Proceedings Paper:
Imrie, Calum Corrie, Howard, Rhys, Thuremella, Divya et al. (8 more authors) (2024) Aloft: Self-Adaptive Drone Controller Testbed. In: SEAMS '24: Proceedings of the 19th Symposium on Software Engineering for Adaptive and Self-Managing Systems. 19th International Conference on Software Engineering for Adaptive and Self-Managing Systems, 15-16 Apr 2024 ACM, PRT.

https://doi.org/10.1145/3643915.3644107

Aloft: Self-Adaptive Drone Controller Testbed

Calum Imrie
Dept. of Computer Science
University of York
York, UK
calum.imrie@york.ac.uk

Rhys Howard
Oxford Robotics Institute
University of Oxford
Oxford, UK
rhyshoward@live.com

Divya Thuremella
Oxford Robotics Institute
University of Oxford
Oxford, UK
divya@robots.ox.ac.uk

Nawshin Mannan Proma
Dept. of Computer Science
University of York
York, UK
nawshinmannan.proma@york.ac.uk

Tejas Pandey
Dept. of Computer Science
University of York
York, UK
tejas.pandey@york.ac.uk

Paulina Lewinska*
Dept. of Computer Science
University of York
York, UK
paulina.lewinska@york.ac.uk

Ricardo Cannizzaro
Oxford Robotics Institute
University of Oxford
Oxford, UK
ricardo@robots.ox.ac.uk

Richard Hawkins
Dept. of Computer Science
University of York
York, UK
richard.hawkins@york.ac.uk

Colin Paterson
Dept. of Computer Science
University of York
York, UK
colin.paterson@york.ac.uk

Lars Kunze
Oxford Robotics Institute
University of Oxford
Oxford, UK
lars@robots.ox.ac.uk

Victoria J. Hodge
Dept. of Computer Science
University of York
York, UK
victoria.hodge@york.ac.uk

ABSTRACT

Aerial drones are increasingly being considered as a valuable tool for inspection in safety critical contexts. Nowhere is this more true than in mining operations which present a dynamic and dangerous environment for human operators. Drones can be deployed in a number of contexts including efficient surveying as well as search and rescue missions. Operating in these dynamic contexts is challenging however and requires the drones control software to detect and adapt to conditions at run-time.

To help in the development of such systems we present Aloft, a simulation supported testbed for investigating self-adaptive controllers for drones in mines. Aloft utilises the Robot Operating system (ROS) and a model environment using Gazebo to provide a physics-based testing. The simulation environment is constructed from a 3D point cloud collected in a physical mock-up of a mine and contains features expected to be found in real-world contexts.

Aloft allows members of the research community to deploy their own self-adaptive controllers into the control loop of the drone to evaluate the effectiveness and robustness of controllers in a challenging environment. To demonstrate our system we provide a self-adaptive drone controller and operating scenario as an exemplar. The self-adaptive drone controller provided utilises a two-layered architecture with a MAPE-K feedback loop. The scenario is an inspection task during which we inject a communications failure. The aim of the controller is to detect this loss of communication and autonomously perform a return home behaviour. Limited battery life presents a constraint on the mission, which therefore means that the drone should complete its mission as fast as possible. Humans, however, might also be present within the environment. This poses a safety risk and the drone must be able to avoid collisions during autonomous flight.

In this paper we describe the controller framework and the simulation environment and provide information on how a user might construct and evaluate their own controllers in the presence of disruptions at run-time.

CCS CONCEPTS

• **Software and its engineering** → *Software libraries and repositories*; • **Applied computing** → Earth and atmospheric sciences.

KEYWORDS

Self-adaptive systems, unmanned aerial vehicles, controller framework testing, mining operations

*Also with AGH University of Krakow, al. A. Mickiewicza 30 30-059 Krakow, Poland.

SEAMS '24, April 15–16, 2024, Lisbon, AA, Portugal

ACM ISBN 979-8-4007-0585-4/24/04…$15.00
https://doi.org/10.1145/3643915.3644107

ACM Reference Format:
Calum Imrie, Rhys Howard, Divya Thuremella, Nawshin Mannan Proma, Tejas Pandey, Paulina Lewinska, Ricardo Cannizzaro, Richard Hawkins,

Colin Paterson, Lars Kunze, and Victoria J. Hodge. 2024. Aloft: Self-Adaptive Drone Controller Testbed. In *19th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS '24), April 15–16, 2024, Lisbon, AA, Portugal.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3643915.3644107

1 INTRODUCTION

The goal of this artifact is to facilitate the development of adaptive aerial autonomous systems capable of operating in challenging or dangerous environments. Additionally a focus on the safety of humans, protection of the environment, and the assured performance of the autonomous agents. To achieve this, we provide a simulation environment within which to develop and test autonomous behaviours, and analyse an example scenario in which an autonomous drone must safely perform a return-to-home in response to a loss of signal event in a mine while avoiding collisions with the environment and humans. We emphasise that although we provide a mine-based environment, our aim is a generalised testbed that may support research into self-adaptive autonomous systems in general.

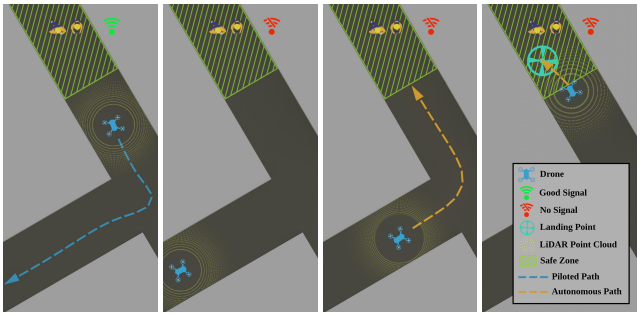
Exploration [13] and inspection of underground mines [13], or search and rescue [14, 16, 23] using drones needs to be safe. Drones are a cost-effective tool for these tasks. The drone's autonomous navigation must mitigate the inherent hazards and risks, be able to adapt to the dynamic environment and be robust to the limited or degraded communications [17] commonly experienced in mines. There are a number of drone navigation approaches described in the literature. These can be subdivided into model-based (classical) and model-free (data-based) approaches. Model-based approaches include the extended Kalman filter (EKF) [20] and vector field histogram [22]. Model-free approaches include approaches such as deep learning [11] and deep reinforcement learning [7, 11]. Any of these can be incorporated into the simulation to evaluate their effectiveness and ability to adapt.

Our exemplar scenario consists of a human-piloted drone being used for a surveying task, during which the drone loses pilot control signal and must safely navigate home and land. The simulation is a two-layered architecture, with a managed system (application) and a managing system (self-adaptation). The Aloft framework facilitates the simulation of normal behaviours, failures and adaptive behaviours of drones at runtime. It provides a 3D physics-based simulation from which the user can create different simulation worlds and drone models, and can analyse the status of the drone's system and tasks. It allows the implementation and evaluation of self-adaptive drone controllers that safely respond to failures and environment changes. Aloft allows runtime data collection to facilitate post-flight analysis.

Our paper's main contributions are as follows:

(1) An identification of challenges of mine-based autonomous drone flight and subsequent need for a self-adaptive controller
(2) 3D mock-mine models constructed from real-world mine point cloud data and real-world laboratory mock-mine designs
(3) An artifact, containing a complete software setup, used to facilitate development of self-adaptive drone controllers
(4) An exemplar return-to-home adaptive drone controller

Sec. 2 gives an overview of drone applications in mines followed by a motivating example of self-adaption in this context. The simulation provided in Aloft is described in Sec. 3. A description of an exemplar is given in Sec. 4; we also describe how the user can expand the exemplar's controller and scenario setup to research additional problems. In Sec. 5 we discuss existing work related to autonomous drone adaptation.



(a) Surveying (b) Signal Loss (c) Return Home (d) Landing

Figure 1: An illustration of the return-to-home exemplar problem used to demonstrate our *Aloft* artifact. The drone begins to survey under pilot control (a) but then loses signal (b). It must then autonomously perform a return-to-home function (c) and land safely (d).

2 SELF-ADAPTIVE DRONES IN MINES

Drones provide significant benefits to mining operations; self-adaptive controllers may greatly assist drone operation in this domain. Aloft aims to provide a software setup to allow the self-adaptive systems community to quickly deploy bespoke solutions and/or investigate the safe use of self-adaptive drones in mines.

2.1 Mine-Based Drone Applications

Drones have become a popular system of choice for remote autonomous operation in mines due to their relative ease of operation and ability to manoeuvre complex environments [13]. Drones are particularly beneficial in parts of mines where it would present a risk to human life if people were to enter. There are a few cases where this can be applied: the surveying of newly excavated sections; participation in search-and-rescue in hazardous environments; and safe inspection of potentially unstable supports. The relevance of this work extends to other environments and robot types. Caves closely resemble mines and are of even greater risk due to the lack of human-made support systems and unmonitored effects of water erosion. Likewise, submersibles share a similarity with aerial drones because they also operate in three dimensions, and can be used to explore flooded mine sections or underwater cave networks.

2.2 Motivating Example

In this example a manually-piloted drone is carrying out a volumetric scan survey (Fig. 1a) to analyse excavation efficiency. During the survey the drone loses the pilot control signal (Fig. 1b) in a section of the mine that is unsafe for humans. To recover the drone without human retrieval, the drone should safely and automatically return to home (Fig. 1c). The surveying mission has allowed the drone to generate a map of the mine, which it uses to compute a list

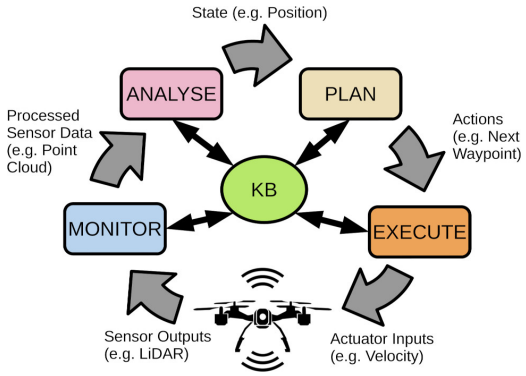


Figure 2: High-level diagram of autonomous drone MAPE-K.

of waypoints to navigate home. Self-adaptation during navigation can safely overcome the adverse conditions of the mine and ensure the drone meets its safety requirements, such as maintaining a safe separation from obstacles. There may also be mine operators present that the drone must detect and safely avoid using its on-board sensors. Finally, the drone should safely land once it reaches the designated safe landing zone (Fig. 1d). In this instance the pilot does not regain control during this return-to-home operation.

2.3 Autonomic Manager Description

In this work we use the MAPE-K approach to describing autonomic management of an element [8]. Here the managed element is comprised of drone hardware and low-level software components that interface with said hardware; see Fig. 2 for a high level interpretation. The MAPE-K autonomic feedback loop consists of cycling Monitor, Analyse, Plan and Execute phases as well as a Knowledge base utilised across them. Here we explain how a self-adaptive drone system relates to these phases.

Monitor: Here the autonomic management system gathers data to guide self-adaptive behaviour. We identify the following as sources the self-adaptive system should monitor:

- Power consumption / battery levels
- Inertial measurement unit (IMU) data
- Camera / LiDAR data
- Temperature sensor data
- RC signal strength

Analyse: In this phase the autonomic management system takes the incoming data and at times from the knowledge base to get an understanding of the state. This stage provides the system the critical information as to whether adaptation is required. Here are some considerations for a self-adaptive drone:

- Determining whether there is a discrepancy between observed data and the drone's world model and updating this model should this become too great
- Checking if the environment has changed since it was mapped; these changes can affect the performance of the system and components if the original plan is still being conducted
- Using computer vision to determine the presence of different objects; these can pose as dangerous obstacles to the drone, as well as in-danger to the drone's operation

Plan: The Plan phase sees the autonomic management system decide a course of action for the managed element and may need to adapt even if the knowledge base is accurate. Below are some considerations a drone must take into account:

- Ensuring that the system adheres to safety constraints by formally verifying that a given course of action does not violate the parameters described in these constraints
- Avoiding the influence of potential causal confounding, which occurs when the Monitor and Plan phases are influenced by a single common source, such as salt build-up in the rotors
- Deciding whether taking a new course of action to better understand the environment in order to reduce risk would be more beneficial. This would be instead of continuing to pursue the mission goal directly

Execute: The Execute phase is where the managed element acts upon the output of the Plan phase. Self-adaptation is still required as the element may need or prefer to react to situations without re-planning. Here are some drone-related examples:

- If there has been a substantial change to the map — described in the Analyse phase — the drone may have to adapt the plan to safely navigate around obstacles while still being able to reach its goal
- In the presence of moving objects (e.g. humans, vehicles, hanging wires), adapting the drone's path ensures sufficient clearance by predicting collision-free trajectories
- Should the Analyse phase determine the world model has significantly changed (e.g. wind, dust), a new generated model would be required to correct the drone's course

Knowledge: The knowledge base contains information relevant to the scene and interactions within it. This information could include:

- The current state of the world; including information no directly observed via the monitoring stage
- A description of the drone mission or a reward metric
- A dynamically updated map of the environment
- A transition model representing the interactions between the drone and its environment
- Generated explanations to determine the reason a given outcome or action was observed

3 ALOFT SETUP

Aloft is packaged with a full simulation setup. This consists of the ROS and Gazebo setup, along with the mock mine models, and modified PX4-vision drone. A qcow2 virtual machine image containing the artifact can be found here: https://github.com/uoy-research/ALOFT.

3.1 Mock Mine

Two mock mine environments are provided with Aloft, see Fig. 3. Each mock mine environment was first constructed physically and then scanned using a terrestrial laser scanner to generate an accurate 3D point cloud. These point clouds were used to construct the simulated mines. The full process can be found on the GitHub repo.

Aloft provides two mine environments to allow users to implement a drone controller and train/validate it in one mock mine

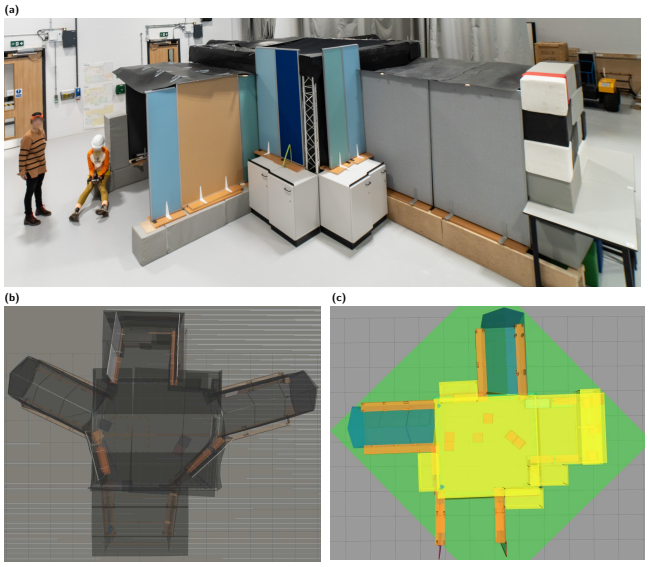


Figure 3: (a) A physical mine constructed. A laser scanner was used to collect the point cloud. Two slightly different physical mock mines were created and scanned (b,c). The physical mine in (a) was to make the simulated mine in (c).

environment, and then test their controller in an unseen mock mine environment. Aloft also contains the collected point cloud from one of the mock mines as this can be used prior to deployment — for example for an initial global navigation plan [5].

3.2 PX4-Autopilot

The target hardware system for the simulation is a PX4 Vision V1.5 quadcopter drone [9]. This drone runs the popular PX4-Autopilot flight controller stack software. PX4-Autopilot is open source with a large development community. It uses the MAVLink messaging protocol to send commands from the flight control unit to the drone's motor controllers. It also transmits system status updates to Ground Control System (GCS) software used to remotely monitor, configure and command the drone. PX4-Autopilot provides the ability for external flight control via software executed on an onboard companion computer. The companion computer interfaces with the flight control unit via MAVLink and the navigation framework implemented via ROS.

ROS is a widely used robotics operating framework [15]. Due to being a middleware software layer, the same ROS software can be run on both real-world robot hardware and simulated robots. ROS provides a development environment where the system's functionality can be adapted and extended rapidly.

Simulation is an easy, fast and importantly, safe, way to test new adaptive drone controller behaviours before attempting to fly in the real world. PX4-Autopilot seamlessly integrates with many simulation tools including Gazebo, a popular 3D simulation environment which integrates easily with ROS. For PX4-Autopilot, Gazebo is particularly suited to testing object-avoidance and computer vision applications [19] due to Gazebo's flexibility of environment set up.

PX4 automatically collects detailed system state and sensor data. To log additional data from ROS-based software, users can use rosbag, which is a tool for recording data being published in ROS.

3.3 Adapting to Dynamic Scenarios

Mine environments are dynamic and often contain wind currents and airborne dust particles. Wind conditions may change quickly and without notice due to activity at air dams, while the exact interactions of the dust with motors and sensors is uncertain and difficult to model in advance. Thus, the drone must adapt online while it operates. This could include, for example, updating its model and understanding of the world, or taking cautious actions if it is known that there are safety critical concerns.

Other challenges include the presence of dynamic obstacles. During the outbound surveying portion of the flight mission, obstacles might have moved. Thus, when returning to home, the drone's map will be outdated and will require adaptive planning.

Changes may happen to the drone itself, including sensor failures. For example, the camera might become dirty due to the airborne dust or the magnetometer may suffer from magnetic interference.

Aloft allows the user to modify various aspects to investigate self-adaptive drone controllers in mine operations. We have categorised these aspects as:

- **Physical concerns**
  - **Environmental**: Intrinsic issues to the mine setting
  - **System**: Components on the drone system that are challenging due to failure and degredation
- **Time related concerns**
  - **Pre-flight**: Issues that could occur prior to the drone's mission
  - **Runtime**: Potential ongoing problems during the mission

It should be noted that the physical and time related concerns are not mutually exclusive. Examples can be found in Table 1.

4 SIGNAL LOSS EXEMPLAR

Aloft contains a setup of the motivating example, described in Fig. 1, where the drone initially performs an operation and loses signal. This now requires the drone to return home autonomously.

4.1 PX4-Vision

The PX4 Vision is equipped with a Structure Core depth camera and Up Core companion computer. A key requirement is to ensure the drone does not collide with any obstacles. We therefore added a bump sensor to the PX4-Vision model. This sensor is not and should not be available to the drone. It is for analysis purposes only.

4.2 Self-Adaptive Architecture

The self-adaptive controller employs a two-layered architecture consisting of a managed system and a managing system. The managed system is concerned with the systems main duties, while the managing system contains the adaptive tasks. The managing system utilises a MAPE-K feedback loop, see Fig. 4.

4.2.1 Managed System. The managed system handles the main function which is a hybrid between the PX4 flight stack and ROS software packages. This hybridisation is achieved using MAVROS, a ROS package that can translate between traditional software using ROS messages and MAVLINK messages. The managed system re-

Physical concerns		Time related concerns	
Environmental	**System**	**Preflight**	**Runtime**
• Pre-built mine environments for immediate use • Hanging obstacles which may swing requiring adaptation of flight plan • Scalar field describing dust density which affects drone motors & sensors, requiring adaptation of drone-environment causal model • Inconsistent lighting conditions require that the system prioritise different sensors / methodologies depending their suitability for a given circumstance	• Ready-to-use drone simulation framework • Fault simulation forcing drone to adapt to be resistant to faults • IMU magnetometer interference simulation requires adaptation to avoid using sensor measurements that may hinder performance • Battery modelling creating a tradeoff between need to complete mission fast and safely • Ability to seamlessly switch between simulated and real drone setups to check adaptability	• Random placement of obstacles requiring adaptable planning • Verification to approximate a Pareto front between mission success metrics • Ability to set epistemic uncertainty of information to help ascertain the ability of the system to decide against exploration for vs exploitation of knowledge • Option to choose between same parameters for repeatability during development vs random parameters for adaptability testing	• Dynamic wind conditions requiring fast adaptation to avoid collision • Utilising causal modelling to offer a system that is robust to the presence of confounders • Collision detection emphasising the avoidance of collisions and appropriate corrective measures taken in the event of a collision • Generation of explanations to help in understanding adaptation failure cases

Table 1: The areas relating to self-adaption that this artifact aims to tackle.

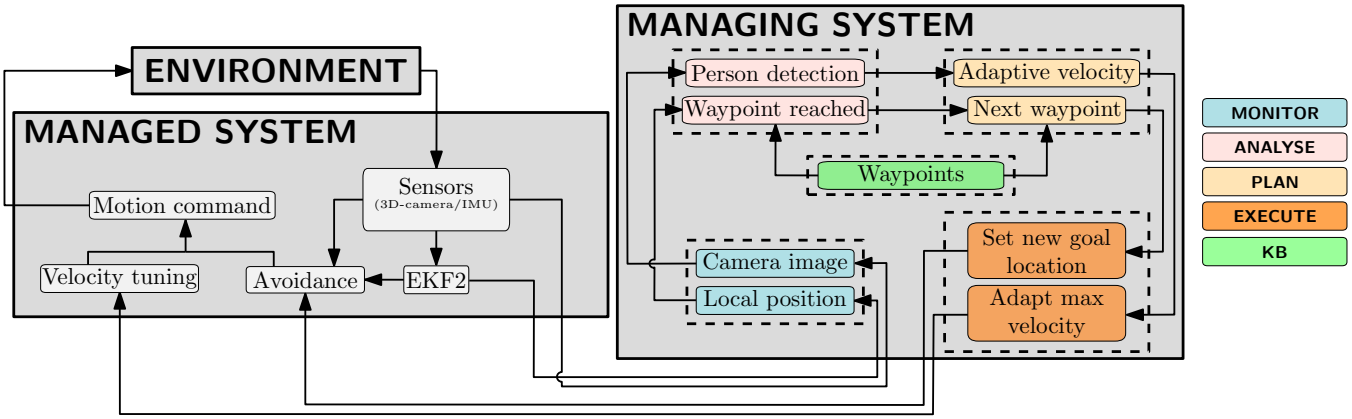


Figure 4: The architecture for the return-to-home behaviour.

ceives sensor information, *Sensors*, from the environment. Primarily this is 3D-camera data, which consists of a RGB image and 3D point data, plus IMU readings. The PX4 uses the IMU readings to localise using the Extended Kalman Filter, *EKF2*. Another deployed package is PX4-Avoidance [21] which when given a navigation goal will attempt to traverse to while avoiding obstacles. PX4-Avoidance uses the vector field histogram (VFH*) [22], *Avoidance*, which uses 3D sensor readings and localisation information to produce intermediate waypoints. MAVROS allows the system to send positions to the PX4 and it will fly quickly to the goal (*Motion command* in Fig. 4). To have control over the maximum velocity *Velocity tuning* ensures that the next step is below a given threshold.

4.2.2 Managing System. The managing system monitors the RGB images provided by *Sensors* and the local position from *EKF2*. The system has a sequence of waypoints it must achieve to return home, and continues to analyse the *Local position* in *Waypoint reached*. This informs *Next waypoint* whether we need to update the current waypoint and if so this is executed in *Set new goal location*, which is then fed into *Avoidance*. Similarly we detect if humans are present with a pretrained *YOLO* module. YOLO [2] (you only look once) is a deep learning network trained to detect objects in an image. The network has been pretrained with the COCO dataset which contains humans as a class. This state estimate of human presence is input into *Adaptive velocity* which determines if the drone needs to change the max speed which is executed by *Adapt max velocity*. Finally this is fed into *Velocity tuning* to inform the drone's actions. The knowledge base consists of the list of waypoints which is used for determining if the PX4-Vision has reached a waypoint, and determining the next waypoint.

4.3 Presence of Mine Operator

A mine operator may be present during the PX4-Vision's return-to-home mission. There is a script included in the Aloft which will randomly place a human in the mine.

In the scenario the drone's battery is constantly depleting. Therefore it is desirable for the drone to return home as soon as possible to avoid low battery problems. Hence this exemplar is concerned with time to complete the mission. A safety requirement is for the PX4-Vision to not have collisions, particularly with any mine operators. Hence the PX4-Vision will need to adapt its maximum velocity if it believes there is a human in the vicinity. Essentially

	Times (s)	
Human presence	**Detect Human**	**Return Home**
Yes	13.6	57.35
No	-	35.1

Table 2: Recorded times for the drone's return home mission.

the PX4-Vision will reduce its maximum velocity if it detects one or more humans with its camera and computer vision.

Due to the signal loss, the PX4-Vision can only interact with the environment and mine operators physically. This is problematic as the PX4-Vision may lose visual on the mine operator due to a variety reasons, including noisy camera images and the mine operator moving out of sight. The PX4-Vision's self-adaptive system therefore assumes that once detected the mine operator is now constantly in the environment and will maintain the lower adapted maximum velocity.

Sample runs of when there was a mine operator present/absent was conducted and reported in Table 2. As expected, the drone takes longer to return home when it has detected a mine operator. A video of a sample run is available on the GitHub repo.

4.4 Expanding the Exemplar

Aloft 's exemplar can be expanded to investigate different topics and themes in self-adaptive systems. As stated in Fig. 4.3 there is a trade off between mission completion time and safety, to both the PX4-Vision and any mine operators present. Navigation in PX4-Vision can be modelled as a Markov Chain (discrete or continuous) using the preset waypoints as states. Transitions can be derived statistically from observing multiple runs. Overall this can enable controller synthesis and generate a Pareto Front which trades-off the multiple objectives, using EvoChecker [6] for example.

Similarly the PX4-Vision ROS network and Gazebo environments can be modified to investigate additional problems, such as those mentioned in Table 1. Wind can be turned on, or lights can be dimmed making it harder for the computer vision. The ROS messages themselves can be perturbed with noise, e.g., to model LiDAR issues the PX4-Vision may experience in the mine.

Aloft can perform in-depth analysis. Automated multiple runs are possible to statistically analyse the system as stated earlier. This process can be useful for the user to compare their own developed self-adaptive controller with the one included in Aloft as a potential baseline. Furthermore, with two mock mines, the user can train on one mock mine and then validate with the other, using the provided controller for evaluation comparison purposes. Currently the PX4-Avoidance typically moves the PX4-Vision so the camera can best plot the next immediate waypoint. An example of a new self-adaptive controller would be the PX4-Vision including the functionality of maintaining visual if it detects a mine operator. This can be perceived as a more cautious controller as the PX4-Vision is ensuring that it has timely information regarding present mine operators location and behaviour.

5 RELATED WORK

Adaptation has been applied to robots generally [1] and more recently to aerial drones [3, 4, 10, 12, 14] and underwater vehicles [18]. These systems frequently use the MAPE or MAPE-K (Monitor-Analyze-Plan-Execute using shared Knowledge) self-adaptation loop [8] to improve reliability. [3] presented MAPE-K runtime adaptive models for human-drone collaboration during emergency response missions which allows the human and machine to work together to achieve the mission goal. The authors note that they still need to aggregate heterogeneous runtime models and align the models' design and implementation in an integrated environment.

Dragonfly [10] is a lightweight MAPE-K drone simulation with runtime adaptation capable of simulating multiple drones in normal and abnormal environmental conditions. It simulates using a 2D gridworld where cells contain obstacles or conditions such as wind. DARTsim [12] is a multi-drone simulator. It treats the team as a single entity and adapts the drones' paths and speeds to maximise the mission's success. It is a C++ library to provide adaptation that can be linked into other simulations programmatically or using a TCP socket. CICADA [14] MAPE-K framework incorporates the PX4 Autopilot flight controller into Gazebo simulation engine. It has a monitoring layer analysing the drone's sensors and parameter settings to monitor the drone and its environment and an adaptation layer to adapt the drone's mission to the prevailing conditions, for example stabilising the drone or landing immediately.

Aloft is a digital twin with a full physics engine and 3D simulation of a real-world mine construction combined with the PX4 Autopilot flight controller. It allows the development of 3D computer vision, collision avoidance and adaptive navigation algorithms to allow the drone to return home safely and not require recovery. It can verify and validate non-functional and functional requirements including safety requirements of drones [10].

Similar to Aloft, SUAVE [18] is a ROS-based simulation that adapts to environmental conditions and system failures. ROS-based systems are node-based and can be easily altered or extended. SUAVE simulates underwater vehicles inspecting pipelines and provides different adaptation frameworks.

6 CONCLUSION

Drones can be of considerable use in mine operations, and self-adaptive systems will play a pivotal role in the drones' operation success and safety. This seems to resonate with the industry's focus on adaptive technologies, hinting at a practical way to navigate real-world complexities. This could potentially enhance efficiency, safety, and data accuracy in mining operations. We presented Aloft, an artifact which contains a multiple mine environments for drone simulations. Aloft was designed to allow the self-adaptive systems research community to investigate this specific problem domain with a physics-based testbed. We provided a high level description followed by an instantiation using a return-to-home exemplar. An example run and corresponding results were recorded to illustrate how one can perform experiments with Aloft. Guidance was given to show how users can continue to research self-adaptive systems for drones by expanding Aloft's exemplar.

ACKNOWLEDGMENTS

This work is supported by the ASUMI project and Assuring Autonomy International Programme (http://www.york.ac.uk/assuring-autonomy).

REFERENCES

[1] Mehrnoosh Askarpour, Christos Tsigkanos, Claudio Menghi, Radu Calinescu, Patrizio Pelliccione, Sergio García, Ricardo Caldas, Tim J von Oertzen, Manuel Wimmer, Luca Berardinelli, Matteo Rossi, Marcello M. Bersani, and Gabriel S. Rodrigues. 2021. RoboMAX: Robotic Mission Adaptation eXemplars. In *2021 International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE, Piscataway, USA, 245–251. https://doi.org/10.1109/SEAMS51251.2021.00040

[2] Marko Bjelonic. 2016–2018. YOLO ROS: Real-Time Object Detection for ROS. https://github.com/leggedrobotics/darknet_ros.

[3] Jane Cleland-Huang, Ankit Agrawal, Michael Vierhauser, Michael Murphy, and Mike Prieto. 2022. Extending MAPE-K to support human-machine teaming. In *Proceedings of the 17th Symposium on Software Engineering for Adaptive and Self-Managing Systems*. IEEE, Piscataway, USA, 120–131.

[4] Jane Cleland-Huang, Theodore Chambers, Sebastian Zudaire, Muhammed Tawfiq Chowdhury, Ankit Agrawal, and Michael Vierhauser. 2023. Human-Machine Teaming with Small Unmanned Aerial Systems in a MAPE-K Environment. *ACM Trans. Auton. Adapt. Syst.* (2023). https://doi.org/10.1145/3618001

[5] Roman Fedorenko, Aidar Gabdullin, and Anna Fedorenko. 2018. Global UGV path planning on point cloud maps created by UAV. In *2018 3rd IEEE International Conference on Intelligent Transportation Engineering (ICITE)*. IEEE, 253–258.

[6] Simos Gerasimou, Radu Calinescu, and Giordano Tamburrelli. 2018. Synthesis of probabilistic models for quality-of-service software engineering. *Automated Software Engineering* 25 (2018), 785–831.

[7] Victoria J Hodge, Richard Hawkins, and Rob Alexander. 2021. Deep reinforcement learning for drone navigation using sensor data. *Neural Computing and Applications* 33 (2021), 2015–2033.

[8] J.O. Kephart and D.M. Chess. 2003. The vision of autonomic computing. *Computer* 36, 1 (2003), 41–50. https://doi.org/10.1109/MC.2003.1160055

[9] PX4 Vision Kit. 2023. https://docs.px4.io/main/en/complete_vehicles/px4_vision_kit.html.

[10] Paulo Henrique Maia, Lucas Vieira, Matheus Chagas, Yijun Yu, Andrea Zisman, and Bashar Nuseibeh. 2019. Dragonfly: A Tool for Simulating Self-Adaptive Drone Behaviours. In *Proceedings of the 14th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*. IEEE, Piscataway, USA, 107–113. https://doi.org/10.1109/SEAMS.2019.00022

[11] Eduardo F Morales, Rafael Murrieta-Cid, Israel Becerra, and Marco A Esquivel-Basaldua. 2021. A survey on deep learning and deep reinforcement learning in robotics with a tutorial on deep reinforcement learning. *Intelligent Service Robotics* 14, 5 (2021), 773–805.

[12] Gabriel Moreno, Cody Kinneer, Ashutosh Pandey, and David Garlan. 2019. DART-Sim: An exemplar for evaluation and comparison of self-adaptation approaches for smart cyber-physical systems. In *2019 IEEE/ACM 14th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE, Piscataway, USA, 181–187.

[13] Sebeom Park and Yosoon Choi. 2020. Applications of Unmanned Aerial Vehicles in Mining from Exploration to Reclamation: A Review. *Minerals* 10, 8 (2020). https://doi.org/10.3390/min10080663

[14] Salil Purandare, Urjoshi Sinha, Md Nafee Al Islam, Jane Cleland-Huang, and Myra B. Cohen. 2023. Self-Adaptive Mechanisms for Misconfigurations in Small Uncrewed Aerial Systems. In *2023 IEEE/ACM 18th Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE, Piscataway, USA, 169–180. https://doi.org/10.1109/SEAMS59076.2023.00030

[15] Morgan Quigley, Brian Gerkey, Ken Conley, Josh Faust, Tully Foote, Jeremy Leibs, Eric Berger, Rob Wheeler, and Andrew Ng. 2009. ROS: an open-source Robot Operating System. In *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA) Workshop on Open Source Robotics*. IEEE, Piscataway, USA.

[16] Tomáš Rouček, Martin Pecka, Petr Čížek, Tomáš Petříček, Jan Bayer, Vojtěch Šalanský, Daniel Heřt, Matěj Petrlík, Tomáš Báča, Vojěch Spurný, François Pomerleau, Vladimír Kubelka, Jan Faigl, Karel Zimmermann, Martin Saska, Tomáš Svoboda, and Tomáš Krajník. 2020. DARPA Subterranean Challenge: Multi-robotic Exploration of Underground Environments. In *Modelling and Simulation for Autonomous Systems*, Jan Mazal, Adriano Fagiolini, and Petr Vasik (Eds.). Springer International Publishing, Cham, 274–290.

[17] Javad Shahmoradi, Pedram Roghanchi, and Mostafa Hassanalian. 2020. Drones in underground mines: challenges and applications. In *2020 Gulf Southwest Section Conference*. ASEE.

[18] G. Silva, J. Pasler, J. Zwanepol, E. Alberts, S. Tarifa, I. Gerostathopoulos, E. Johnsen, and C. Corbato. 2023. SUAVE: An Exemplar for Self-Adaptive Underwater Vehicles. In *2023 IEEE/ACM 18th Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE, Piscataway, USA, 181–187. https://doi.org/10.1109/SEAMS59076.2023.00031

[19] PX4-Autopilot: Gazebo Classic Simulation. 2023. https://docs.px4.io/main/en/simulation/.

[20] Satoshi Suzuki. 2018. Integrated navigation for autonomous drone in GPS and GPS-denied environments. *Journal of Robotics and Mechatronics* 30, 3 (2018), 373–379.

[21] PX4 Developement team. [n. d.]. https://github.com/PX4/PX4-Avoidance.

[22] Iwan Ulrich and Johann Borenstein. 2000. VFH/sup */: local obstacle avoidance with look-ahead verification. *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)* 3 (2000), 2505–2511 vol.3. https://api.semanticscholar.org/CorpusID:5615398

[23] Sonia Waharte and Niki Trigoni. 2010. Supporting Search and Rescue Operations with UAVs. In *2010 International Conference on Emerging Security Technologies*. IEEE, Piscataway, USA, 142–147. https://doi.org/10.1109/EST.2010.31