A Generic Unifying Ontology for Civil Unmanned Aerial Vehicle Missions

Benjamin Schumann*, Jim Scanlan† and Hans Fangohr‡
University of Southampton, Southampton, Hampshire, SO17 1BJ, UK

The aerospace industry struggles to account appropriately for the operational environment of a product during the early design phase. This can lead to suboptimal designs that can compromise the commercial success of a product. An agent-based operational simulation is shown to reduce the knowledge gap and increase understanding of aerospace products interacting with their operational environments early on. The simulation aims to be a generic tool to model any mission scenario for Unmanned Aerial Vehicles. This paper suggests a unifying ontology aiming to find a small set of parameters that map to almost any conceivable mission. This is achieved by combining Geographical Information Systems (GIS) maps with an agent-based framework. Database structure and integration into the operational simulation is demonstrated by introducing a generic mission case study.

Nomenclature

CAD	Computer-Aided Design
CFD	Computational Fluid Dynamics
GIS	Geographical Information Systems
SAR	Search and Rescue
UAV	Unmanned Aerial Vehicle & System

I. Introduction

Aerospace product complexity has grown manifold during the past decades due to growing computational power.[1] However, organizational and managerial processes did not develop at the same speed, effectively hampering efficient use of available resources.[2] Companies still rely on decision processes guided by chief engineers that have to negotiate consensus-driven results between specialist teams.[3] The quality of design decision is strongly influenced by communication skills and information clarity. It is yet rare to find truly integrated software tool-chains.

However, even under ideal information exchange, design trade-offs are often focused on short-term success, neglecting long-term costs, environmental impact and life-cycle considerations.[4] One reason for this is a lack of rigorous operational analysis of the product in question. Whereas engineers are used to analyzing design changes with regards to performance changes, they struggle in linking design changes to operational outcomes. In other words, engineers can easily answer how much faster an aircraft will fly by changing the wing span but they struggle to even approach answering how much money they would save or loose over the lifetime of the aircraft by implementing this design change.

This lack is most acute in the early (or conceptual) design phase. Here, the most fundamental and most critical design decisions are agreed in a short space of time, largely relying on engineering judgment and expert experience.[5] Erroneous decisions can make the difference between a highly successful product and a commercial failure. The early design phase is characterized by budget, information and time constraints that do not allow exploring the design space fully.[6] Information about the intended market and operational

*PhD-Researcher, Computational Engineering and Design, University of Southampton, UK
†Professor of Aerospace Design, Computational Engineering and Design, University of Southampton, UK
‡Professor of Computational Modelling, Computational Engineering and Design, University of Southampton, UK

environment of the product are not yet available or not used to the full extend. Therefore, a link between design ideas and operational constraints is not clearly identified. Design problems relating to the operational environment are only detected at a later stage of design where the principal design is already decided upon.[7] This can lead to delays, cost overruns and inferior quality.

An operational simulation can bridge the knowledge gap between the design and its intended environment early on. By using available early (possibly stochastic) design information such as CAD and CFD results, performance characteristics can be used to simulate the design within its intended operational environment. At this stage, designers know the kind and number of missions the product will conduct during its intended life cycle. Such an operational simulation returns metrics about operational costs, crashes or maintenance which can, in turn, be used to compare and rank designs using a value function (see Collopy[8] for an introduction into using value functions for aerospace design). This allows to choose the best design for the operational requirements but also helps identifying optimal operational procedures which may differ from customer expectations. Gorissen et al.[3] describe a case study designing operationally-optimized UAVs using this "Value-driven design" approach. This includes using the operational simulation described here.

Despite the improvements that an operational simulation can deliver, there are high uncertainties inherent in the design process. Today, large aerospace products are designed over several years, to be used by customers for several decades.[9] Modelling these time frames is very challenging due too growing uncertainties, requiring expert knowledge of the operational environment, experienced modelers and trustworthy data. Therefore, in order to demonstrate the usefulness of operational simulations, it is advantageous to concentrate on aerospace products with much smaller time frames. During recent years, civil Unmanned Aerial Vehicles (UAVs) are becoming more and more important for a variety of "dull, dirty and dangerous" missions.[10] Civil UAV systems are typically developed within months and their life cycle rarely lasts for more than a few years. Therefore, studying the benefits of operational simulations using civil UAVs can lead to insights for studying larger aerospace structures.

This research aims to develop a generic operational simulation that is capable of recreating any type of civil UAV mission easily. This allows engineers to develop products for a wide variety of use cases. Moreover, engineers can compare a design idea within different operational scenarios to not only optimize the design but also the operations of the product. Lastly, a generic tool allows easy extension for larger aerospace structures such as airliners.

One critical element to achieve a generic simulation is the use of GIS-based maps to define UAV mission scenarios. This allows users to define missions to a high degree of realism, taking into account critical geographical features such as airways, coast lines or roads. Moreover, it saves time to the user as they can create missions much faster, drawing on existing maps instead of manually creating agent environments. GIS maps also enable the user to define missions to any desired level of detail depending on existing operational information. Finally, the model can be extended easily with more realistic modules such as weather or real flight schedules.

This paper introduces an ontology for generic operational simulations for civil UAV missions. Section II details the theory of this ontology, including the categorization of missions, their key concepts and the definition of parameters. The theory is the result of practically developing operational simulation mission scenarios for designing UAVs (see Schumann et al.[11] and Schumann et al.[12] for detailed descriptions of the simulation). Section III shows how to apply the theory by introducing and explaining a sample mission. It includes details about the operational simulation developed in this research such as software selection and database implementation.

II. Theory

This Section describes the theoretical framework for civil UAV missions. Subsection A introduces a generic approach of assigning any UAV mission to one of four categories based on their mission goal. Subsection B defines key building block concepts required to build any mission. Subsection C shows 11 characteristics required to define a Mission. Subsection D goes on to demonstrate how eight characteristics are sufficient to define any stage of a UAV mission. The definitions are applied in Section III.

A. Civil UAV Mission Goals

UAVs are well-suited to conduct "dull, dirty or dangerous"[10] missions currently conducted by manned aircrafts or not conducted at all due to high risk or high cost. Missions can be grouped by their purpose as in Figure 1.

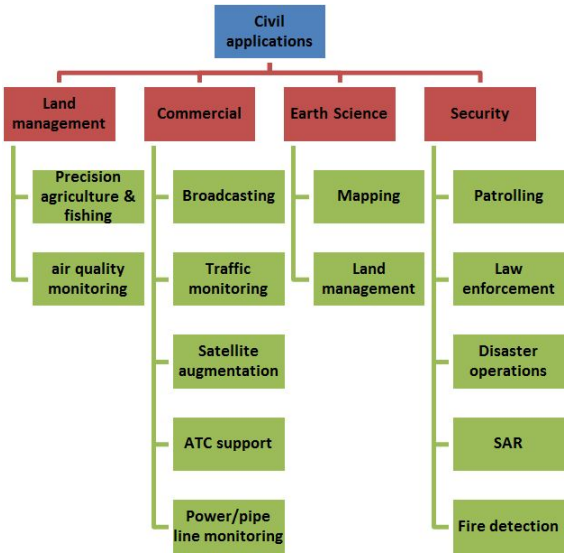

Figure 1. UAV civil missions. Adapted from Cox.[10]

This list is not exhaustive but covers the majority of likely application areas. In order to create a unifying ontology for this wide range of applications, a first step is to group missions not by their purpose but by their main task. Any conceivable civil UAV mission aims to achieve one or more of only four goals: To stay over a fixed point, to follow a certain path, to cover a specified area or to search for something or someone. Naturally, complex missions can contain a combination of these goals. A UAV might be required to patrol along a coastline to look out for drowning people (path) until an emergency happens out at sea and the UAV is required to search for a sinking ship (search). Alternatively, a neighborhood patrol UAV (area) may be diverted by police forces to monitor a car crash (point). The four categories are explained in more detail below.

1. Point missions

The UAV is required to fly to and loiter above one or more fixed points for a specified time (see Figure 2). Fixed-wing UAVs will circle points continuously (and its visual instruments realign themselves to keep the point in sight). Rotary-wing UAVs (i.e. unmanned helicopters) are able to hover stationary above points. In order to define a point mission, the following minimum information is necessary: The coordinates of the points, loiter height, loiter speed, arrival time at the points and the duration to stay at specific points.

2. Path-missions

The UAV is asked to follow one or more paths such as roads, pipe lines, ship routes or shore lines at a specified speed. A path can contain any number of straight line elements (compare Figure 3). Path segmentation is used to create realistic flight profiles with varying characteristics. The defining characteristics include the geographical position of each path, the arrival time at the mission start point, the departure time for each path start point, the cruise height for each path and the cruise speed for each path. Moreover, it is necessary to specify a departure action to be performed upon reaching the end of a path: Either the UAV proceeds with the next path of this mission, or it loiters at the current path end point until the start time for the next path or the UAV flies to its home base until the start time for the next path.

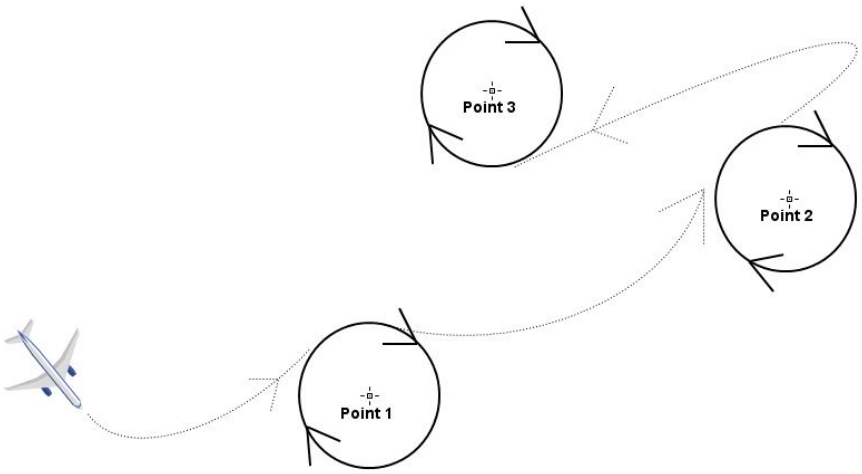

Figure 2. Sample Point mission for fixed-wing UAVs

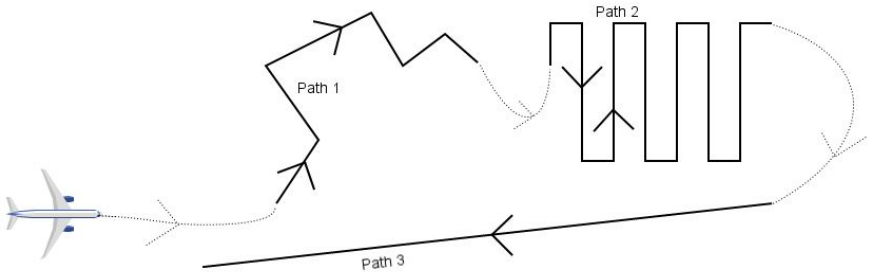

Figure 3. Sample Path mission with three sections

3. Area missions

The UAV is given a specific area that it is required to sweep for purposes such as crop spraying, drug boat detection, air quality monitoring or volcano ash analysis. Technically, area missions can be defined as path missions because sweeping an area is simply following a specific path. It is beyond the scope of this research to create algorithms that automatically create sweep paths for any given area. Instead, specific paths are created to emulate area sweeping (see Path 2 in Figure 3). Area missions require identical information as path missions. From now on, area missions are treated as path missions and are not mentioned separately.

4. Search missions

For this mission type, UAVs are requested to search for objects, humans or events at specific positions. Possible missions include supporting search-and-rescue activities at sea or on land, searching for smugglers or detecting forest fires. Naturally, exact incident location information is unknown at the time of the mission, therefore UAVs are given initial search positions to start searching from (see Figure 4). In order to specify search missions, the time of the search start must be given together with the type of search pattern and some generic pattern parameters such as sweep width. Moreover, the search height must be given along with a hover duration that characterizes how long the UAV should loiter above the incident once it is found. Each search mission can contain any number of incidents. Search missions are different to Path-missions in that the mission outcome is stochastic: the UAV may find the Incident upon crossing it or it may fail and keep searching (until some specified stop condition). Therefore, the designer cannot create the Search path manually as with Path missions: rather, it is required to only specify a pattern and let the UAV agent search autonomously.

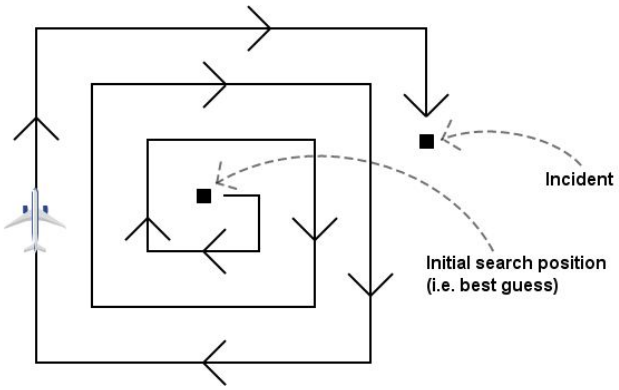

Figure 4. Sample search using an "Expanding-Square"-pattern

B. Ontology

This section will define and describe the ontology to be used for this research. Concepts defined by this ontology will be written with a **C**apital letter for distinction. In general, the life cycle of a UAV is made up of *Missions* that are combinations of *Tracks* which again consist of *Segments*. Each Segment can be either a *Point*, a *Path* or an *Incident* as shown in Figure 5. Any number and combination of Segments define a Track. Any number and combination of Tracks define a Mission. A UAV can conduct any number and combination of Missions during its lifetime.

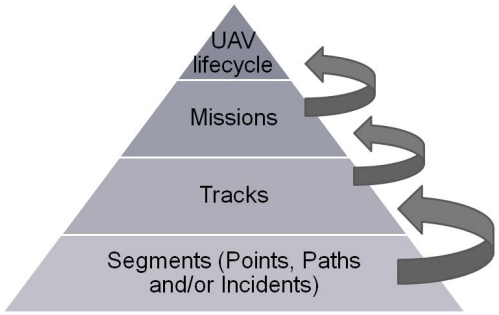

Figure 5. Additive principle for mission generation

The terms are defined as:

Mission Any number and combination of Tracks conducted by a UAV. A Mission is defined as all actions that take place between the start of the first Track and the end of the last Track of this Mission. Sometimes, a UAV runs out of fuel during a Mission and has to return home for refueling. This does not constitute a new Mission. Moreover, it may be required by the Track-definition to land during a Mission. This again does not constitute a new Mission. Depending on the definition of its Tracks, a Mission can contain any number of takeoffs or landings and can last anything between minutes or weeks as defined by user requirements.

Track Any sequence of actions conducted by a UAV between the start- and endpoint of a Track. It consists of any number of Segments. A Track must contain Segments of the same type but it is possible to create mixed Missions by including Tracks of varying categories. The geographical information of Tracks are stored in GIS files whereas the operational information is stored in a database (see sub-section D).

Segment The smallest unit of action for the UAV. A Segment can be either a Point, a Path or an Incident.

Point A one-dimensional geographical position of latitude and longitude. The UAV flies towards that point and conducts specific actions above its position (i.e. loitering) before continuing a Mission.

Path A two-dimensional array of one or more straight lines. The number of lines per Path depends on user requirements. Each Path is known by its geographical position. The UAV follows a Path as close as possible from origin to destination at a specified constant speed.

Incident A one-dimensional geographical position of latitude and longitude (like a Point). Incidents refer to events with unknown position. Incidents must be searched by UAVs following specific patterns starting at an initial search position (i.e. best guess).

C. Generic Mission characteristics

A UAV can fly any number of Missions during its lifetime. A Mission can contain any number and combination of Tracks (see Subsection D). In order to define a Mission, each of its Tracks must be defined using the 11 characteristics defined below. A sample Mission-table can be found in Figure 6.

UAVType A string used to load the correct UAV from a database of UAV configurations. This allows to simulate and analyse various configurations in any combination and judge their operational performance.

UAVID An integer used to distinguish individual UAVs from each other. A new UAV individual is created by assigning a unique combination of "UAVType" and "UAVID". This means that two UAVs can have the same UAVID but be of different type. Care must be taken to ensure that any individual UAV is not assigned physically impossible Missions and Tracks (i.e. be at different locations at the same time...).

Base A string indicating at which airport the UAV should be based before starting the current Track. This is where the UAV will takeoff in order to fly to the first Segment of the given Track.

Track A string indicating the name of the Track that the UAV should follow. Details about Tracks can be found in Subsection D.

Destination A string indicating where the UAV should land after it has finished the current Track. Care must be taken to ensure that subsequent Tracks (or Missions if this is the last Track-entry for the current Mission) of this individual UAV will start at the same location, otherwise the UAV will be "beamed" to its new location.

Time A string in the date-time standard format (*YYYY-MM-DD*T*hh:mm:ss*) indicating when the current Track should be started. More specifically, this entry indicates when the UAV will takeoff from its Base to follow the current Track. All "Time"-entries of Segments in the current Track-table refer to this point in time (see sub-section D).

Repetition An integer value indicating if and at what frequency (in seconds) the current Track should be repeated. If the value is "0", no repetition is required. If the value is smaller than the time it takes to complete the current Track, the UAV will repeat the Track as soon as possible, building up delays for subsequent Tracks. Note that "Base" and "Destination" must be identical in order to have a realistic repetition of Tracks, otherwise UAVs are "beamed" from their "Destination" to the "Base" for repetition.

DashHeight An integer value indicating the height (in meters) of the dash-out segment of this Track. The UAV will fly at this height between the takeoff location ("Base") and the first Segment of the current Track.

DashSpeed An integer value indicating the speed (in meters per second) of the dash-out segment of this Track. The UAV will fly at this speed between the takeoff location ("Base") and the first Segment of the current Track.

ReturnHeight An integer value indicating the height (in meters) of the return segment of this Track. The UAV will fly at this height between the last Segment of the current Track and its final landing "Destination".

ReturnSpeed An integer value indicating the speed (in meters per second) of the return segment of this Track. The UAV will fly at this speed between the last Segment of the current Track and its final landing "Destination".

D. Generic Track characteristics

A Track can contain any number of Segments of the same type (see sample Point Track-table in Figure 7, sample Path Track-table in Figure 9). However, Track characteristics must stay constant independent of the Segment type (Point, Path or Incident). Therefore, it is necessary to define characteristics that can be used with all three types of Segments. These characteristics are defined as:

Origin A string indicating a name of a Point, the starting point for a Path or the name of an Incident. This string can be used to distinguish Points, Paths or Incidents from each other. However, it is not required to name the origin and has no direct operational function.

Destination A string indicating a name of a Point, the destination point for a Path or the name of an Incident. This string can be used to distinguish Points, Paths or Incidents from each other. However, it is not required to name the destination. Note that destination and origin can have the same string if required. This is recommended good practice for Incidents and Points as their origin and destination are equal.

Time An integer value (in seconds) indicating when this Segment should be started *relative* to the start "Time" of the current Mission. Therefore, the first Segments's time-value of a Track is always "0". Time-values can be overridden by the "UponArrival" and "Hover" characteristic. If a Path-Segment is finished faster than the "Time"-value of the subsequent Segment suggests, the UAV will loiter at the destination of the Segment until the subsequent Segment should be started.

UponArrival A string that indicates what should be done upon arriving at this Point or Incident or upon arriving at the destination of this Path-Segment. The keyword "*home*" suggests that the UAV should proceed to its destination airport upon arriving until it is time to proceed to the next Segment of this Track (only applicable if not the last Segment of a Track). This is used to enforce refueling or pausing during a Track. The keyword "*stay*" requires the UAV to loiter at the current Point, Path destination or Incident until it is time to proceed to the next Segment or the subsequent Track. The keyword "*next*" informs the UAV to proceed right to the next Segment disregarding the Time-value of the subsequent entry. This allows to sweep through a Track without interruption. "UponArrival" can override the "Time"-characteristic: if the UAV should go "home" it will not judge if it has enough time to fly to its home airport before starting the subsequent Segment. Therefore, careless data entry may cause the UAV to accumulate significant delays.

Type A string or integer that can be used to cause specific UAV behavior for the current Segment. Depending on the application of this ontology, users can define keywords and UAV actions to make Missions more realistic. As an example, the string "Runner" could indicate to a Path-following fixed-wing UAV, that it is monitoring a marathon at very low speeds. Repeated loitering may be incorporated to account for the slow speed of the runners compared to the indicated flying speed of the UAV. Alternatively, Search missions can use this category to indicate the position uncertainty of this Incident as an integer value in meters.

Loiter An integer value indicating how many seconds the UAV should loiter upon reaching this point, path destination or Incident. A value of 0 indicates that the UAV should proceed with any action defined in the category "UponArrival". "Loiter" overrides "UponArrival" and "Time" entries. Hence, a UAV will loiter even if it should proceed to the next segment indicated by "UponArrival" or if the Time of the subsequent Segment has already passed.

Height An integer value indicated at what height the UAV should fly in meters for the current Segment. If the Segment is a Point, the distance covered to reach the Point and any loitering will be flown at this height (except for the first Point of a Track whose Height is defined in "DashHeight" in the Mission-table for the current Track). If the Segment is a Path, the total length of the Path will be flown at this height as well as possible maneuvers to reach the current Path. If the Segment is a Search, the search and loitering will be flown at this height. A value of "99999" indicates that the UAV should fly at the maximum possible altitude.

Speed An integer value indicating at what speed (in meters per second) the UAV should fly for the current Segment. If the Segment is a Point, the distance towards the Point and any loitering will be flown at

this speed (except for the first Point of a Track whose Speed is defined in "DashSpeed" in the Mission-table for the current Track). If the Segment is a Path, the path itself, any maneuvers conducted to reach the Path and any loitering will be flown at this speed. If the Segment is a Search, the Search and any loitering will be flown at this speed. A value of "9999" indicates that the UAV should fly at its maximum possible speed. A value of "0" indicates that the UAV should fly at its minimum possible speed.

III. Modelling

This section details how the theoretical concepts from Section II will be applied practically. Sub-section A justifies the software selected for this project. The practical database implementation is described in sub-section B, presenting sample tables for all possible cases. Finally, sub-section C will discuss what work remains to apply the theoretical concepts of this paper in practice.

A. Software selection

This research combines three different kinds of software to create the functionality described. Segments are drawn and saved in GIS-map shapefiles using ArcMap©. Data about Missions, Tracks and Segments are stored in SQLiteStudio©. The operational simulation is programmed in AnyLogic©.

1. GIS tool

The industry leader in GIS tools is esri's ArcGis toolbox that allows creating and editing maps easily (www.esri.com). The software was used for this research due to expert support and the large number of existing (and editable) maps. However, future users of this work can use any other GIS program that allows to create so-called "shapefiles", many of which are open source.

2. Database

Segment-,Track- and Mission information is stored in an SQLite database. SQLite is a simple relational database management system that is free and easy to use. Databases were populated using SQLiteStudio (www.sqlitestudio.one.pl).

3. Operational Simulation

The operational simulation was created using AnyLogic (Version 6.7), a software tool developed by XJ Technologies©(www.xjtek.com). AnyLogic offers a true agent-based framework embedded in a classical discrete-event environment. The Java-based software allows multi-core evaluation of design points, enabling exploration of large design spaces and long life-cycles. Moreover, it is possible to create Applets that can be implemented into software tool chains and run in any browser without license restrictions.

B. Database implementation

Two databases are required to model and store operational UAV information. One database stores all Missions whereas the other database stores all Tracks. This structure allows users to build up a library of Missions and Tracks over time that can reduce future life-cycle assignment workload. The following Subsections introduce a sample Mission-table and its corresponding Track-tables to demonstrate the theory described.

1. Missions database

Each table in the Missions-database represents one Mission as defined in Chapter II. UAVs can fly any number and combination of Missions by loading the appropriate Mission-tables using the table names. Therefore, table names should feature suitable names such as "`ForrestFires_UK_2012`". However, no naming convention is prescribed. Figure 6 shows a sample Mission table. This Mission includes three individual UAVs (two of "Type1" and one of "Type2"), two airports and three Tracks ("POINT_Track0" is conducted twice). It can be seen that the first Track "POINT_Track0" is flown by the UAV with index 0 of "Type1" taking off

UAVType	UAV_ID	Base	Track	Destination	Time	Repetition	DashHeight	DashSpeed	ReturnHeight	ReturnSpeed
Type1	0	Airport0	POINT_Track0	Airport0	2012-01-01T08:30:00	0	99999	50	2000	9999
Type2	0	Airport1	PATH_Track1	Airport0	2012-01-03T04:30:00	0	300	9999	300	0
Type1	1	Airport0	SEARCH_Track2	Airport0	2012-01-07T08:30:00	0	500	0	99999	25
Type1	1	Airport0	POINT_Track0	Airport0	2012-01-08T08:30:00	86400	1500	25	400	35

Figure 6. Sample Mission Table

from "Airport0" on January 1^{st} 2012 at 8.30am. After track completion, the UAV returns to "Airport0" and is idle for the rest of this Mission. Meanwhile, the UAV with index 0 of "Type2" departs from "Airport1" on January 3^{rd}, 4.30am to conduct the Track "PATH_Track1", landing afterwards at "Airport0". The last two rows relate to the third UAV in this mission with index 1 of "Type1": after completing its first Track "SEARCH_Track2" on January 7^{th}, it lands at "Airport0" in order to be able to conduct its subsequent Track "POINT_Track0" one day later which also starts from "Airport0". This last Track is repeated daily (i.e. every 86400 seconds) until the end of the simulation runtime. This runtime is defined elsewhere and it is sufficient to mention here that it is possible to prescribe any time value that is larger than the duration of the Missions used. Flight details about dash and return are self-explanatory.

2. Tracks database

Each table in the Tracks-database represents one Track as defined in Chapter II. A Mission can load any number and combination of Tracks by referring to the correct Track-names. Unlike Mission tables, Track tables require a prescribed naming convention. The reason is that the simulation interprets Track-data differently based on the Segment type, i.e. Point, Path or Incident. The convention requires Track-table names to start with the type of Segments used in this Track in capital letters (i.e. "POINT", "PATH", or "SEARCH"), followed by an underscore and a more descriptive name of this Track. The descriptive part can be of any format. The Mission in Figure 6 loads three sample Tracks which are discussed in more detail below:

Track "POINT_Track0": A sample table of this Track can be seen in Figure 7. It consists of three Points that are flown to in turn by the UAV. A possible shapefile is shown in Figure 8.

Origin	Destination	UponArrival	Time	Type	Hover	Height	Speed
Point1	Point1	next	0	Type0	0	200	25
Point2	Point2	home	300	Type1	200	1000	35
Point3	Point3	stay	650	Type0	0	50	9999

Figure 7. Track table for "POINT_Track0"

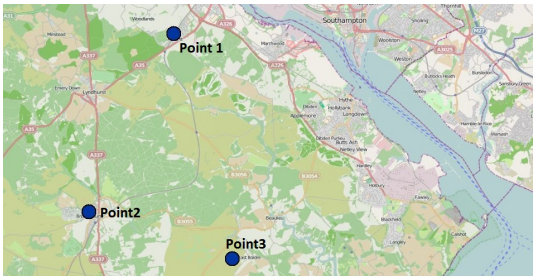

Figure 8. GIS representation

Note that each row represents one Point that is named equally in the "Origin" and "Destination" column for simplicity. The UAV initially proceeds to "Point1" right after taking off from the "Base" indicated in the Mission-table. Upon arrival, it flies towards "Point2", disregarding the "Time"-value of this entry due to the "next" command of "Point1". After loitering for 200 seconds, the UAV flies to its home base, waiting until 650 seconds after the start of this Mission. If this time has passed already, the UAV will take off for "Point3" straight after landing. Refueling is conducted in any case. Note that flying towards "Point3" is conducted at the maximum possible speed, indicated by the Speed-entry 9999.

Track "PATH_Track1": A sample table of this Track can be seen in Figure 9. It consists of three Paths that are followed by the UAV. A possible shapefile is shown in Figure 10.

Initially, the UAV flies towards "Place0" from its "Base" as specified in the Mission-table for this Track. The first Path of this Track lies between "Place0" and "Place1" and is flown at maximum speed and 200

Origin	Destination	UponArrival	Time	Type	Hover	Height	Speed
Place0	Place1	next	0	Type0	0	200	9999
Place1	Place2	next	0	Type0	0	250	9999
Place2	Place3	next	0	Type0	0	300	9999

Figure 9. Track table for "PATH_Track1"

Figure 10. GIS representation

meters height. Upon arrival at "Place1", the UAV proceeds to follow the next Path from "Place1" towards "Place2". Therefore, no special "Time" entry is required for this Path-Segment. The same is true for the last Path-Segment and the UAV will return to the "Destination" of this Mission upon arriving at "Place3". Note that each of the Paths contains multiple straight lines. Each Path can consist of just one or any number of straight lines. The arbitrary separation into Path-Segments allows for varying performance during a flight (i.e. changing flight altitude or speeds) or for enforcing varying behavior using "Hover", "UponArrival" or the additional category "Type". Also note that Segments do not have to be linked together as in Figure 10 but can be geographically separated as in Figure 3. UAVs will cover the inter-Segment distances using flight information specified in the subsequent Segment.

Track "SEARCH_Track2": A sample table of this Track can be seen in Figure 11. It consists of three Incidents that are searched for by the UAV. A possible shapefile is shown in Figure 12.

Origin	Destination	UponArrival	Time	Type	Hover	Height	Speed
Incident0	Incident0	home	0	7200	500	100	25
Incident1	Incident1	home	12000	4000	600	100	25
Incident2	Incident2	home	15000	500	0	100	25

Figure 11. Track table for "SEARCH_Track2"

Figure 12. GIS representation

The first "Incident0" appears at the time of the Mission-start and is searched for at 100 meters "Height" with 25 meters per second "Speed". "Type" indicates the distance in meters between the initial search position and the Incident: it can be drawn randomly somewhere around "Incident0", possibly taking into account the geography of the environment (no searching above land in this example). After spotting "Incident0", the UAV will loiter for another 500 seconds to await rescue services before returning home. After loitering, the UAV is told to go home to refuel as is most realistic in such a scenario. Subsequently, "Incident1" appears much later at 12000 seconds after Mission-start. It will be found faster because the initial search position is much closer. The final "Incident2" has a very low "Type" entry, effectively representing an Incident whose position is known very well, resulting in a very quick search.

C. Current state and future work

The operational simulation used for this research currently has limited functionality with regards to the theoretical framework outlined in this paper. So far, it includes an initial simple flight performance module and is able to simulate Search-missions as described above. It is possible to use external GIS maps and create Missions based on the database. However, Point- and Path-missions are not yet fully possible as their detailed functionality must be programmed. Once completed, the full functionality will be available and work will shift towards easier database and GIS integration, allowing users to setup life-cycle scenarios swiftly.

IV. Conclusion

This paper introduces a unique ontology for civil Unmanned Aerial Vehicle missions. This ontology aims to be a simple yet flexible tool to define a wide range of missions with minimal information input. It has been shown that any mission scenario is a combination of only three possible goals: to loiter above a point, follow a path or search for an incident. Based on this insight, UAV life cycles are broken into missions, tracks and segments that can be combined freely. GIS maps store geographical information about segments whereas a database stores information about track and mission flight profiles. This allows designers to build up a portfolio of maps and mission definitions for reuse. It is shown that any mission can be defined by 11 parameters whereas any track requires 8 parameters.

A sample mission is presented demonstrating how a small set of information can produce a rich variety of UAV missions. The ontology will be used to simulate UAV operations in order to support the early design phase of such products. It is hoped to gain unique insights into the product before critical design decisions are fixed. Moreover, such a simulation shall be used to compare and optimize designs to choose the best design candidate for the detailed design phase.

Acknowledgments

This work was funded by the EPSRC Doctoral Training Center grant no. EP/G03690X/1.

References

[1]Sobieszczanski-Sobieski, J. and Haftka, R. T., "Multidisciplinary Aerospace Design Optimization: Survey of Recent Developments," *Structural Optimization*, Vol. 14, 1997, pp. 1–23.

[2]Keane, A. J. and Nair, P. B., "Problem Solving Environments in Aerospace Design," *Advances in Engineering Software*, Vol. 32, 2001, pp. 477–487.

[3]Gorissen, D., Quaranta, E., Ferraro, M., Schumann, B., van Schaik, J., Keane, A., and Scanlan, J., "A Decision Environment for Complex Design Evaluation," *AIAA Journal*, 2012, Under review.

[4]Price, M., Raghunathan, S., and Curran, R., "An Integrated Systems Engineering Approach to Aircraft Design," *Progress in Aerospace Sciences*, Vol. 42, No. 4, 2006, pp. 331–376.

[5]Raj, P., "Aircraft Design in the 21st Century: Implications for Design Methods," *AIAA Journal*, , No. AIAA 98-2895, 1998.

[6]Quinn, G. F. and Breza, M. J., "Integrating Avionics in the Conceptual Design Phase," *IEEE Aerospace and Electronic Systems Magazine*, Vol. 2, 1987, pp. 2–4.

[7]Will, P. M., "Simulation and Modeling in Early Concept Design: An Industrial Perspective," *Research in Engineering Design*, Vol. 3, 1991, pp. 1–13.

[8]Collopy, P. D. and Hollingsworth, P., "Value-driven Design," *9th AIAA Aviation Technology, Integration and Operations Conference (ATIO)*, No. 2009-7099, AIAA, AIAA, Hilton Head, South Carolina, 21-23 September 2009.

[9]Kirby, M. R., *A Methodology for Technology Identification, Evaluation and Selection in Conceptual and Preliminary Aircraft Design*, Phd-thesis, Georgia Institute of Technology, March 2001.

[10]Cox, T. H., Nagy, C. J., Skoog, M. A., Somers, I. A., and Warner, R., "Civil UAV Capability Assessment," Tech. rep., NASA, December 2004.

[11]Schumann, B., Scanlan, J., and Takeda, K., "A Generic Operational Simulation for Early Design Civil Unmanned Aerial Vehicles," *SIMUL2011: The Third International Conference on Advances in System Simulation*, IARIA, Barcelona, Spain, 2011, ISBN: 978-1-61208-169-4.

[12]Schumann, B., Scanlan, J., and Fangohr, H., "Complex Agent Interactions in Operational SImulations for Aerospace Design," *Proceedings of the 2012 Winter Simulation Conference*, edited by C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A. Uhrmacher, Berlin, Germany, December 2012, Unpublished.