibit non-linear charge and discharge characteristics due to a number of factors. The voltage varies with the state of charge (SOC) because of changing chemical properties within the cell, such as increasing electrolyte resistance, non-linear diffusion dynamics and Warburg inductance [15]. Fig. 1-(Left), derived from the experimental test in [38], shows such non-linear results of voltage and current vs SOC profile for a constant power demand.

A constant power demand means that an increase in current is drawn as the voltage falls with SOC. For our study, we are interested in a UAV with a battery capacity of around 11Ah and nominal voltage of 22V. The energy supply is 180Wh from a lithium polymer battery. For a 22V battery the voltage at full charge is ∼25V and will drop to ∼20V at a safe threshold of 30% SOC.

The easiest way to measure a change in SOC is by integrating the current discharge over time from a known initial SOC, called Coulomb counting [17]:

$$SOC(k+1) = SOC(k) - \frac{I(k) \times \Delta t}{Q_{max}} \tag{1}$$

where Q_{max} is the maximum SOC, $I(k)$ is the time dependant current, $SOC(k)$ is the SOC percentage at the discrete time step k, Δt is the time step interval. Although this simplification does not take into consideration inaccuracies in the battery initial SOC estimation or account for the internal losses, it is proposed as a first approximation to model the power usage and discharge characteristics as discrete states using the known battery characteristics.

Batteries also degrade over successive recharges due to decreased lithium-ion concentrations so that over 1000 discharge cycles a 20% loss in capacity may occur [36]. Fig. 1-(Right) shows a typical lithium-ion battery capacity fade characteristics the ("real data" curve), cited from [1]. We can observe, after the first 80 discharge cycles, the battery's capacity drops from 1.85(Ah) to 1.55(Ah).

There is a growing interest in the use of PHM techniques to reduce life-cycle costs for complex systems and core infrastructure [13]. Battery health management is also a critical area in regards to the safe and reliable deployment of UAVs. Numerous studies into battery PHM techniques have been carried out, e.g. the use of Neural Nets [10], Unscented Kalman Filters [17, 19], Unscented Transform [4], Hardy Space H_∞ Observers [41] and Physics Based models [16]. Although we are assuming a hypothetical/generic battery PHM method in this paper to provide parameters in our latter modelling, it is envisaged that advanced PHM techniques can be integrated in our future verification framework.

3 The Running Example

As UAV technology improves, energy companies are looking to adopt the technology to reduce maintenance and operating costs. The *resident drone* idea is to station a UAV at locations where aerial surveys are conducted repeatedly. The advantages of such resident drone inspection system are the possibility of increased availability for data collection (e.g. to feed in techniques reviewed in [37]), reduced manual labour, improved safety and more cost effective maintenance strategies. We model a typical application of such system in this paper, based on a survey utilising commercial technologies.

A simplified wind farm drone inspection mission as a 6×6 grid of turbines with a UAV located at the centre is considered. Wind turbines are typically distributed between 5 and 12 turbine blade diameters apart, so a square distribution of turbines is modelled, each 500 meters apart, as shown in Fig. 2-(Left).

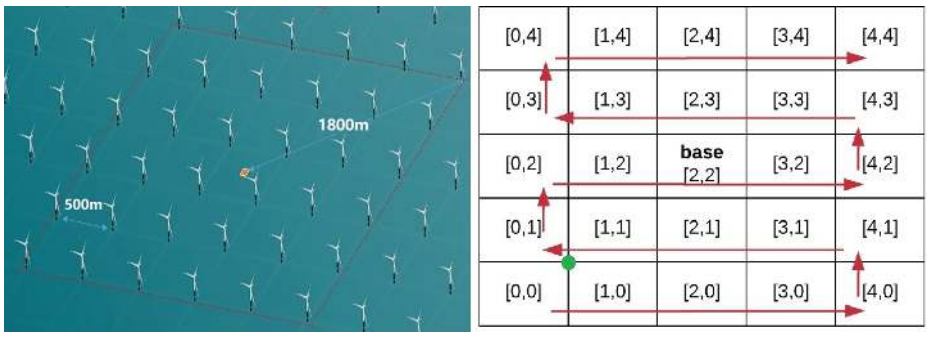

Fig. 2. (Left) A fully autonomous UAV inspection mission in a 6×6 wind farm. (Right) The fixed controller of a UAV inspection mission on the wind farm. Intersections and cell spaces represent wind turbines and transportation channels respectively.

The drone mission requires the drone take-off and land at the base station, fly a distance determined by the number of grid spaces to a turbine, carry out an inspection and return to the base or continue the mission with a single battery charge. For this mission a drone transit velocity of ∼10 m/s is assumed. An inspection is expected to take 15 minutes and the take-off and landing time is estimated less than 1 minute. The battery recharge time is around 1.5 hours.

4 The Modelling in PRISM

Our formal model of the running example presented in section 3 is a product (via parallel composition in PRISM) of four modules – *Drone*, *Grid*, *Environment* and *Battery*. Depending upon the model parameters used, a typical instance of our model has roughly 100,000 states and 170,000 transitions. In what follows, we introduce the modules separately and describe key assumptions, constants, and variables used in each module. Given the page limits, we only show some typical PRISM commands in the modules and omit some sophisticated synchronisation and parallel composition among modules. The complete sources code in the PRISM language can be found in our repository[6].

4.1 The *Drone* Module

The *Drone* module is essentially a finite state machine describing the behaviour of the UAV during the inspection mission, as shown in Fig. 3. The UAV begins the mission in a fully charged state (S0) at the base. Once the UAV successfully takes off (S1), it may either directly land due to violation of the *battery safety strategy* (see section 4.4), or fly to the target cell (S2) and then carry out an inspection of the wind turbine (S3). Depending upon the battery safety strategy and the battery SOC left after the inspection, the UAV will either fly back to the base for recharging (S4), stay in the same cell if there are more than one wind turbines to be inspected, or fly to the next target cell (S2) if all wind turbines of the current cell have been inspected. Once landed at the base (S5), the UAV will declare success of the mission if all wind turbines on the wind farm have been inspected, or recharge and continue the above work-flow otherwise.

The dotted lines in Fig. 3 represent events where the battery SOC falls to 0, leading to an out-of-battery state (S6). Note, the transition from S0 to S6 means that the fully charged capacity is not sufficient to do the next inspection at the target cell and thus the UAV declares the mission failed without further actions.

It is worthwhile to mention that, realistically, there should be some probability of failure for each action, e.g. 10^{-4} for landing. However, since we are only interested in the particular failure mode of out-of-battery here, we simplify our model by setting the failure probability of each action to 0. Thus, the only source of uncertainty we consider is from the dynamic environment which causes different levels of power demand for each action. We will discuss this in Sec. 4.3.

[6] https://x-y-zhao.github.io/files/VeriBatterySEFM19.prism.

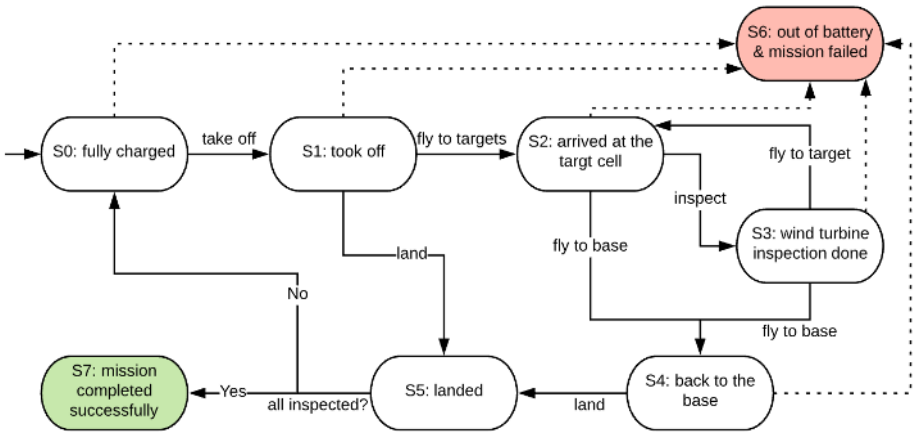

Fig. 3. A finite state machine of the UAV behaviour modelled by the *Drone* module.

4.2 The *Grid* Module

We formalise the wind farm as a 5×5 grid as shown in Fig. 2-(Right) in which the intersections represent wind turbines and the cell spaces (labelled by coordinates $[x, y]$) represent transport channels. In this study, we assume a given control policy (CP) of the UAV as follows:

- CP1: The UAV will follow the snake shaped route, as shown by the red arrows in Fig. 2-(Right), to carry out the inspection in each cell.
- CP2: Depending upon the coordinates of the cell, there can be 1, 2 or 4 *appointed* wind turbines to be inspected within a cell. For instance, at cell [0,0], the wind turbine located at the left-bottom corner is the only appointed one; both the two bottom corners at cell [2,0] need to be inspected; and for cell [2,2], all 4 wind turbines around it should be inspected. Indeed, it would be unwise (i.e. requiring more energy) to fly to cell [0,0] to inspect the green dotted wind turbine in Fig. 2-(Right), rather than fly with the shortest route to cell [1,1].
- CP3: Depending upon the battery safety strategy and the remaining SOC, the UAV may suspend the mission and return to the base for recharging. It will resume the mission at the cell where the mission was suspended.

A part of the PRISM commands in this module are shown in Fig. 4. Note, the transitions probabilities are simplified to 1.

4.3 The *Environment* Module

We explicitly consider the environmental dynamics due to its primary impact on the battery's power demand. For simplicity, only one major factor – wind speed – was considered when developing the *Environment* module. We formalise two

```
module Grid
  cpos_x : [0..4] init base_x;// the current position
  cpos_y : [0..4] init base_y;// the current position
  tpos_x : [0..4] init 0;// the target position
  tpos_y : [0..4] init 0;// the target position
  //if the target cell has been visited before: 0 -- no.  1 -- yes.
  is_tar_visited : [0..1] init 0;
  n_uc_wt :[0 ..4 ] init 1;// number of unchecked wind turbines in a given cell
  ...
  // in the cell [0,0], there is only one bottom-left corner needs to be inspected
  [ins] (s=2)&(cpos_x=0)&(cpos_y=0)&(n_uc_wt=1)
          -> (tpos_x'=cpos_x+1)&(tpos_y'=cpos_y)&(n_uc_wt'=0)&(is_tar_visited'=0);
  ...
  // in the cell [2,0], the bottom-left and bottom-right corners need to be inspected
  [ins] (s=2)&(cpos_x=2)&(cpos_y=0)&(n_uc_wt=2) -> (n_uc_wt'=1);
  [ins] (s=2)&(cpos_x=2)&(cpos_y=0)&(n_uc_wt=1)
          -> (tpos_x'=cpos_x+1)&(tpos_y'=cpos_y)&(n_uc_wt'=0)&(is_tar_visited'=0);
  ...
  //after the inspection, fly to next cell, excluding the case of the last cell [4,4].
  [flytt] (s=3) & ((cpos_x!=tpos_x) | (cpos_y!=tpos_y)) &!((cpos_x=4)&(cpos_y=4))
          -> (cpos_x'=tpos_x) & (cpos_y'=tpos_y)&(n_uc_wt'=n_wt_to_ins)&(is_tar_visited'=1)
  ...
endmodule
```

Fig. 4. Some PRISM commands of the *Grid* module.

levels of wind speed, and use a parameter *p_wsp_c* to capture the dynamics of the wind. Environmental assumptions (EA) are listed below:

- EA1: The UAV will only attempt to take off in a low wind speed condition.
- EA2: The change of wind speed (either from low to high or the other way around) occurs, with a probability of *p_wsp_c*, before each action is taken in the *Drone* module.

```
module Environment
  wsp : [1..2] init 1;// 1 -- low wind speed. 2-- high wind speed.

  [] wsp=1 -> (1-p_wsp_c) : (wsp'=1) + p_wsp_c : (wsp'=2);
  [] wsp=2 -> (1-p_wsp_c) : (wsp'=2) + p_wsp_c : (wsp'=1);
endmodule
```

Fig. 5. The PRISM commands of the *Environment* module.

From EA2, we know that the higher the *p_wsp_c* is, the more dynamic the environment, and it is this assumption that introduces uncertainty in the energy consumption for a given action. The PRISM commands are shown in Fig. 5.

4.4 The *Battery* Module

Fig. 6 shows two abstracted state machines of the *Battery* module which run in parallel to describe the battery behaviour. The battery features of capac-

ity fading (over successive recharges) and increasing discharge rate (in a single discharge cycle) are captured by battery assumptions (BA) as follows:

- BA1: After each recharge, the battery's fully charged capacity (i.e. c_full in Fig. 6) cannot be recovered to the new battery's capacity. Rather, it decreases with a rate which can be obtained from battery PHM experiments, e.g. as observed from the results in [1], for the first 100-ish discharge cycles, at each recharge, the capacity will fade at an average rate of 0.2%.
- BA2: For a given wind speed, the UAV is working at a constant power demand for all actions. Since we considered 2 levels of wind speed in the *Environment* module, there are 2 levels of constant power demand as well.
- BA3: In one discharge cycle, the battery's voltage is essentially a non-linear function of its SOC. We use a step-wise function to approximate the non-linear function – high voltage V_2 (SOC > 0.75), medium voltage V_1 ($0.75 \geq$ SOC ≥ 0.25) and low voltage V_0 (SOC < 0.25).

In line with the BA2 and BA3, we use the following Eq. (2) to estimate the battery consumption (Ah) for an action j (denoted as c_act in Fig. 6) under different levels of voltage V_i and a power demand level k:

$$C_j = \frac{E_{spec} \cdot t_j}{V_i \cdot T_k} \tag{2}$$

where E_{spec} is the specified battery energy, T_k is the total running time under a constant power level k, V_i is the level of voltage, and t_j is the estimated execution time of action j.

For instance, a typical UAV battery with a specified energy of 180Wh ($E_{spec} = 180$) can fly 30 minutes at the normal level of workload ($T_k = 0.5$ and k represents the normal level of power demand). The specified normal working voltage is 22V ($V_1 = 22$) (with a maximum level of 25V, $V_2 = 25$, and minimum level of 20V, $V_0 = 20$). The average time for inspecting a wind turbine is 15 minutes (if action j represents the inspection, then $t_j = 0.25$). Via Eq. (2) and those estimated parameters above, we obtain the last row in Table 1 (results are rounded to one decimal place). Similarly for the battery consumption of each action at the high power demand level (high wind speed environment), the values can be calculated in the same way but are not shown in this paper.

Table 1. Battery consumption (Ah) of actions under different levels of voltage in low wind speed environment (i.e. the normal level of power demand).

	low voltage	medium voltage	high voltage
take-off/land	0.3	0.2	0.1
transport per cell	0.5	0.4	0.3
inspection per wind turbine	4.5	4.0	3.6

So far, in our modelling, instead of assuming a fixed battery consumption for each action, we have 6 possibilities of the battery consumption after an action (3 voltage levels $\times$ 2 power demand levels).

Engineers are aware of the higher risks associated from operating with a lower SOC battery, thus there are requirements on the battery PHM to provide warnings when the SOC falls below a certain threshold [35] (and to recommend that the mission be discontinued), e.g. a typical 30% threshold is adopted by NASA in [19]. In line with that battery safety strategy (BS), we also define a parameter *safe_t* as a safety threshold:

– BS1: before each of the actions, take-off, fly-to-target and inspect, the UAV will check if the SOC will fall below *safe_t* after a sequence of actions to perform an intended inspection. If there is sufficient SOC, then take the action, otherwise return for recharging.

An instance of BS1 is that, before flying to the target cell, the UAV will predict the remaining SOC (based on the current battery health/state and wind conditions) *after* flying to the target and performing one inspection. If there is no safe battery life remained (i.e. SOC< *safe_t*) after the intended inspection, the UAV will go back for recharging. Some typical PRISM commands of this module and associated formulas are shown in Fig. 7.

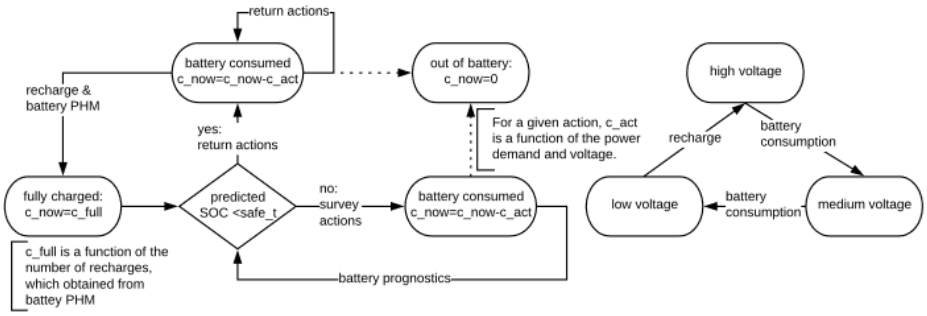

Fig. 6. Abstracted state machines of the *Battery* module.

5 Results

The main properties of interest and their corresponding PCTL formulas are:

– The probability of mission success[7]: $P_{=?}[\, F\ (s = 7)]$;
– The expected mission time: $R\{\text{“mt”}\}_{=?}[\, F\ (s = 7)|(s = 6)]$;

[7] Since we focus on the particular failure mode of out-off-battery in our model, rigorously this should be the probability of seeing no out-off-battery failures in a mission.

```
module Battery
  v : [0..2] init 2;
  // 2 - high terminal voltage , 100%-75% SoC
  // 1 - medium terminal voltage , 75%-25% SoC
  // 0 - low terminal voltage , 25%-0% SoC
  c_now:[0..c_full] init c_full;//The capacity of the battery now.
  n_charge: [0..max_charge_num] init 0;//number of recharges

  // according to the SoC to determine v
  [TO] ((c_now-c_tol)/c_full_vary>0.75)& ((c_now-c_tol-c_ftt-c_ins)/c_full_vary>=safe_t)
          -> (c_now'=c_now-c_tol)&(v'=2);
  ...
  [TO] ((c_now-c_tol)/c_full_vary<0.25) & ((c_now-c_tol-c_ftt-c_ins)/c_full_vary>=safe_t)
          -> (c_now'=c_now-c_tol)&(v'=0);
  //don't have enough battery to do a single inspection
  [lack_batt] (s=0)&((c_now-c_tol-c_ftt-c_ins)/c_full_vary<safe_t)-> true;
  //recharge
  [recharge] n_charge < max_charge_num-> (c_now'=c_full_vary)&(v'=2)&(n_charge'=n_charge+1)
  ...
endmodule
...
// the capacity cost of flying to target asuming, per cell
// v-low takes 0.5Ah, v-medium takes 0.4Ah ,v-high takes 0.3Ah.
formula c_ftt = dis_ct* (v=2 ? (3*wsp): (v=1 ? (4*wsp) : (5*wsp)));
// to calculate the distance from the current position to the target position
formula dis_ct=(max(cpos_x,tpos_x)-min(cpos_x,tpos_x))+(max(cpos_y,tpos_y)-min(cpos_y,tpos_y))
...
```

Fig. 7. Some PRISM commands of the *Battery* module and global formulas. Note, the key variables *c_full_vary* (the fully-charged capacity considering capacity fading) and *c_ftt*, *c_ins* etc. (the battery consumption of each action, which are generically denoted as *c_act* in Fig. 6) should be obtained dynamically from the PHM system in reality, whilst we make simplified assumptions in the source codes.

– The expected number of recharges: $R\{\text{“rc”}\}_{=?}[\, F\ (s = 7)|(s = 6)]$.

We use the PRISM tool [24] to check the properties given different model parameters in later subsections. Indeed, we may only be concerned with the expected mission time (or number of recharges) *given* the mission is successful. However, PRISM can only solve the “reachability reward” properties when the target set of states is reached with probability 1, thus our target state here is $(s = 7)|(s = 6)$. In our later numerical examples, we only show the expected mission time when the probability of the mission failing is very small so that its contribution to the average mission time is negligible. Note, this limitation of PRISM has been studied in [29].

5.1 Effects of Battery Safety Strategies and Dynamic Environments

For a typical new battery with 11Ah capacity, we highlight the verification results of four representative cases, as shown in Table 2, by setting the above mentioned model parameters (cf. BS1 and EA2) as:

– #1, the common case and baseline: $safe\_t = 0.3$, $p\_wsp\_c = 0.1$.

- #2, a risky battery strategy: $safe\_t = 0.25$, $p\_wsp\_c = 0.1$
- #3, a more dynamic environment: $safe\_t = 0.3$, $p\_wsp\_c = 0.3$.
- #4, a risky battery strategy in a more dynamic environment: $safe\_t = 0.25$, $p\_wsp\_c = 0.3$.

Table 2. Verification results of some typical cases with a new battery capacity of 11Ah.

	No. of states	No. of transitions	Prob. mission success	Exp. mission time	Exp. no. of recharges
#1	108,688	163,076	1	4700.20	42.65
#2	117,765	177,278	0.91	3885.95	34.59
#3	108,688	163,076	1	7482.86	72.16
#4	117,765	177,278	0.89	5621.66	53.07

The example of #1 represents the common case that serves as a baseline in Table 2. Case #2 represents the use of a relatively risky strategy by reducing the battery safety threshold from 0.3 to 0.25. Indeed, in Table 2, we see a decreased probability of mission success (from 1 to 0.91), whilst the expected mission time and number of recharges also significantly reduce, which is the benefit of taking more risk. Comparing case #3 and #1, given a fairly safe battery strategy (i.e. $safe\_t = 0.3$), a more dynamic environment will significantly increase the mission time and number of recharges. Because the more dynamic the environment is, the more often the UAV decides to go back for recharges for battery safety reasons. Note, the probability of mission success remains (i.e. 1), since the battery strategy is conservative enough to guarantee a safe trip back to base in all possible circumstances. On the contrary, if we adopt a risky battery strategy in a more dynamic environment (#4), then not only the expected mission time increases but also the probability of mission success decreases (cf. #4 and #2), because there are cases that the UAV does not reserve enough battery to fly back to base due to a sudden change of environments.

5.2 Comparison of Models, Disregarding the Battery Features

Most existing verification studies of autonomous robots, when considering energy constraints, formalise the energy component in a generic/simplified manner such that battery features are overlooked. In this section, we illustrate the difference between a simplified battery model and our relatively advanced model, considering the battery chemical features.

Fig. 8 shows the probability of mission success, via different models, as a function of the new battery's capacity (Ah). The solid curve labelled as "advanced" represents our proposed model considering the battery features (BA1 and BA3). The other curves represent the basic models without considering battery features, e.g. the "basic_high" curve is the case when there is no capacity fading and the battery always works at a high level of voltage.

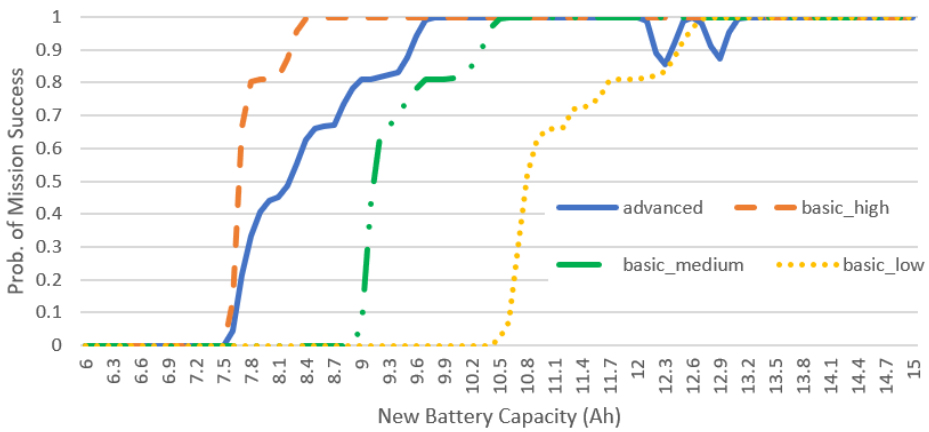

Fig. 8. Probability of mission success, via different model assumptions on batteries, as a function of the new battery capacity.

In Fig. 8, we can observe that, for a given model, there is a required minimum new battery capacity to have non-zero probability to succeed. Indeed, the capacity should be at least enough for the inspection of the first wind turbine and a safe trip back. Since the battery consumption of each action is higher (and highest) when assuming that the battery is always working at a medium (and low) voltage level, such a required minimum capacity increases. Similarly, to guarantee a successful mission, "basic_high" requires that the relatively smallest new battery capacity (around 8.4Ah) due to its obviously optimistic assumptions, i.e. no capacity fading and always working at a high voltage level.

Note, although the "advanced" model is bounded by "basic_high" and "basic_medium", it is still dangerous to use such simplified bounds to do approximation, due to the observed "dip" on the "advanced" curve in the range of 12Ah–13Ah. That dip of probability of mission success happens because, when the new battery's capacity increases (but is still not big enough), the UAV may decide to take more risk to perform more actions in one trip (i.e. one discharge cycle), after which there might not be enough SOC left for a safe trip back in some edge cases (e.g. a degraded battery working at a low voltage in a high wind speed environment). To eliminate this phenomenon, the simplest way is to raise the battery safety threshold, which is confirmed by our extra experiments.

An example path of a failed mission is presented in Fig. 9, in which the new battery's capacity is 12.8Ah (thus within the "dip" range in Fig. 8). After 22 recharges at step #148, the UAV flies to the target cell [0,4] and the wind speed changes to high ($wsp = 2$). At step #150, the predicted SOC after the intended inspection is higher than the safety threshold of 0.3. So instead of returning to the base, the UAV continues the inspection in high-speed wind. Although after managing to return to base, the drone fails to land in a high wind speed and at the lower voltage level. Also, if we naively ignore the capacity fading and/or assuming the battery never works at a low voltage, then the UAV would land

safely in this example. That's why we don't observe the "dips" on the curves of the basic models in Fig. 8.

Step		Drone	Grid				Environment	Battery		
Action	#	s	cpos_x	cpos_y	tpos_x	tpos_y	wsp	v	c_now	n_charge
[land]	147	5	2	2	0	4	2	2	121	21
[recharge]	148	0	2	2	0	4	1	2	122	22
[TO]	149	1	2	2	0	4	1	2	121	22
[flytt]	150	2	0	4	0	4	2	2	109	22
[ins]	151	3	0	4	1	4	2	1	37	22
[rettb]	152	4	2	2	1	4	2	0	5	22
[lack_batt]	153	6	2	2	1	4	2	0	5	22

Fig. 9. A fragment of a failed mission path generated by the PRISM simulator, which is an example of the "dip" in Fig. 8 with the new battery's capacity as 12.8Ah. Note, only key variables of the 4 modules are configured to be viewed here.

Fig. 10 shows the expected mission time (upper graph curves) from the specified battery models and the differences (lower curves) between them. We observe that, in the practical range of the new battery's capacity (i.e. <13Ah, base on our survey), the basic models could give either too optimistic (500 minutes less) or over pessimistic (1500 minutes more) results. Such a variance of 1 to 3 working days will mislead wind farm maintenance activities and thus cause significant economic loss. Not surprisingly, as the new battery capacity tends to infinity (i.e. the battery is no longer a bottle neck of the given mission), the verification results of all models tends to the same value (as do the results in Fig. 8).

6 Related Work

How autonomous robots should be verified is a new challenging question [9, 6], and it has received great attention in recent years, e.g. [11, 30, 33, 42]. When considering energy constraints, the energy consumption is usually formalised in a linear way that being generic for both liquid-fuel and batteries. For instance, in the analysis of robot swarms [22, 27] the authors assume constant energy cost at each time step and a fixed capacity when obtaining energy from "food". Again, in the modelling of UAV missions [12, 18], a fixed battery capacity and constant battery consumption over time is assumed. In [11], energy consumption of UUV sensors is modelled as a reward/cost for each state, which exhibits a linear behaviour over time. Indeed, such generic and simplified assumptions do not necessarily mean that they are unrealistic, whilst we believe more rigorous discussions and studies should be carried out prior to their adoption.

The study in [3] highlights the difference between real and ideal batteries, with a case study on controlling an energy-constrained robot. But it focuses on another battery feature – "recovery effect" (e.g. a smart phone might shutdown due to an out-of-battery failure, but then become live again after an idle period).

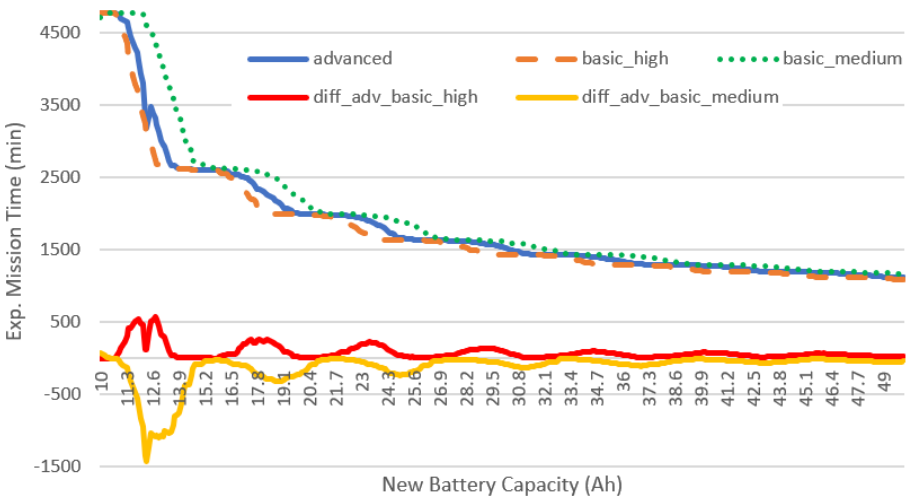

Fig. 10. The upper graph curves show the expected mission time, for the specified battery model, as a function of the new battery capacity. The lower graph curves show the differences of the expected mission times from the specified models.

Beyond the scope of robotics systems, battery behaviour does draw attention for verification. For instance, in [39], the battery of a satellite is described by the Kinetic Battery Model which is formalised as a timed automata to precisely model the discharge behaviour. However they leave out the capacity fading feature as future work. Similarly in [20], the Kinetic Battery Model is used for analysing wireless sensor protocols. For smartphones, [5] uses runtime verification to check whether the actual battery consumption is within the expected limits that are derived from battery consumption profiles for smartphone apps.

7 Discussions, Conclusions and Future Work

In this paper, we formalise a UAV inspection mission of an offshore wind farm, and then do probabilistic model checking in PRISM to show (i) how the battery's non-linear features significantly affect the verification results in most practical cases; and (ii) how battery safety strategies, dynamic environments and battery features jointly affect the verification results.

Most existing formal verification studies of robots make simplified linear assumptions on energy consumption, which is indeed preferable in the case that the capacity is far beyond the total battery cost of the whole mission (i.e. when there is no recharges and the battery's working voltage is fairly stable due to a considerable SOC margin remaining at the end of the mission). In contrast, our work shows how such a simplification can significantly affect the verification results in the case that there are multiple recharges in the autonomous mission. Thus, we believe our work highlights this risk and calls for more rigorous dis-

cussions prior to any battery assumptions made in future formal verification of robots, especially when recharges are expected in the mission scenarios.

Moreover, we believe that battery PHM techniques should be explicitly integrated into the formal verification of robots. Although in this paper we use a hypothetical/generic battery PHM technique to provide the parameters used in the *Battery* module, it is clear how PHM can aid the rigorous modelling of battery for formal verification. For now, both the battery PHM experiments and formal verification are assumed to be carried out in the lab, i.e. prior to the mission. To improve the accuracy, an appealing idea is to integrate both at runtime, since there is a trend of doing online battery PHM based on real-time readings from the sensors deployed on the battery, e.g. [2, 40]. Whilst there will be a scalability issue if running both online battery PHM and formal verification algorithms at runtime. A compromised solution, in our example, is to invoke the formal verification and PHM during the recharging at the base (where substantial computing resources can be used) with newly collected log-data from recent flights. Thus the verification result will be updated with the up-to-date data. We plan to implement this solution in our future work.

Apart from highlighting the need of integrating battery PHM techniques, this work only serves as a first approximation[8] of the verification of the residential drone inspection mission. More rigorous verification/planning of the mission is needed in future, e.g. by gradually refining the fundamental PRISM model based on observations from various sources of data [31, 32].

In summary, our main contributions are:

- We formalise a UAV inspection mission on a wind farm based on a real industry survey project, which can be reused and extended as an exemplar for future research of similar UAV missions.
- We do a sequence of what-if calculations, via probabilistic model checking, to show (i) the importance of considering non-linear battery features in formal verification of autonomous robots; and (ii) how such battery features, together with the dynamic environments and battery safety strategy, jointly affect the verification results.
- We discuss the need of explicitly integrating battery PHM techniques into formal verification of robots, and propose a potential solution which forms important future work.

References

1. Andoni, M., Tang, W., Robu, V., Flynn, D.: Data analysis of battery storage systems. CIRED - Open Access Proceedings Journal **2017**(1), 96–99 (2017)
2. Barré, A., Suard, F., Gérard, M., Riu, D.: A real-time data-driven method for battery health prognostics in electric vehicle use. In: Proc. of the 2nd European Conf. of the Prognostics and Health Management Society. pp. 1–8 (2014)

[8] It is a first approximation in the sense of, e.g. the simplification of two levels of wind speed and the round estimations of battery consumption in Tab. 1.

3. Boker, U., Henzinger, T.A., Radhakrishna, A.: Battery transition systems. In: Proc. of the 41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. pp. 595–606. POPL '14, ACM, San Diego, California, USA (2014)
4. Daigle, M., Goebel, K.: Improving computational efficiency of prediction in model-based prognostics using the unscented transform. Annual Conference of the Prognostics and Health Management Society (2010)
5. Espada, A.R., del Mar Gallardo, M., Salmerón, A., Merino, P.: Runtime verification of expected energy consumption in smartphones. In: Model Checking Software. LNCS, vol. 9232, pp. 132–149. Springer International Publishing, Cham (2015)
6. Farrell, M., Luckcuck, M., Fisher, M.: Robotics and integrated formal methods: Necessity meets opportunity. In: Proc. of the 14th Int. Conf. on Integrated Formal Methods. LNCS, vol. 11023, pp. 161–171. Springer, Cham (2018)
7. Filieri, A., Tamburrelli, G.: Probabilistic verification at runtime for self-adaptive systems. In: Cámara, J., de Lemos, R., Ghezzi, C., Lopes, A. (eds.) Assurances for Self-Adaptive Systems: Principles, Models, and Techniques, LNCS, vol. 7740, pp. 30–59. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)
8. Fisher, M., Collins, E., Dennis, L., Luckcuck, M., Webster, M., Jump, M., Page, V., Patchett, C., Dinmohammadi, F., Flynn, D., Robu, V., Zhao, X.: Verifiable self-certifying autonomous systems. In: 2018 IEEE Int. Symp. on Software Reliability Engineering Workshops (ISSREW). pp. 341–348 (2018)
9. Fisher, M., Dennis, L., Webster, M.: Verifying autonomous systems. Communication of the ACM **56**(9), 84–93 (Sep 2013)
10. Gao, D., Huang, M., Xie, J.: A novel indirect health indicator extraction based on charging data for lithium-ion batteries remaining useful life prognostics. SAE International Journal of Alternative Powertrains **6**(2), 183–193 (2017)
11. Gerasimou, S., Calinescu, R., Banks, A.: Efficient runtime quantitative verification using caching, lookahead, and nearlyoptimal reconfiguration. In: Proc. of the 9th Int. Symp. on Software Engineering for Adaptive and Self-Managing Systems. pp. 115–124. SEAMS 2014, ACM, New York, NY, USA (2014)
12. Giaquinta, R., Hoffmann, R., Ireland, M., Miller, A., Norman, G.: Strategy synthesis for autonomous agents using PRISM. In: NASA Formal Methods. LNCS, vol. 10811, pp. 220–236. Springer International Publishing, Cham (2018)
13. Goebel, K., Celaya, J., Sankararaman, S., Roychoudhury, I., Daigle, M., Saxena, A.: Prognostics: The Science of Making Predictions. CreateSpace Independent Publishing Platform, 1st edn. (2017)
14. Guiochet, J., Machin, M., Waeselynck, H.: Safety-critical advanced robots: A survey. Robotics and Autonomous Systems **94**, 43 – 52 (2017)
15. Hariharan, K.S.: Mathematical Modeling of Lithium Batteries From Electrochemical Models to State Estimator Algorithms. Green Energy & Tech., Springer (2018)
16. He, W., Pecht, M., Flynn, D., Dinmohammadi, F.: A physics-based electrochemical model for lithium-ion battery state-of-charge estimation solved by an optimised projection-based method and moving-window filtering. Energies **11**(8), 2120 (2018)
17. He, W., Williard, N., Chen, C., Pecht, M.: State of charge estimation for electric vehicle batteries using Unscented Kalman Filtering. Microelectronics Reliability **53**(6), 840–847 (2013)
18. Hoffmann, R., Ireland, M., Miller, A., Norman, G., Veres, S.: Autonomous agent behaviour modelled in PRISM – A case study. In: Bošnački, D., Wijs, A. (eds.) Model Checking Software. LNCS, vol. 9641, pp. 104–110. Springer International Publishing, Cham (2016)

19. Hogge, E.F., Bole, B.M., Vazquez, S.L., Celaya, J.R., Strom, T.H., Hill, B.L., Smalling, K.M., Quach, C.C.: Verification of prognostic algorithms to predict remaining flying time for electric unmanned vehicles. International Journal of Prognostics and Health Management **9**(1), 1–15 (2018)
20. Ivanov, D., Larsen, K.G., Schupp, S., Srba, J.: Analytical solution for long battery lifetime prediction in nonadaptive systems. In: Quantitative Evaluation of Systems. LNCS, vol. 11024, pp. 173–189. Springer, Cham (2018)
21. Kalra, N., Paddock, S.M.: Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability? Transportation Research Part A: Policy and Practice **94**, 182 – 193 (2016)
22. Konur, S., Dixon, C., Fisher, M.: Analysing robot swarm behaviour via probabilistic model checking. Robotics and Autonomous Systems **60**(2), 199 – 213 (2012)
23. Koopman, P., Wagner, M.: Autonomous vehicle safety: An interdisciplinary challenge. IEEE Intelligent Transportation Systems Magazine **9**(1), 90–96 (2017)
24. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: Verification of probabilistic real-time systems. In: Proc. 23rd Int. Conf. on Computer Aided Verification. LNCS, vol. 6806, pp. 585–591. Springer (2011)
25. Kwiatkowska, M., Norman, G., Parker, D.: Probabilistic model checking: Advances and applications. In: Formal System Verification: State-of-the-Art and Future Trends, pp. 73–121. Springer International Publishing, Cham (2018)
26. Lane, D., Bisset, D., Buckingham, R., Pegman, G., Prescott, T.: New foresight review on robotics and autonomous systems. Tech. Rep. No. 2016.1, Lloyd's Register Foundation, London, U.K. (2016)
27. Liu, W., Winfield, A.: Modeling and optimization of adaptive foraging in swarm robotic systems. The Int. Journal of Robotics Research **29**(14), 1743–1760 (2010)
28. Luckcuck, M., Farrell, M., Dennis, L., Dixon, C., Fisher, M.: Formal specification and verification of autonomous robotic systems: a survey. arXiv preprint arXiv:1807.00048 (2018)
29. Märcker, S., Baier, C., Klein, J., Klüppelholz, S.: Computing conditional probabilities: implementation and evaluation. In: Cimatti, A., Sirjani, M. (eds.) Software Engineering and Formal Methods. LNCS, vol. 10469, pp. 349–366. Springer International Publishing, Cham (2017)
30. Norman, G., Parker, D., Zou, X.: Verification and control of partially observable probabilistic systems. Real-Time Systems **53**(3), 354–402 (May 2017)
31. Paterson, C.A., Calinescu, R.: Observation-enhanced QoS analysis of component-based systems. IEEE Trans. on Software Engineering (2019). https://doi.org/10.1109/TSE.2018.2864159, (Early Access)
32. Paterson, C., Calinescu, R., Wang, D., Manandhar, S.: Using unstructured data to improve the continuous planning of critical processes involving humans. In: 14th Int. Symp. on Software Engineering for Adaptive & Self-Managing Systems (2019)
33. Pathak, S., Pulina, L., Tacchella, A.: Verification and repair of control policies for safe reinforcement learning. Applied Intelligence **48**(4), 886–908 (Apr 2018)
34. Robu, V., Flynn, D., Lane, D.: Train robots to self-certify as safe. Nature **553**(7688), 281–281 (2018)
35. Saxena, A., Roychoudhury, I., Celaya, J., Saha, B., Saha, S., Goebel, K.: Requirements flowdown for prognostics and health management. In: Infotech@Aerospace. American Institute of Aeronautics and Astronautics (2012)
36. Spotnitz, R.: Simulation of capacity fade in lithium-ion batteries. Journal of Power Sources **113**(1), 72–80 (Jan 2003)

37. Stetco, A., Dinmohammadi, F., Zhao, X., Robu, V., Flynn, D., Barnes, M., Keane, J., Nenadic, G.: Machine learning methods for wind turbine condition monitoring: A review. Renewable Energy **133**, 620 – 635 (2019)
38. Traub, L.W.: Calculation of constant power lithium battery discharge curves. Batteries **2**(2) (2016)
39. Wognsen, E.R., Hansen, R.R., Larsen, K.G.: Battery-aware scheduling of mixed criticality systems. In: Leveraging Applications of Formal Methods, Verification and Validation. Specialized Techniques and Applications. LNCS, vol. 8803, pp. 208–222. Springer Berlin Heidelberg, Berlin, Heidelberg (2014)
40. Zhang, C., Allafi, W., Dinh, Q., Ascencio, P., Marco, J.: Online estimation of battery equivalent circuit model parameters and state of charge using decoupled least squares technique. Energy **142**, 678 – 688 (2018)
41. Zhang, F., Liu, G., Fang, L., Wang, H.: Estimation of battery state of charge with H_∞ observer: Applied to a robot for inspecting power transmission lines. IEEE Transactions on Industrial Electronics **59**(2), 1086–1095 (Feb 2012)
42. Zhao, X., Robu, V., Flynn, D., Dinmohammadi, F., Fisher, M., Webster, M.: Probabilistic model checking of robots deployed in extreme environments. In: The 33rd AAAI Conf. on Artificial Intelligence (In Press). Honolulu, Hawaii, USA (2019)