A Discrete Event Simulation and Evaluation Framework for Multi UAV System Maintenance Processes

Thomas Dietrich*, Silvia Krug† and Armin Zimmermann*
*Systems and Software Engineering Group, Technische Universität Ilmenau, Germany
{thomas.dietrich, armin.zimmermann}@tu-ilmenau.de
†Communication Networks Group, Technische Universität Ilmenau, Germany
silvia.krug@tu-ilmenau.de

***Abstract*—Unmanned Aerial Vehicles (UAVs) are used in all kinds of applications. The utilization of multiple UAVs acting in a cooperative manner within the same operation field is especially interesting. However, many deployment scenarios pose the challenge of continuous applications. UAVs are equipped with a limited battery which enables short term applications. Replacement and recharging maintenance processes are needed to achieve uninterrupted mission execution. This paper presents a novel simulation framework for mobile robotic systems with the focus on energy maintenance and node replacement strategies.**

***Index Terms*—Multi UAV Systems; Multicopter Systems; Consumption Prediction; Energy Maintenance; Replacement Processes; Discrete Event Simulation; OMNeT++.**

I. Introduction and Related Work

UAVs are small versatile airborne robotic systems and enjoy high popularity in various application fields. One UAV has the size and power to carry and use sensors, cameras or actuators to fulfill a wide variety of tasks. Acquirable sensor data includes images and video, environmental measurements, and other types of data collected from the exploration area. Thereby, UAVs are capable of assisting in reconnaissance, locating objects or creating a comprehensive communication network in areas without or with damaged infrastructure [1].

All mentioned use cases benefit from multi UAV deployment. Multiple UAVs acting independently or in a cooperative manner within the same operation field can help to fulfill missions quicker and/or more efficiently and enable new possibilities.

However, deployment scenarios as mentioned above pose a special challenge to UAVs and mobile robots in general. While a communication network provision or a reconnaissance mission might be scheduled for hours or days, the vast majority of UAVs are equipped with rechargeable batteries as their source of energy, allowing flight times of 10 to 60 minutes, depending on the hardware configuration and the mission parameters.

Prior to depletion, the battery needs to be replaced or recharged before further tasks can be executed. The UAV has to abandon its current task and leave its location for this maintenance process. In case of multi UAV deployments, other UAVs could take over the task of the UAV in question.

The management of such processes should be automated to let system operators and designers concentrate on the actual mission instead of allocating UAVs to tasks or deciding which UAV should be replaced at what time. Several necessary technical solutions for autonomous charging have been proposed recently, e.g., in [2], [3] and [4]. Another prerequisite is a communication protocol supporting UAV swarms and cooperative missions considering maintenance processes, which has been developed in a previous work [5].

The system integration and management level of multi UAV systems has not yet received equal consideration in the literature. For instance, replacement and recharging strategies are needed to achieve uninterrupted mission execution. Such strategies should schedule UAVs in a way that optimizes non-functional system parameters such as utilization, energy use, and mission reliability. Development and evaluation of such strategies have to be done based on models and simulations before their real world validation to save time and costs.

Existing UAV simulation frameworks such as ROS/Gazebo, Microsoft AirSim or Drone Flight Simulator primarily focus on the physical behavior, movement and control of single UAVs.

This paper presents a novel simulation framework for mobile robotic systems with the focus on energy maintenance and node replacement strategies involving multiple UAVs. The goal of the simulation framework is to evaluate and later optimize mission scheduling strategies and management processes.

The remainder of the paper is organized as follows: First, we will analyze the requirements of a multi UAV system simulation framework in section II and define a reference scenario to test against in section II-A. Following the theoretical discussion, the implementation of the simulation framework based on OMNeT++ will be presented in section III. To proof and evaluate the fitness of the framework to provide the means to model and simulate the required maintenance functionalities, section III-C presents the implementation of the reference scenario. Finally, conclusions are given.

II. Requirements Analysis

The goal of this work is a simulation framework for multi UAV systems, with special attention to coordinated and cooperative maintenance processes including recharging and individual node replacements. This section will address the proposed UAV maintenance simulation framework from two perspectives: In order to cover a wide range of UAV utilization scenarios including maintenance processes, a general standpoint will be discussed. On the other side, a specific scenario will be taken into account.

A. Centralized Multicopter Control (CMC) Scenario

Even though the framework should be capable of simulating a wide range of multi UAV scenarios, we focus on a cooperative multicopter scenario as an example. In our scenario, several multicopter UAVs are utilized to support a search and rescue mission. All UAVs are controlled by one central mission control entity, which assigns missions to individual UAVs. Maintenance actions are solely decided by the mission control. Backup or redundant UAVs are necessary to cover the overall mission when realistic energy consumption is considered.

In this scenario the replacement decision is done based on the full utilization of the battery. The UAV node is exchanged at the last possible moment, so that the UAV can still reach a nearby charging station with almost no energy left. The replacing UAV is a fully charged UAV located at a charging station. The number of UAVs is not limited in this example and should thus exceed the number of required active UAV nodes.

B. System Components and Functionalities

The following components and processes should be supported by the anticipated framework.

a) UAVs and Multicopters: Several UAVs need to be supported as generic airborne mobile vehicles with basic movement and behavior properties. Actual simulation scenarios will utilize specific UAV models with specific functionality and restrictions. Currently, we focus on multicopter UAVs as an example. Multicopters are aircrafts with multiple symmetrically arranged motors and propellers oriented in the upward position. The multicopter UAV is fairly unrestricted in its movement, as it can move in all directions as well as hold a certain position.

b) Mission Control: The mission control is a central entity with a combined view on multiple UAVs. Depending on the simulation scenario, multiple mission controls could be in charge of a subset of UAVs, or UAVs could be self-organized without the existence of a mission control. A mission control entity is responsible for the overall mission goal and has knowledge of the current state of all controlled UAVs. Missions assigned to the individual UAVs are either predefined or generated by the mission control during the execution. If not otherwise specified, missions will be repeated indefinitely.

c) Mission Execution: The execution of missions happens inside the UAVs, autonomously from other systems. A mission consists of individual maneuvers defined by commands. Looking specifically at multicopters, all relevant commands can either be represented by hovering or point-to-point movement maneuvers. Hovering means that the UAV will stay at a fixed position for a given time, while a point-to-point maneuver describes a movement from point A to point B on a straight line. This simplified representation of each multicopter for the simulation framework is exact enough for our high-level multi UAV control view because of the direction-symmetrical nature of multicopters and their movement capabilities. Plane-like UAVs are also possible but require a different representation of their behavior.

d) Communication: Control and data communication between all simulation nodes is a basic requirement for the simulation framework. For a first iteration, simple communication links with perfect reception and unlimited range can be applied but more complex signal propagation models and communication protocols should be applicable at a later point.

e) UAV Energy Storage: Similar to most mobile robots, UAVs are usually equipped with rather small batteries as energy storage because of space and weight limitations. A requirement of the simulation framework is thus the implementation and application of energy storage and energy consumption models. Such models can be found in the literature, either based on a mathematical analysis of the UAV's physics [6], [7] or based on statistical analysis of measured data [8]. The energy consumption of the multicopter UAV in the proposed simulation framework will follow the energy consumption profile definition for a real world multicopter model introduced in [8] .

f) Maintenance Processes: In order to achieve the overall goal of a multi UAV system, an uninterrupted execution of all missions has to be ensured. Possible causes for mission interruptions are communication errors, hardware failures and energy depletion. For now, communication errors are not considered, but the communication layer should be prepared to handle erroneous states, e.g., by following a best-guess approach or by local cooperation with other UAVs. Hardware failures in the form of unexpected UAV losses should be a supported optional feature of the simulation framework. Depletion of the size-limited UAV battery is the most probable reason for mission failure.

g) Replacement and Recharging: The mentioned problem of mission interruption due to battery depletion needs to be overcome with a replacement and recharging strategy. A UAV reaching a certain threshold of battery depletion (and/or depending on the next planned tasks) will be replaced by another UAV and recharged for later use. The replacement process is critical for the uninterrupted higher-level mission execution. Different strategies regarding when, where and with whom the exchange occurs are possible and should be supported by the simulation framework.

h) Charging Station: The logical last step in the energy maintenance process is the recharging of the depleted UAV

Fig. 1. Example of the Simulation Framework Visualization; Snapshot from the Reference Implementation Execution

battery, which takes considerable time. For the abstraction level of the simulation framework no functional requirements need to be defined for the charging station. A charging station can be abstracted as a non-interactive element with a fixed position and minimal energy characteristics. UAVs are only allowed to start a recharging process in close proximity to a charging station.

C. Evaluation

The introduced simulation framework should allow to define mission scenarios and simulate UAV movement and UAV behavior based on implemented behavior models and maintenance strategies. The outcome of these simulation runs will be numerical performance results, characterizing the costs, efficiency and reliability of the simulation setup.

These results can be compared for different combinations and parameter values to define maintenance strategy recommendations. However, as each simulation run will be a stochastic experiment depending on uncertainties captured in the model, many evaluations with different random sequences will have to be carried out and the samples need to be treated with statistical rigor. Further computations and comparison are thus best done in independent numerical tools like R. The simulation framework should therefore be able to exchange data, i.e., write status data into a parsable log file format or read varying parameter settings.

D. Visualization

Presentation of the behavior and model debugging require a graphical visualization of the mission and the maintenance processes. A visualization will also be beneficial during the development and validation of new features of the simulation framework itself. It should be possible to accelerate the visualization of the node movement w.r.t. real time to comprehend and verify long-term behavior rapidly.

III. UAV Maintenance Simulation Framework

The presented framework focuses on three major tasks: One task is the simulation of UAV movement and mission execution, another is the simulation and evaluation of UAV maintenance processes. The third task is the simulation of network communication processes between UAVs and other simulated entities. As discussed in the requirements part, the communication is currently only represented by a simplified implementation and will grow as soon as new requirements arise.

A. Discrete Event Simulation

For our purpose of maintenance strategy development and evaluation, the communication and coordination between UAVs can be reduced to simple timed interactions. The movement and behavior of UAVs is highly dependent on time-continuous processes, which could be represented by sets of differential equations at a detailed physics level. However, for the simulation environment only the resulting state of the UAV after an executed maneuver is of interest. The continuous processes can thus be reduced to timed events with summarized impacts and new UAV state results. The described models and processes justify a pure discrete event simulation, which simplifies the program structure and greatly speeds up the computations.

B. Framework Structure

Based on the defined requirements the following implementation decisions were made.

1) Simulation environment OMNET++: Instead of programming a framework from scratch, we searched for an existing environment that could be extended and adapted for our needs. The simulation environment OMNeT++, a discrete event simulation library and framework, was found to fit our needs [9], [10]. It is highly extensible and provides a wide variety of components. Available add-ons provide further functionality and predefined components for various research fields. OMNeT++ is especially oriented at network simulations, but also provides components and libraries for other aspects of a communication network.

The support of ad-hoc networks, the easy representation of network node movement and the overall high customizability of simulation model aspects made OMNeT++ an excellent base for the presented simulation framework.

Version 5.0 of OMNeT++ introduced a new visualization component. While so far visualization of communication node positions was done on a simple 2D plane, the introduction of OpenSceneGraph into OMNET++ allowed the visualization of communication nodes in a three-dimensional representation. An example of the visualization results can be seen in Figure 1. The OpenSceneGraph add-on with the osgEarth library is furthermore capable to present real world map data and projected three dimensional building models. Even though these features are not essential for the generation of end results, they help in the development and testing phase, and support better visualization for UAV system operators. At the time of this writing, the utilized software versions were OMNeT++ 5.1, OpenSceneGraph 3.5.3 and osgEarth 2.7.0.

2) Mission Control: This component represents the entity to manage the UAVs in centralized scenarios. It is capable to perform maintenance related tasks and reorganize nodes and redistribute missions.

Missions to be executed by UAVs as part of the overall scenario are loaded from files of the `waypoints` file type. These files contain lines of comma-separated maneuver commands and command parameters. The specific structure of the files is generated by the software Mission Planner [11] inside its Flight Planning pane. The software was already used during real world flight scenario measurements documented in [8]. Mission Planner provides all features to easily define and edit missions for UAVs and also transfer these to the real world UAV. By using the same file format, not only can the mission definition process be simplified, the same mission can also be executed both in the real world and the simulation environment without translation efforts.

3) Commands: The Mission Planner software communicates with compatible UAVs using the MAVLink communication library [12]. MAVLink defines a very lightweight, header-only message set and structure and is widely incorporated in modern flight control firmwares, management software and other ground to MAV/UAV communication links.

Table I lists all MAVLink movement commands considered and supported by the simulation framework as part of a `waypoints` file.

Additionally, in accordance with [5], the `CHARGE` command was implemented. The command was especially defined for maintenance considerations and instructs the UAV to stay in position (e.g., connected to a charging station) as long as the battery is recharging. The command terminates when the full charge state or a defined intermediate percentage has been reached.

4) Node Hierarchy: The simulation framework supports different kinds of nodes, from stationary devices to airborne vehicles. Nodes as shown in Figure 2 create a hierarchy for a wide range of robotic systems. The abstract class `GenericNode` defines the common base for all nodes, consisting of properties like the geographic coordinates or a current operational state representation. Further base classes `StationaryNode` and `MobileNode` are derived from the `GenericNode` with additional properties. The `ChargingStation` is implemented as a direct specialization of the `GenericNode` because of its special properties in energy maintenance processes.

TABLE I
SUPPORTED MAVLINK MOVEMENT COMMANDS

MAVLink	Description
WAYPOINT	Position change to specific coordinates
LOITER_TIME	Hold position for a certain duration
LOITER_UNLIM	Hold position indefinite
LAND	Land at the current coordinates
RETURN_TO_LAUNCH	Move and land at the launch coordinates
TAKEOFF	Ascent to a certain height at the current position

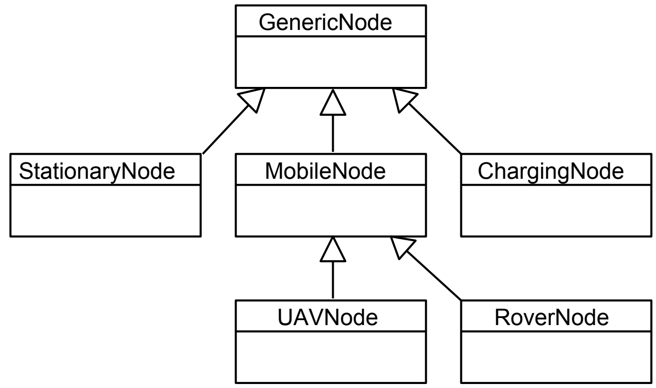

Fig. 2. Class Diagram of the Provided Node Hierarchy Structure

The shown hierarchy represents the currently implemented set of node types. Further specialized classes can be derived by the framework user. All methods needed for the intended simulation framework execution are either provided by `GenericNode` or `StationaryNode`/`MobileNode`, or are abstract methods to be implemented by full classes.

Inheriting from the `MobileNode`, the `UAVNode` is a specialized example and is the main node to be used in further analyses. The UAV node has concrete implementations for the behavioral functions and contains and utilizes an energy consumption profile as described in more detail below. In order to work with the commands as stated above, a number of command execution engines are implemented specifically for the UAV.

5) Command Execution Engine: The behavior of a node is highly dependent on the current operation state of the device. The state of the node is linked to to command that the node is currently executing. These states are represented for each UAV by the supported MAVLink commands shown in Table I. Each command has a specific impact on the node in terms of functionality, restrictions and behavior.

To cope with this tight dependency, the behavioral software design pattern "state pattern" was applied. Figure 3 shows the design and relations of the implemented structure following the commands definition from above.

The UAV node implements all methods needed to simulate the behavior of the UAV. Command-specific details of these methods are however outsourced to various command-specific subclasses of the abstract `CommandExecEngine` class. A UAV owns exactly one command execution engine object as a member and passes down all state/command-specific compu-

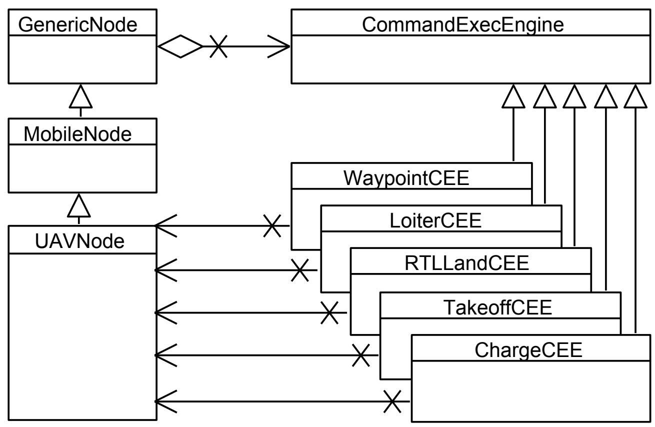

Fig. 3. The Implemented Structure for Command Execution Engines

tations. The command execution engine object is exchanged upon command completion.

It should be noted that the class `CommandExecEngine` is associated with the `GenericNode`, but all shown full implementations are associated with the specialized `UAVNode` class. This ensures that all derived nodes can utilize the concept of commands and command execution engines, while at the same time specific nodes will be associated with matching execution engines, able to interact with the associated specialized node class. To allow direct access from command execution engines to all members of the respective node class, the command execution engines are defined as C++ friend classes of the node class.

The concept allows for a wide range of different node and command types, the behavior and functionality of a node is not restricted by the simulation framework.

6) Communication and Interaction: So far the structure of a typical simulation setup is defined by a mission control entity and by multiple UAV or charging nodes in the simulation field. The simulation framework implements and provides simple communication and coordination means between them. The behavior of nodes relies on the existence of a mission, represented as a list of commands to be executed. The mission control will therefore distribute missions among the existing nodes for later execution. Nodes without a mission will stay in an inactive state.

Active nodes will start executing the assigned commands autonomously and will report status data back to the mission control regularly (if available in the scenario), e.g., upon command completion. As mentioned in the last section, the command to be executed decides on the execution engine used by the node. The node will be able to update its data (e.g., its position) independently of the active execution engine, and report status information (e.g., the estimated time until command completion).

The communication between mission control and field nodes is implemented using the communication features provided by OMNeT++. These already feature input/output queues and propagation delays as well as radio distance restrictions.

7) Energy Consumption Prediction: The functionality to model and simulate UAV maintenance processes is the main goal of the presented simulation framework. The limited energy capacity and the energy consumption of nodes are implemented in multiple parts of the presented structure.

Figure 4 depicts a simplified view of the involved classes and methods. Energy consumption is a common functionality for all nodes and hence part of all command execution engines. A limited energy source, in the form of a battery, is on the other hand only relevant to a part of nodes not connected to a constant power supply. The battery class is thus modeled as a member of `MobileNode`.

Dietrich et al. present an empirical study in [8] to create and parametrize a consumption profile for multicopter UAVs. The profile was implemented as part of the specialized UAV node and is represented by the proxy function `getCurrent()`. Both the battery object and the energy profile are subsequently used to simulate the energy consumption during a specific maneuver as well as to predict the consumption of a future maneuver. The latter is of high importance for the maintenance management as it will provide the data needed to plan processes ahead of time.

8) Maintenance and Replacement Processes: The specification of maintenance processes can vary between different scenarios, from centralized solutions involving a single mission control to self-organized multi UAV systems with cooperative maintenance and node replacement decisions. The current implementation state of the simulation framework components provides the flexibility and extendability to build simulations of different kinds of scenarios. A key feature for maintenance processes is the aggregation of combined knowledge about the status of multiple UAVs in the field. As discussed in the previous paragraphs, information like the remaining battery capacity and the energy consumption for future maneuvers (and hence the gross remaining time of operation) is know by the individual UAVs. This data can be transmitted to other UAVs or a mission control to aid in the maintenance process decision making. Methods and

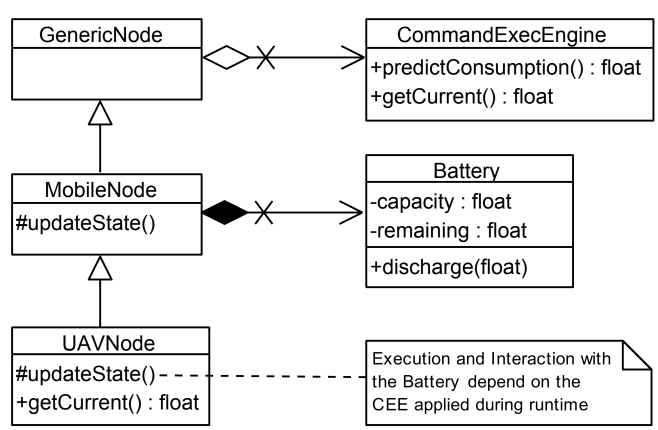

Fig. 4. Energy Profile and Energy Prediction, Simplified Class Diagram

data members to store, retrieve and transmit this data are implemented by the simulation framework in all nodes.

The mission control simulation component provides an internal representation of all managed UAVs in the form of a lookup table. The table consists of a snapshot view of all UAVs, containing the current state, the currently executed mission and future maintenance needs.

The described implemented features will be used and combined with other variable aspects of the simulation framework to build scenarios. Additional effort is needed to specialize, connect and parameterize these components.

C. Reference Scenario Implementation

The Centralized Multicopter Control (CMC) Scenario can be realized by the current simulation framework implementation state. The mentioned components are used to implement the scenario on top of the framework.

Our simulation setup consists of one mission control, two charging stations and 50 multicopter UAVs and is confined to an area of $400 \times 800\mathrm{m}$. The charging stations are positioned at opposite sides of the operation field and the UAVs are equally located at both. The mission control manages all UAVs over an uninterrupted communication link and is in charge of the mission assignment and of all maintenance decisions. The overall goal consists of five missions: three waypoint-based paths inside the operation field and two continuous hovering maneuvers at given positions. All missions are supposed to to repeated/executed indefinitely.

At the beginning of a simulation run, five UAVs are selected and assigned to a mission by the mission control. The rest of the UAVs will stay inactive. A snapshot of the simulation during execution can be seen in Figure 1 on page 3.

In the initialization phase of the scenario all UAVs need to travel to their respective starting points. From there, UAVs transition into the operational phase and follow the mission instructions, i.e., fly maneuvers one after another or hold the current position. During all these flights, the framework will simulate a realistic battery consumption.

The UAV is able to predict the energy needed for the following maneuvers and (based on the position after the maneuvers) can also predict the energy needed to go to the nearest charging station afterwards. By applying simple mathematics and by comparing the sum of consumptions with the known remaining battery capacity of the UAV, we can decide how far the UAV is able to proceed in its mission before having to go back to a charging station. This predicted moment in time is the time of latest replacement and will be calculated and transfered to the mission control regularly during flight and after completion of every command.

The mission control will receive the replacement demand and schedule an inactive UAV to act as the replacement UAV. The new UAV will be sent to the exchange coordinates just in time to meet with the old UAV, which will then transfer the mission instructions to the new UAV. After this process, the old UAV will shift into a maintenance phase and proceed to the charging station, before running out of energy. The new UAV will resume the mission execution in the old UAV's stead. The described flight operation phases and the replacement demands are tracked by the mission control in its lookup table to support all maintenance decisions.

The implemented and described reference scenario is a straightforward approach to satisfy the need for an uninterrupted higher-level mission execution applying maintenance processes provided by the presented simulation framework.

IV. Conclusion and Future Work

We have shown which aspects of a multi UAV system are dependent on maintenance processes and confirmed the need for a simulation framework to evaluate different maintenance strategies. We have analyzed the requirements for such a simulation framework with a generic UAV scenario in mind and defined a reference scenario to test against. The found requirements were the basis for an implementation of a multi UAV maintenance simulation framework presented in the main part of this paper. The showcased reference scenario implementation proves the capability of the simulation framework as defined in the requirements analysis. Improvements in the maintenance strategy and in the maintenance processes, and an evaluation and comparison of performance measures amongst different settings will be the subject of further research into the field of multi UAV system maintenance processes.

References

[1] P. Dasgupta, "A multiagent swarming system for distributed automatic target recognition using unmanned aerial vehicles," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 38, no. 3, pp. 549–563, 2008.

[2] N. Ure, G. Chowdhary, T. Toksoz, J. How, M. Vavrina, and J. Vian, "An Automated Battery Management System to Enable Persistent Missions With Multiple Aerial Vehicles," *Mechatronics, IEEE/ASME Transactions on*, vol. 20, no. 1, pp. 275–286, Feb 2015.

[3] J. Leonard, A. Savvaris, and A. Tsourdos, "Energy management in swarms of unmanned aerial vehicles," in *Unmanned Aircraft Systems (ICUAS), 2013 International Conference on*, May 2013, pp. 124–133.

[4] T. Toksoz, J. Redding, M. Michini, B. Michini, J. P. How, M. A. Vavrina, and J. Vian, "Automated Battery Swap and Recharge to Enable Persistent UAV Missions," *Infotech@Aerospace*, p. 10, 03 2011. [Online]. Available: http://dx.doi.org/10.2514/6.2011-1405

[5] T. Dietrich, O. Andryeyev, and A. Zimmermann, "Towards a Unified Decentralized Swarm Management and Maintenance Coordination Based on MAVLink," in *IEEE Int. Conf. on Autonomous Robot Systems and Competitions (ICARSC 2016)*, 05 2016.

[6] A. Abdilla, A. Richards, and S. Burrow, "Power and endurance modelling of battery-powered rotorcraft," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, Sept 2015, pp. 675–680.

[7] F. Morbidi, R. Cano, and D. Lara, "Minimum-energy path generation for a quadrotor UAV," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 1492–1498.

[8] T. Dietrich, S. Krug, and A. Zimmermann, "An Empirical Study on Generic Multicopter Energy Consumption Profiles," in *2017 Annual IEEE International Systems Conference (SysCon)*, April 2017, pp. 1–6.

[9] OMNeT++ Network Simulation Framework. [Online]. Available: http://www.omnetpp.org/

[10] A. Varga, "Using the OMNeT++ discrete event simulation system in education," *IEEE Transactions on Education*, vol. 42, no. 4, pp. 11 pp.–, Nov 1999.

[11] "Mission Planner Home," (visited on 05-Nov-2016). [Online]. Available: http://ardupilot.org/planner/index.html

[12] "Mavlink micro air vehicle communication protocol," (visited on 12-Feb-2016). [Online]. Available: http://qgroundcontrol.org/mavlink/start