Safety assurance challenges for autonomous drones in Underground Mining Environments*

Philippa Ryan[1][0000−0003−1307−5207], Arjun Badyal[1][0000−0002−1868−3343], Samuel Sze[2], Benjamin Hardin[2][0009−0000−9938−3336], Hasan Bin Firoz[1][0009−0007−1198−1018], Paulina Lewinska[1][0000−0002−8141−754X], and Victoria Hodge[1][0000−0002−2469−0224]

[1] Department of Computer Science, University of York, UK
{philippa.ryan, arjun.badyal, hasan.binfiroz, paulina.lewinska, victoria.hodge}@york.ac.uk
[2] Oxford Robotics Institute, University of Oxford, UK
{samuels, bhardin}@robots.ox.ac.uk

Abstract. Autonomous drones have been proposed for many industrial inspection roles including building infrastructure, nuclear plants and mining. They have the benefit of accessing hazardous locations, without exposing human operators and other personnel to physical risk. Underground mines are extremely challenging for autonomous drones as there is limited infrastructure for Simultaneous Localisation and Mapping (SLAM), for the drone to navigate. For example, there is no Global Navigation Satellite System (GNSS), poor lighting, and few distinguishing landmarks. Additionally, the physical environment is extremely harsh, affecting the reliability of the drone. This paper describes the impact of these challenges in designing for, and assuring, safety. We illustrate with experience from developing an autonomous Return To Home (RTH) function for an inspection drone. This is initiated when the drone suffers a communications loss whilst surveying newly excavated corridors that are unsafe for personnel. We present some of the key safety assurance challenges we faced, including design constraints and difficulties using simulations for validation and verification.

Keywords: Autonomous drones · Safety assurance · Underground Mining

1 Introduction

Recent advances in autonomous drone capabilities, their cost-effectiveness and manoeuvrability have allowed drones to be used for complex and challenging missions. These include search and rescue, inspecting buildings, bridges and tunnels, mapping a nuclear emergency, and inspecting challenging environments

* Supported by the Assuring Autonomy International Programme (a partnership between Lloyd's Register Foundation and the University of York) and UKRI (under Grants No EP/R025479/1 and EP/V026801/1)

such as underground mines [22]. Drones have the benefit of being able to access hazardous locations whilst avoiding difficult terrain, and without exposing human operators or inspectors to physical risk. Underground environments such as mines can be particularly challenging for drones. For example, there is no GNSS, poor lighting, and a lack of distinguishing landmarks for SLAM [18]. Also, the physical environment is hot and dusty affecting the reliability of drone navigation systems. There is the risk of explosion in a mine due to the presence of methane and combustible coal dust. Drones have LiPo batteries which can suffer thermal runaway and ignite methane or coal dust, hence allowing the drone to crash even in an unpopulated area of the mine could pose a significant risk.

This paper describes some of these safety assurance challenges in more detail, illustrating with an autonomous "return to home" (RTH) function used to recover an inspection drone in an underground mine. The RTH is a failsafe response that activates in the event of a failure such as loss of communication with the human pilot to ensure the drone can be recovered safely.

This paper is laid out as follows. Section 2 contains related literature. In section 3 we describe our approach to developing a safety assurance case. Section 4 describes our initial model based design solution, and how it implemented some derived safety requirements. In section 5 we describe some issues we encountered using simulations to support the assurance case. Finally, we present final conclusions and future work in section 6.

2 Related Literature

We have not found any papers in the literature that specifically developed a safety case for autonomous drones in underground mines. A safety case, also known as a safety assurance case, is documented body of evidence which provides a convincing and valid argument that a specified system is safe for a given application in a given context or environment [2]. A small number of papers have developed safety case artefacts for above-ground drone missions. Coldsnow et al. [10] assured beyond visual line-of-sight (BVLOS) human-piloted operations by developing a safety case to acquire a BVLOS certificate for flying in a bounded airspace.

Many papers focus on risk assessment and management for above-ground drone missions. Valapil et al. [23] used a formal model to identify risk during flights and the impact of mitigations on those risks. Barr et al. [4] performed a preliminary risk analysis using a probabilistic model-based approach. Aslansefat et al. [3] combined fault tree analysis (FTA) with "complex basic events" which update the fault tree in real-time to enable real-time reliability evaluation and risk assessment. The authors focused on sensor reliability with ML reliability as future work. Chowdhury [7] used reliability modelling and failure modes and effects analysis (FMEA) coupled with minimum Bayes risk analysis to estimate the conditional risk probability during missions.

Other authors elicit safety requirements and argumentation for above-ground drone missions. Clothier et al. [9] produced a risk informed safety case for drone operations that identified risk mitigations (commensurate with the risk level) and argumentation to support and assure them. Hodge et al. [12] developed map-free drone navigation using deep reinforcement learning and produced a functional failure analysis (FFA) with safety case and rationale for failure mitigations. Similarly, Shafiee et al. [20] performed FFA for a drone inspecting an offshore wind farm with safety testing and verification in a laboratory. Rahimi et al. [19] developed an approach for highlighting safety assumption mismatches (traceability) as safety-critical drone products are developed. Cleland et al. [8] also used traceability to identify change as safety-critical drone products are developed using Agile methods.

3 Safety Assurance Approach

3.1 Mine Scenario

Autonomous drones flying and navigating in uncertain environments such as mines may require map-free navigation [12] and iterative replanning of the flight path (run-time adaptation) [14]. The autonomous drone must mitigate the inherent risks, adapt to the environment and be robust to degraded communications [21] commonly experienced in mines. Crucially, the missions need to be safe - this means reducing risk to safety of humans and the environment itself, and providing evidence/assurance we have done so.

In this scenario, a human-piloted drone is inspecting a newly excavated section of mine that is unsafe for humans to enter. During the inspection, the drone loses communication with the human pilot. It is unsafe for the pilot to attempt to recover the drone and leaving the drone in the newly excavated area is hazardous as it presents an explosion risk, and may become damaged attempting to land on the uneven floor surface. Routine loss of equipment would not be acceptable. Hence, the drone must autonomously and safely return to home.

3.2 Safety Assurance Case

The main aim of the safety case is to demonstrate that risk of hazardous behaviour is reduced to a level that is at least considered tolerable (measured in terms of likelihood and severity) and further it has been reduced As Low As Reasonably Practicable (ALARP), in line with UK health and safety at work regulations [11]. Our strategy is to demonstrate that the drone is *designed* to be operated safely, and then that it *is* being operated safely. The former is demonstrated via formal modelling that safety requirements are implemented (section 4.2) and simulation-based testing to validate the performance of the RTH function (section 5). The latter is addressed by personnel adhering to operational procedures derived from system safety analysis (section 4.1).

Operational Design Domain. In keeping with what is becoming typical practice for safe autonomous systems, we first developed a Operational Design

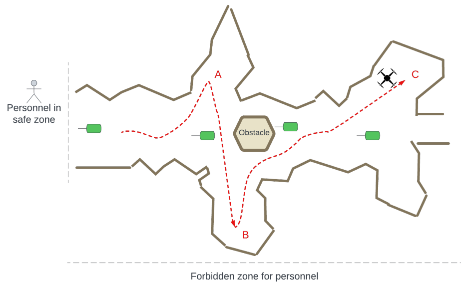

Fig. 1. Top down view of ODD in the mine

Domain (ODD) description in order to understand the context of use and potential safety risks in more detail [15].

The drone will be flying in an unmanned area, surveying newly blasted side tunnels from one main corridor (30-35 by 10-15 feet), all with uneven floors and walls. The tunnels are inspected for safety and viability for mining using laser scans, infrared and visual inspection. Some debris and continuing rock falls can be expected so the tunnels may not be identical when flying to and from an inspection site. No GNSS will be available. The tunnels will include reflective survey markers from the roof a few metres apart (e.g., 5m and 6m), with markers visible at all times from each branch tunnel to be inspected. No survey markers will be in the side tunnels in the mine, as these are unsafe newly blasted areas.

A top down representation of the ODD of a tunnel system is shown in Fig. 1. Drone operators and other staff will be separated from the drone inspection zone (described as the safe zone, with the survey tunnels being the forbidden zone). The figure shows ceiling markers roughly arranged along the central corridor, and a large obstacle that needs to be avoided. An example drone flight path to inspect three side tunnels during manual control is shown with the red dotted line (from start, A, B to C). If communications are lost with the drone, it should return autonomously to the initial starting point, avoiding collision with the obstacle and the tunnel walls. Landing at the location where communications were lost is not a viable solution, due to the risk of damaging the equipment and of explosion, and due to the difficulty in retrieving the drone from unsafe newly blasted tunnels.

As noted, the mine is an extremely dusty (particularly on the floor), dark, hot, harsh environment, which impacts reliability even on a short flight. For example, the dust will be caught in rotor systems, which may lead to drift during flight. Small fragments of rock may fall frequently. Flights are a maximum of approximately 15 minutes due to the harsh environment and limited battery life of the drone.

Hazard analysis Having defined the ODD, we then considered the main accidents, hazards, and risk acceptance criteria, to support our safety assurance case [2]. We focused on risks associated with the RTH function. Accidents range

Table 1. Hazards the RTH function can contribute to

ID	Hazard	Comment
H1	Flying too close to person or moving object	Operational procedures should reduce risk of co-location, but risk can remain
H2	Controlled flight into surface or stationary object	Where drone has lost situational awareness in relation to surfaces or fixed objects, or makes a mistake sending flight commands.
H3	Loss of control	Typically internal failures (e.g., broken propeller, or poor course calculations) that can cause a collision.
H4	Incursion outside of forbidden zone	To avoid populated areas of the mine, or areas without communication / waypoint.

in severity from death to minor injuries, and include loss of equipment. The hazards related to the RTH function are in Table 1. We differentiated between moving and stationary objects, with the former largely referring to mine personnel, but could include, for example, falling debris or other movable equipment. For the purposes of our experimental system, and for this paper, our principle concern was *H2 - Controlled flight into surface or stationary object.* This could occur due to the RTH function failing to localise itself, calculating incorrect control commands and/or failing to compensate for physical problems with the drone rotors, hence flying into the walls or obstacles.

3.3 Design and Assurance Challenges

The following design and assurance challenges were identified.

Design: Physical limitations of the drone. The inspection drone used for the project was a Holybro PX4 [13] autonomous drone kit. Although the drone has a camera and infrared sensor attached, this is fixed facing forward, meaning that using visual comparisons from one direction would not work for the opposite. This would be a problem for localisation. Performing a mapping run from multiple directions is not practical as there is limited battery life, and it could not be guaranteed to complete without loss of communications. It's hard to add additional equipment to the drone as extra weight further limits battery life.

Design: External environment limitations. As there is no GNSS, waypoints cannot be calculated from defined positions. An alternative approach is to navigate via a combination of distinctive landmarks and the reflective ceiling markers. However, the reflective material is difficult for depth and distance sensing via imaging. Future work will look at these challenges.

Assurance: Simulations. Extensive testing in the mine is not practical, due to interference with day to day operations, and the inevitable wear and tear on equipment. Instead, our main options are the use of dedicated test spaces and simulations. Simulations have the advantage of being able to examine multiple scenarios, including failures, environment changes and extreme conditions. However, it is often difficult to justify differences between the simulation and the real world, and predict how these could reduce confidence in safety case evidence. Some specific challenges are described in section 5.

Assurance: Real-world testing. Physical testing is also required, but is limited to simple, non-destructive, scenarios for reasons of cost, safety and effort. Further, our test space does not replicate the harsh conditions of the mine. This would undermine our confidence in validity of test flights in predicting performance in the mine.

4 Proposed solution

The design challenges for this scenario limited our options, particularly for SLAM and navigation. Hence, our initial version of the RTH function simply attempts to reverse the exact course taken from take-off up until the loss of communications. Whilst this is not an efficient method (a longer path than necessary would be taken, returning in and out of the blasted side tunnels), it has the advantage of taking a path that was known to avoid obstacles. The implementation does this by continuously recording operator commands for roll, pitch, yaw, speed, etc. and then attempting to appropriately invert these. For example, in Fig. 1 the drone has travelled to junctions A, B and C around the obstacle. The reverse course ensures that this can be avoided (travelling C $\rightarrow$ B $\rightarrow$ A), and is a safe route at the point the RTH is activated. However, the obvious problems are that there may be new obstacles, and the drone will suffer from drift, for example, from dust interfering with the propellers, drafts or from artefacts in inertial measurement unit (IMU) data. The next stage in our safety process is to derive safety requirements (DSRs) to be implemented to manage these issues.

4.1 Safety Analysis

For our initial design, we performed both a high-level system hazard analysis, and a low-level guideword-based analysis (using [16]) of the RTH function. For reasons of space, small illustrative excerpts are presented here.

System level hazard analysis. System level hazard analysis considers means to reduce risk of each hazard in turn. For example, when considering Hazard H2 (table 1) we are concerned preventing ways the drone could fly into a surface or object without any mitigating action. To reduce the risk of this leading to accidents a number of DSRs were suggested, including:

- DSR1: No new man made objects added to the inspection area during inspection flight.
- DSR2: No other objects can be deliberately moved during inspection flight.
- DSR3: Utilise visual scanning of survey markers and environmental features to support SLAM
- DSR4: RTH function will detect and adapt course to objects in path.
- DSR5: If no RTH path can be calculated, drone attempts to move to nearest survey marker, initiates hover, then lands if no recovery of communications and max operating time met.

Table 2. Extract of safety analysis of RTH function

Guideword	Cause	Mitigation
Omission - Route command missed from set of commands	Command was not recorded Set of commands missed one or more data points Command is not required due to change in corridor/object layout Command queue processing error	DSR.RTH1: Sanity check on route commands DSR.RTH2: Timestamps used on recorded route commands and checked for ordering DSR.RTH3: Hover mode initiated when uncertain of current status

DSR1 and DSR2 are procedural requirements, and would be evidenced in the "is being operated safely" part of the safety case e.g., through training of staff. DSR3-DSR5 are design requirements.

Safety analysis of RTH function. The guideword based analysis considers how internal faults inside components can contribute to risk, and is another source of DSRs. An extract is shown in Table2.

4.2 Control Software Design

In this section, we describe how we designed a formal model of the RTH control software, particularly focusing on the implementation of the DSRs from the high and low-level analyses. We have used a model-based approach, and in particular the RoboSim notation to model the control software [6] and implemented code based on that model. There are several advantages to using the model-based approach of RoboSim. It can support the maintenance of the safety case and analysis. We could, for example, perform automatic code generation to obtain consistent simulation or automatic test case generation, test with a model checker, and carry out formal proofs of correctness [5]. The availability of all these approaches support our case.

In this work, we have used an iterative approach to improve the model, software and DSRs in the safety case, as described in Fig. 2.

In RoboSim, the formalisation process begins with writing the services provided by the robotic platform to the control software to support the DSRs (Fig. 5). Next, the relation between any parallel state machines within a controller is described as shown in Fig. 4. Finally, the details of the state machines defining

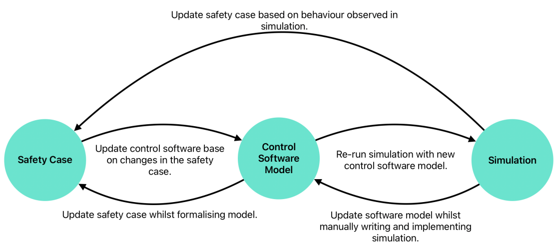

Fig. 2. Bi-directional workflow for improving the control software and the safety case.

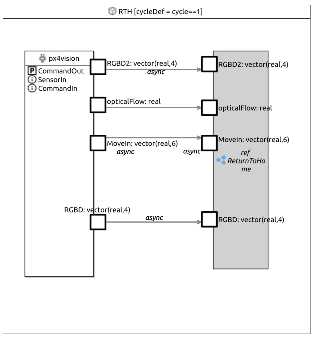

Fig. 3. RoboSim module relating the services of the drone (LHS) with the *ReturnToHome* controller(RHS). The connections represented by the arrows describe the data flow from the services of the drone to the controller.

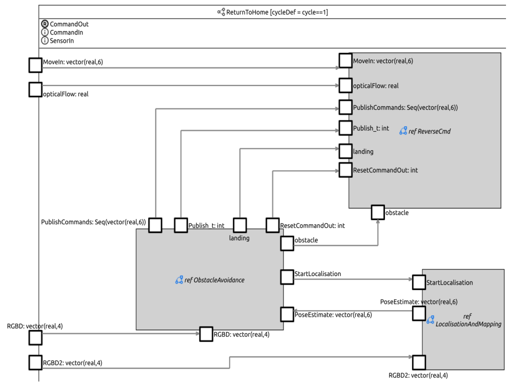

Fig. 4. Control software for the drone, demonstrating the data flow between the *ReverseCmd*, *LocalisationAndMapping*, and *ObstacleAvoidance* state machines (the grey boxes). In particular, several events (the black-bordered boxes) are used within the controller to communicate between the state machines.

the behaviour of the controllers are described, as in Fig. 5. When formalising the software, there have been several steps taken to update the DSRs at each level of abstraction. For example, we are forced to state precisely which services of the drone are used by the control software; this has informed the scope of our DSRs. The DSRs have been integral in the development of the control software. For example, DSR5 presents a clear account of the control flow in a situation where no path can be calculated by the obstacle avoidance state machine. This is reflected in the transition between the *ObstacleAvoidance* and Landing states in the *ReverseCmd* state machine in Fig. 5.

We specified the details of the *ReverseCmd* state machine, however, writing the model allowed us to identify more clearly other aspects of the model such as localisation and mapping and obstacle avoidance, as shown in Fig. 4. Members of the team implementing the software for each controller could refer to the model for a clear indication of the inputs and outputs required by them. This is an area often overlooked in the modelling process. A formal account of what is not known with an indication of how to achieve these unknowns in a precise manner is invaluable to any future adaptations of the model and its implementation.

Once a model is written, it can be translated into a form that can be executed as part of a simulation. In our system, we used the *ReverseCmd* state machine to write a Python node in ROS2 [17] which communicates with the wider simulation (in Gazebo, see Section 5). One of the inherent advantages of this approach is its modularity, as our software artefacts can be re-used in the deployment of the drone or elsewhere without modification. In writing and executing the simulation code, the control software model has been frequently updated due to

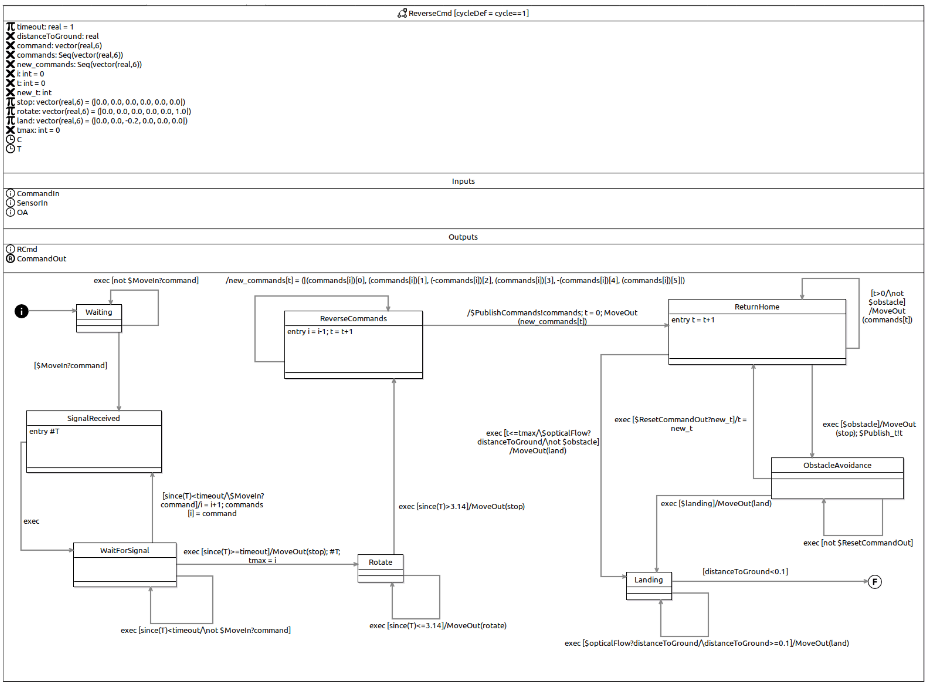

Fig. 5. ReverseCmd state machine. This state machine formally describes how the state of the control software changes depending on events (e.g. MoveIn) or conditions (e.g. since(T) $<$ timeout). It also describes when operations are to be sent to the drone e.g. MoveOut.

observed incorrect behaviour. For example, the drone initially did not reverse the commands due to incorrect logic in the model's ReverseCommands state. We therefore, updated the model to fix this. Whilst it is true that in this process both the code and the model have to be updated, it is much easier to update the model and resolve issues at the model level first. Since it is a precise formal model, it has been relatively low-cost to translate back to our low-level source-code implementation. Finally, the safety case has been updated based on the results of the simulation. In our case, the drift of the drone in simulation reinforced the importance of the inclusion of SLAM, for example, in DSR3.

5 Evaluation: Updating The Safety Case and Model

A key part of our safety approach is the use of simulations to provide assurance in the performance of the drone over many varied scenarios [3] Fig. 6. Simulation has the advantage of testing the drone in extreme situations without loss of

[3] The simulation source-code, model, and video are available at `https://github.com/uoy-research/UAV_Hackathon_T3`

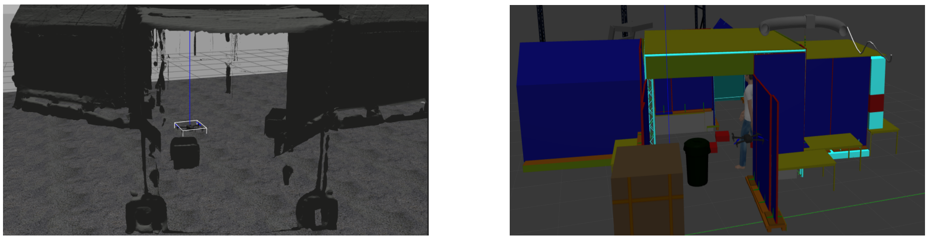

Fig. 6. Screenshots from the simulations

equipment. We use our observations to update and evaluate our safety case considering the ODD, DSRs, physical reality-gap, reliability, and repeatability.

Is the ODD adequately represented? A physical mockup of the mine tunnels was produced and our simulation layout used mesh files of these, including representations of floor objects (Fig: 6). It accurately represented the physical proportions of the deployment environment, but it suffers from physical considerations (e.g. collision detection was not used for tunnel walls and the model does not include unexpected drafts). Nevertheless, this was felt to be sufficient to observe the behaviour of the drone in respect to its environment for early design improvements, but would undermine confidence in any final stage design testing and reduce our ability to automate tests. A physical model of the drone and the environment, and its relation to the physical behaviour of the simulation would be required to quantify how well these aspects of the ODD are reflected [1]

Are the DSRs reflected? No additional objects are spawned in simulation which ensures DSR1. Some man made objects may move in an unnatural way due to the physics engine which may violate DSR2. DSRs 3-5 were not observed in simulation. As mentioned in 4.2 the state machines in the model need to be updated to reflect these requirements.

Is the physical performance of the drone representative? The PX4 model of the drone contains all the key sub-systems necessary to test the RTH function in its current implementation, but we made a number of simplifications to improve the performance of the simulator. For example, we used camera messages from the Gazebo model, rather than a model of the drone camera. It would be necessary to ensure that images from this simulation were sufficient to test any future obstacle avoidance algorithm implemented using visual object detection. Other internal performance aspects, such as natural drift in the IMU or compute time taken to respond to commands, were not considered. Nor are there any health monitoring functions and other key sensor and actuator data to test robustness to inputs. The ODD and DSRs may be updated to more precisely formalise performance requirements of the drone.

Is the simulation repeatable? Our Gazebo runs were not repeatable (repeatability is a typical expectation for robust safety assurance). Additionally, the non-deterministic nature of the gazebo simulations limited our investigation. This non-determinism could be due to a number of factors such as the use of ROS, or the accumulation of error in the computation pipeline of the physics engine. Therefore, we could validate the state machine's general behavior but not the coverage over all trajectories.

Does the simulation reliably model the elements as configured? No quality assurance of the simulation tools (the combination of PX4, Gazebo, and ROS2) we used are available, therefore we do not have full confidence that the behaviour we specify is the behaviour the tool represents.

6 Summary and future work

When developing our RTH function we faced many challenges in developing a feasible solution, and, most crucially from a safety perspective, in assuring the design. Using the RoboSim framework gave us access to strong formal approaches to support software verification and validation. However, we also need assurance in the drones performance via dynamic testing. We used simulations for this purpose as they can cover many scenarios (including situations hard to replicate in the real-world) but found many issues which undermined confidence in the results. This meant that although simulations were useful for early feedback and safety requirements development, they may provide limited assurance for a mature design.

Future work should look at improving the control software model. Models of the drone and its environment should also be developed. The implementation can then be further refined based on these models and any changes to the safety case.

References

1. A. Miyazawa, et al., A.: RoboSim Physical Modelling Reference Manual. Technical report, University of York (Nov 2020)
2. ACWG: Assurance Case Guidance "Challenges, Common Issues and Good Practice". Tech. Rep. SCSC-159 v1.0, Safety Critical Systems Club (2021), `https://scsc.uk/scsc-141C`
3. Aslansefat, K., Nikolaou, P., et al.: Safedrones: Real-time reliability evaluation of uavs using executable digital dependable identities. In: Seguin, C., Zeller, M., Prosvirnova, T. (eds.) Model-Based Safety and Assessment. pp. 252–266. Springer International Publishing (2022)
4. Barr, L.C., Newman, R., et al.: Preliminary risk assessment for small unmanned aircraft systems. In: 17th AIAA Aviation Technology, Integration, and Operations Conference. p. 3272 (2017)
5. Cavalcanti, A., Barnett, W., et al.: RoboStar Technology: A Roboticist's Toolbox for Combined Proof, Simulation, and Testing, pp. 249–293. Springer International Publishing, Cham (2021)
6. Cavalcanti, A., Sampaio, A., et al.: Verified simulation for robotics. Science of Computer Programming **174**, 1–37 (2019)

7. Chowdhury, A.: Dynamic Risk Assessment of Unmanned Aerial Vehicles (UAVs). Master's thesis, Department of Mechanical Engineering, University of Alberta, Canada (2023)
8. Cleland-Huang, J., Agrawal, A., et al.: Visualizing Change in Agile Safety-Critical Systems. IEEE Software **38**(3), 43–51 (2020)
9. Clothier, R., Denney, E., Pai, G.J.: Making a risk informed safety case for small unmanned aircraft system operations. In: 17th AIAA Aviation Technology, Integration, and Operations Conference. p. 3275 (2017)
10. Coldsnow, M.W., Glaab, L.J., et al.: Safety Case for Small Uncrewed Aircraft Systems (sUAS) Beyond Visual Line of Sight (BVLOS) Operations at NASA Langley Research Center. Tech. rep., NASA Langley Research Center, Hampton, VA, USA. (No. NASA/TM–20230003007) (2023)
11. Health and Safety Executive: Health and Safety at Work Regulations, `https://www.legislation.gov.uk/uksi/1999/3242/regulation/3` (1999)
12. Hodge, V.J., Hawkins, R., Alexander, R.: Deep reinforcement learning for drone navigation using sensor data. Neural Computing and Applications **33**, 2015–2033 (2021)
13. Holybro: PX4 Vision Dev Kit V1.5. `https://holybro.com/products/px4-vision-dev-kit-v1-5` (2024)
14. Imrie, C., Howard, R., et al.: Aloft: Self-adaptive drone controller testbed. In: 19th International Conference on Software Engineering for Adaptive and Self-Managing Systems (SEAMS'24) (2024)
15. Kaakai, F., Adibhatla, S.e.a.: Data-Centric Operational Design Domain Characterization for Machine Learning-Based Aeronautical Products. In: Computer Safety, Reliability, and Security: 42nd International Conference, SAFECOMP 2023, Toulouse, France, September 20–22, 2023, Proceedings. p. 227–242. Springer-Verlag, Berlin, Heidelberg (2023)
16. McDermid, J.A., Nicholson, M., Pumfrey, D.J., Fenelon, P.: Experience with the application of HAZOP to computer-based systems. In: IEEE proceedings of the 10th Conference on Computer Assurance Systems Integrity, Software Safety and Process Security. pp. 37–48 (1997)
17. Open Robotics: ROS2 documentation. `https://docs.ros.org/en/foxy/index.html` (2024)
18. Park, S., Choi, Y.: Applications of Unmanned Aerial Vehicles in Mining from Exploration to Reclamation: A Review. Minerals **10**(8) (2020)
19. Rahimi, M., Xiong, W., et al.: Diagnosing assumption problems in safety-critical products. In: 2017 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE). pp. 473–484 (2017)
20. Shafiee, M., Zhou, Z., et al.: Unmanned aerial drones for inspection of offshore wind turbines: A mission-critical failure analysis. Robotics **10**(1), 26 (2021)
21. Shahmoradi, J., Roghanchi, P., Hassanalian, M.: Drones in underground mines: challenges and applications. In: 2020 Gulf Southwest Section Conference. ASEE (2020)
22. Shakhatreh, H., Sawalmeh, A., et al.: Unmanned Aerial Vehicles (UAVs): A Survey on Civil Applications and Key Research Challenges. IEEE Access **7**, 48572 – 48634 (04 2019)
23. Valapil, V.T., Herencia-Zapana, H., et al.: Towards Formalization of a Data Model for Operational Risk Assessment. In: 2021 IEEE/AIAA 40th Digital Avionics Systems Conference (DASC). pp. 1–10 (2021)