ICAS 2021 SHANGHAI
International Council of the Aeronautical Sciences
32nd Congress of the International Council of the Aeronautical Sciences
September 6-10, 2021
Pudong Shangri-La, Shanghai, China

HYBRID MODELLING FOR FULL MISSION SIMULATION

Petter Krus[1], Robert Braun[2] & Emilia Villani[2]

[1-3]Linköping University, LiU, Sweden
[4]Instituto Technologico Aeronautica, ITA, Brazil

Abstract

Aircraft conceptual design has mostly been characterized by sizing by the use of rather simple analytical models. Such models serve a purpose since they can be easily be manipulated for e.g. trade studies to see the effect of requirements on the design parameters and general performance of the aircraft. However, there is a trend to involve more advanced calculations, e.g. more advanced aerodynamics calculations and flight dynamics analysis, already at the conceptual design, in order to further reduce uncertainty. In the same way, it could be argued that more detailed analysis of behaviour of the aircraft in a mission should be useful.

We propose the simulation of hybrid systems in the framework of transmission line modelling, TLM. We extend an existing continuous time methodology, and adopt UML Activity Diagram and Petri nets for the discrete event dynamics, to describe complex behaviour. This allows for full system simulation of a mission where a more accurate evaluation of mission performance can be obtained. In this paper the methodology is outlined and demonstrated on a UAV mission application.

Keywords: mission simulation, hybrid modelling, flight control, behavioral modelling, activity diagram

1. Introduction

System simulation is increasingly used in aircraft conceptual design. E.g. it is possible to study the effects of subsystems on aircraft level, see e.g. [6] and [8]. In [3] simulation is used to simulate take off and landing, arguably the most critical parts of the flight mission, is included in the design loop for conceptual design. There is also a scope to use mission, or even scenario simulation, for aircraft conceptual design. On one hand there is a trend towards higher fidelity in the physics models, but there should also be a trend towards a higher fidelity in the scenario and behaviour modelling (mission modelling) as a means to have a higher fidelity, and to be able balance system requirements.

Also with a wide enough system boundary most time continuous systems also have discrete event parts. Therefore, the capability to simulate hybrid systems is becoming important also in general.

From a design perspective it can also be useful to start with a discrete event system to clarify the functions, the system can the be gradually elaborated into the physical and time continuous domain. For an aircraft it is useful to start the design with a mission concept [10]. From this the requirement on the aircraft system can then be derived. Using simulation, the feasibility of a design can then be tested in a mission simulation, in the conceptual phase.

For this purpose, both the causality rules of the mission specification and the aircraft aerodynamics, flight dynamics and subsystem behaviour, must be integrated in a single model, requiring a mix of continuous time and discrete event simulation. Existing software solutions with this capability normally lack efficiency when it comes to e.g. simulation based optimization where the system needs to be evaluated a large number of times.

The approach used here is to model discrete event systems in the framework of transmission line modelling, TLM, [1], [4], [7]. This is a methodology that has proven itself for simulation of complex continuous time systems. It allows for pre-compiled component models that can be assembled in run-time. Furthermore, it is inherently parallel and can thus be used to run efficiently on multi-core processors.

2. Mission simulation

Fig. 1 shows the elements needed for mission simulation. There is the aircraft model with its control system. In addition there is the mission control model that can be seen as a high level control system giving commands to the flight control system. There is also a level above the mission of the aircraft, which is indicated in the figure by the external input and output. This is the system of system that the aircraft is a part of, e.g. the air transport system, defence system, or a multi-asset rescue mission. However, this is beyond the scope of this paper. When integrating the discrete-

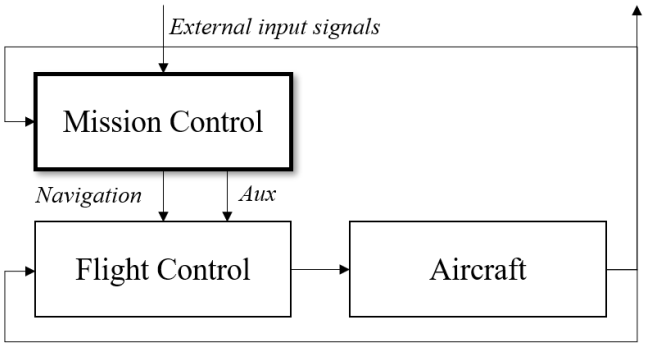

Figure 1 – Model elements for mission simulation.

event dynamics in the existing continuous-time solution, a critical step is the choice of the modelling language. Among the various existing solutions, we rely on the Unified Modeling Language (UML) [11], due to its widespread acceptance and dissemination. The UML diagrams have an intuitive representation that makes them suitable for modeling of control systems. They can also be used as a first representation of the mission in order to derive the requirements on e.g an aircraft.

UML provides two main diagrams to model discrete event behaviour: State Machine and Activity Diagrams. UML State Machines are based on the Statechart formalism of David Harel [2], which is an extension of the conventional formalism of state machines adapted to model parallelism and hierarchy. One example of simulation tool based on Statecharts is the Stateflow toolbox of Matlab Simulink and it can be used for hybrid system simulation [12].

On the other hand, UML Activity Diagram is based on the Petri nets [9]. A Petri net is a bipartite graph composed of two types of nodes: places, which are related to local states, and transitions, which are related to discrete events. Differently from Statecharts, it is a suitable solution to model distributed systems and is therefore adopted in this work.

3. Modelling Activity Diagrams in the TLM formalism

In dynamic systems, the equations of a component is usually not independent of the other components, as they share inputs and outputs with them. When simulating the system, the equations have to be carefully sorted for execution in each time step, which can lead to numerical instability if not implemented appropriately. Alternatively, all equations from all system components are assembled into one system of equations that can then be solved, which is usually not a good solution for scale ability. Another approach is to introduce time delays between all the components as in TLM. Here the components are divided into two types (C-type and Q-type) that are connected with bi-directional ports. In this way all C-type and all Q-type components can be executed in parallel respectively.

The proposal of this paper is to model the main components of Activity Diagram, such as actions and control nodes, in the TLM formalism and provide the appropriate interface to connect them with continuous time components. As a general rule, actions are implemented as C-type components and edge, or edges combined with control nodes, are implemented as Q-type components. As UML specification is not always clear about the expected simulation behaviour of each component, we resort to the Petri net formalism for the definition of the current proposal.

4. Executable UML Activity Diagram for simulation

There is a library of Activity Diagram components in the HOPSAN simulation package. This is a subset of the UML Activity Diagram, and is useful for control of hybrid systems, i.e. systems that

contains both discrete event and time continuous systems. In reality it can be argued that given high enough fidelity all systems containing physical hardware (or models of physical hardware) have time continuous parts.
In order to be executable, the functions of the UML Activity Diagram component have to be defined, Fig. 2 shows an Action node component.

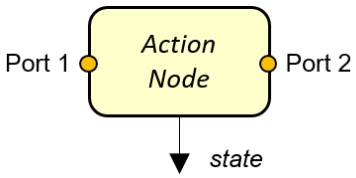

Figure 2 – Executable Action component.

The block is set by a signal from the left node representing a token. The signal can be both positive and negative and last one time step per token. This is propagated to the next port within one time step. The ports are bi-directional and present the presence of a token as a set *state*. The *state* is also displayed in a signal port that can be used to control an external process. The Edge node component, Fig. 3, fires a token if there is a potential across the edge. I.e. the port 1 is set but port 2 is 0 at the same time as the edge is enabled by the event. Two edges cannot be directly connected to each other, but needs to be connected through an Action node.

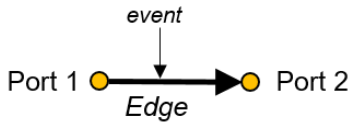

Figure 3 – Executable Edge component.

A sequence of events can then be simulated. In Fig. 4 a UML Activity Diagram of a sequence of three actions is shown. There are also an initial node and a final node. It also contains edges in between that when activated, allows for the transition of a token from one action to another.

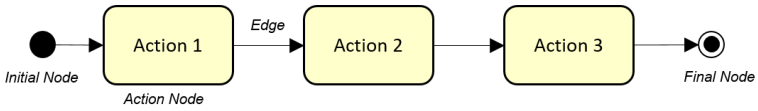

Figure 4 – A simple UML Activity diagram showing a sequence of three activities.

More complex relations can be expressed using decision nodes an merge nodes. There are also fork and join function to create parallel activities.

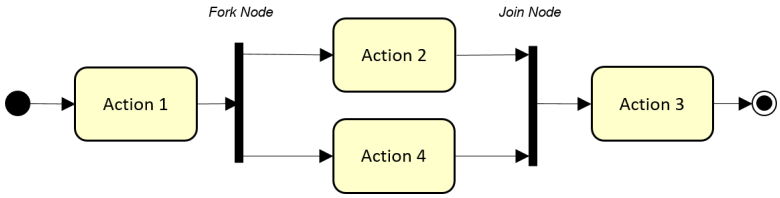

Figure 5 – Activity diagram with fork and join node to create two parallel paths.

With the decision node, and a corresponding merge node, alternative paths can be expressed.

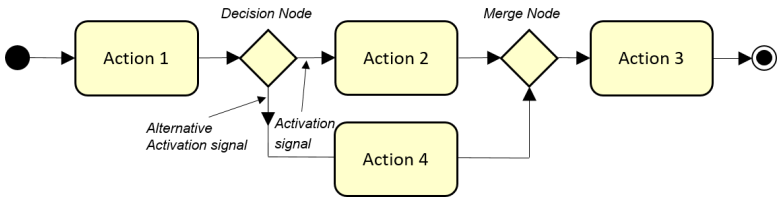

Figure 6 – Activity diagram with decision node and corresponding merge node to create two alternative paths.

A decision node and merge node can also be used to create a loop.

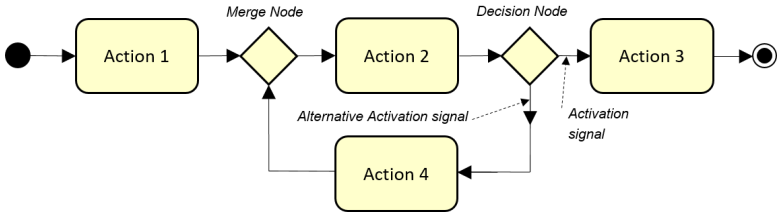

Figure 7 – Activity diagram with merge and decision node to create a loop.

The way point component is used to implement the navigation logic within the mission control system. These are to be activated/deactivated by the action nodes. If the the way point component has a non-zero value on the activate port the reference values for altitude heading and velocity are displayed at the ports. Otherwise these values will be zero. The heading reference is calculated based on the current position of the aircraft (longitude and latitude). The distance to a way point is used to set the signal "set when WP is reached" to one when the distance is less than a given tolerance. This signal can be used e.g. to activate a transition to the next action.

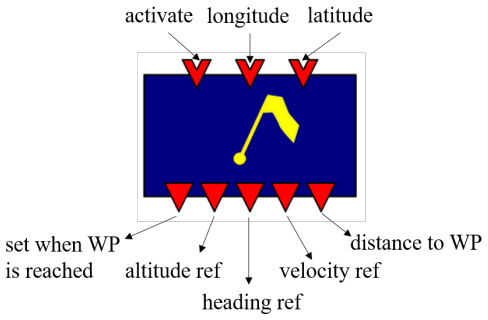

Figure 8 – The Hopsan way point component used to calculate the altitude, speed and heading reference within each segment of the mission controller.

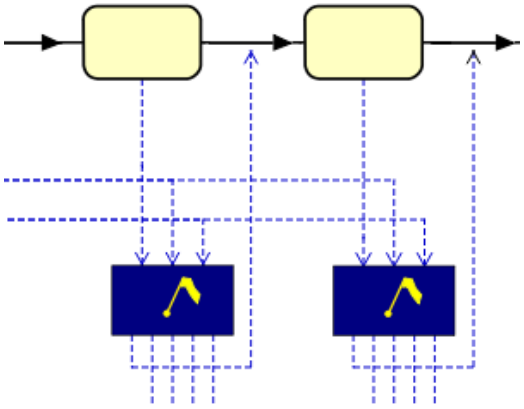

Figure 9 – The waypoint is active when the action node is set. When the waypoint is reached there will be a signal to the edge component to transfer the token to the next action node.

5. Mission modelling and simulation

As an example a simulation of a reconnaissance mission of a UAV is used. First the mission itself is modelled using the UML Activity diagram in HOPSAN and it is then connected to a simulation model of, in this case, an electric unmanned aircraft. In this way a high-fidelity model of the behavioural aspects of the mission is obtained.

The aircraft model is a rather standard six degree of freedom non-linear flight dynamics model, although modeled with TLM-interfaces if e.g. actuator systems are to be incorporated in the simulation model. It is a simplified version of the model presented in [5]. Here, the signals from the flight controller are fed directly into the angles of the aircraft control surfaces.

In Fig. 11 the electric propulsion system is shown. The model includes the electric motor, motor controller and a fuel cell with a hydrogen tank. This can also be swapped with a battery for comparison. In addition to thrust and torque, the model also calculates the mass of the system. This is connected to the flight dynamics model.

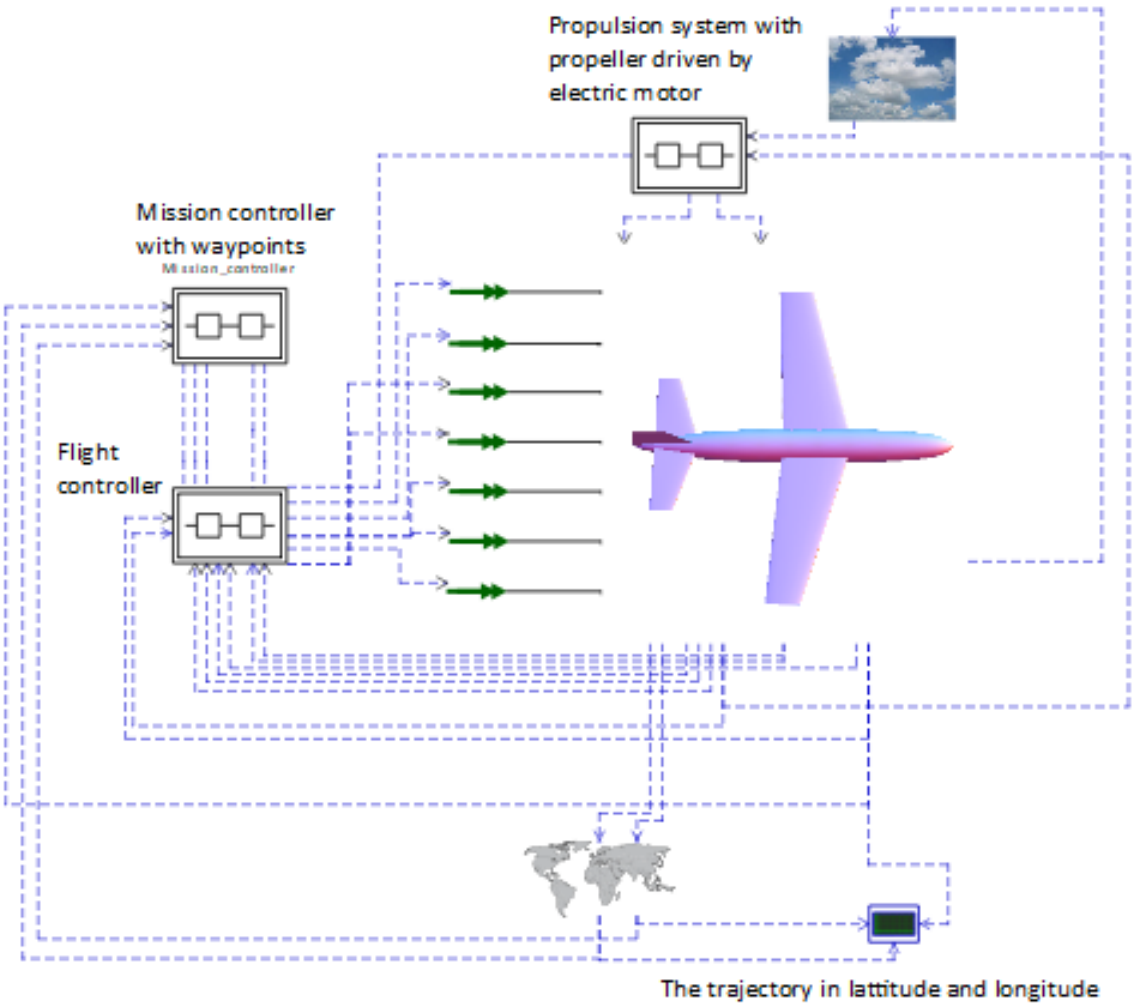

Figure 10 – System simulation model of an electric UAV

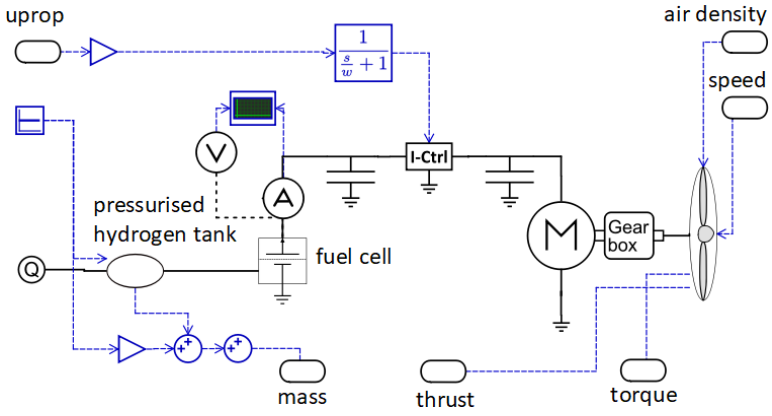

Figure 11 – Electric propulsion system with a fuel cell and pressurized hydrogen tank. (The two capacitance are there for causality reasons for the simulation

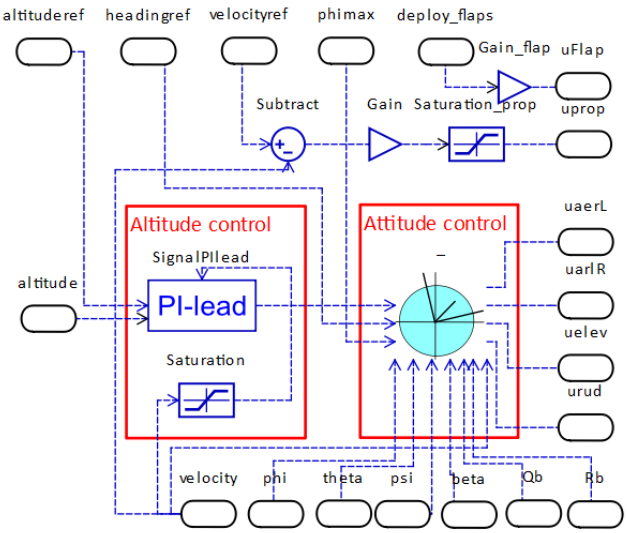

Figure 12 – The flight control system used in the simulation. It is a cascade controller with a altitude controller that gives an reference pitch to the attitude controller

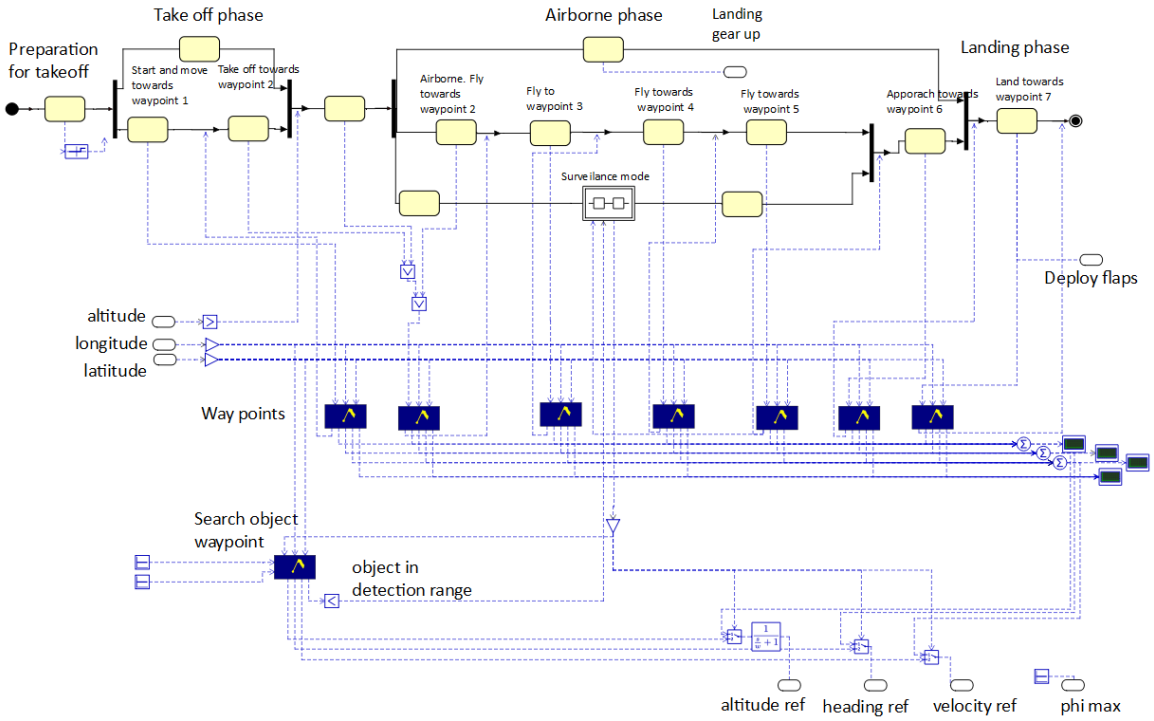

Figure 13 – Mission model using UML Activity diagram to represent different stages in the search mission. The location of the search object is in the lower left part of the diagram. The other way points represents the mission plan.

Fig.13 shows the subsystem representing the mission model in Fig. 10. This module generate the appropriate reference values for the aircraft control system using the position of the aircraft as input. When an action node is activated a signal goes to the appropriate way point to make it active. When an way point is reached, within a certain tolerance, a signal is set to activate the appropriate edge and cause a transition to the next action. Since a way point needs an activation signal to yield nonzero outputs and only one way point is activated at a given time, the output from the way points can simply be added to yield the reference altitude, heading and velocity, to the flight control system.
However, there is also a search object way point that can be activated at any time during the flight. If this comes into range the UAV will make a detour to investigate and circle this for four minutes, before

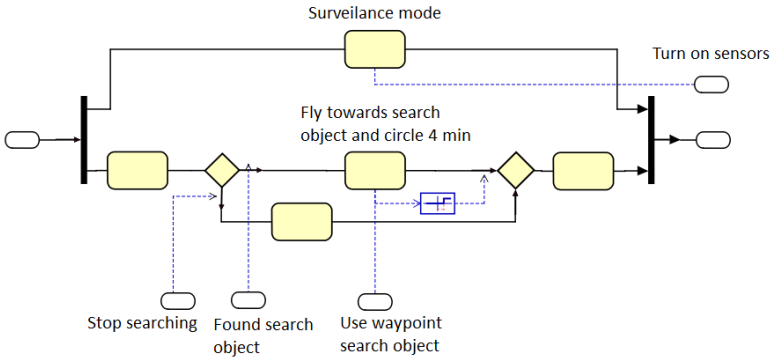

Figure 14 – The subsystem for the search mode. It has two parallel paths. One to indicate the search phase to e.g. turn on sensors, and the other to wait for a signal to go to fly towards the way point and loiter there for four minutes, or to give up and proceed to the next way point.

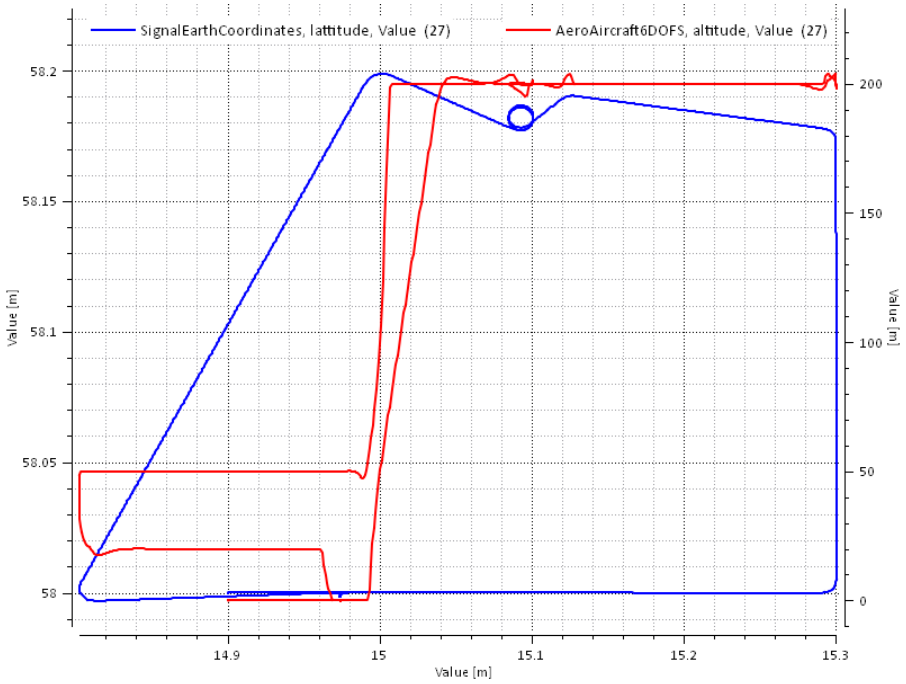

Figure 15 – System simulation of a UAV mission.When the target is detected the UAV starts to fly towards it and circle it for four minutes before proceeding to the next way point.

proceeding to the next way point. Here selectors are instead used, to let the search object way point override the signals from the other way points (lower part of the diagram). The surveillance mode is modelled as a subsystem shown in Fig. 14. This has two parallel paths. One to indicate the search phase to e.g. turn on sensors, and the other to wait for a signal to go to fly towards the way point and loiter there for four minutes, or to give up and proceed to the next way point, using the decision and merge components.

In Fig. 15 the simulation result is shown in the form of horizontal and vertical positions. it can be seen how the UAV changes course when the search objects gets within detection range, and then circles the object for the preset time, before going on to the next way point. This demonstrates how a reactive behaviour can be modeled into the Activity diagram formalism, and shows that it is a viable concept, and a convenient graphical method to model behaviour.

6. Conclusions

In order to have a higher fidelity in modelling of complex system behaviour, UML Activity diagrams have been incorporated in a dynamic simulation for an efficient modelling for whole mission simulation. The UML Activity diagram provides a graphical language to represent complex behaviour. Here a UAV search mission was modelled where a reactive behaviour, upon detection, is modelled. The presented methods for modelling of hybrid system is numerically very efficient and mission simulation can be run hundreds of times faster than real time on a PC. This means that methods that requires that the system is simulated a large number of times e.g. for simulation based optimization or design analytics. In this way it is possible to optimize a system, e.g. an aircraft, using a high fidelity behavioural model where it is evaluated through simulation in a more complete scenario, than is normally used.

Acknowledgments

The authors acknowledge the financial support of the Swedish-Brazilian Research and Innovation Centre (CISB) and the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), Institutional Internationalization Program (PRInt).

Copyright Statement

References

[1] D M Auslander. Distributed System Simulation with Bilateral Delay Line Models. *Journal of Basic Engineering, Trans. ASME*, (June):195–200, 1968.

[2] David Harel. Statecharts: A Visual Formalism for Complex Systems. *Science of Computer Programming*, 8(3):231–274, 1987.

[3] Agostino De Marco, Vittorio Trifari, Fabrizio Nicolosi, and Manuela Ruocco. A simulation-based performance analysis tool for aircraft design workflows. *Aerospace*, 7(11):1–32, 2020.

[4] P B Johns and M O'Brien. Use of the transmission-line modelling (t.l.m.) method to solve non-linear lumped networks. *Radio and Electronic Engineer*, 50(1.2):59–70, 1980.

[5] Petter Krus and Alvaro Abdallah. Modelling of Transonic and Supersonic Aerodynamics for Conceptual Design and Flight Simulation. In Ingo Staack and Petter Krus, editors, *Proceedings of the 10th Aerospace Technology Congress*, volume 162, pages 30–34, Linköping, Sweden, 2019. Linköping University Electronic Press.

[6] Petter Krus, Robert Braun, and Peter Nordin. Aircraft System Simulation for Preliminary Design. In *28th International Congress of the Aeronautical Sciences*, Brisbane, 2012. ICAS.

[7] Petter Krus, Arne Jansson, Jan-Ove Palmberg, and Kenneth Weddfelt. Distributed Simulation of Hydromechanical Systems. In *The Third Bath International Fluid Power Workshop*, 1990.

[8] Petter Krus, Birgitta Lantto, Manuel Rodriguez, and Emilia Villani. Effects of Tailored Control Surface Compliance on Aircraft Stability and Control. In *31st Congress of the International Council of the Aeronautical Sciences*, Belo Horizonte, Brazil, 2018.

[9] Tadao Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580, April 1989.

[10] NASA. *NASA Systems Engineering Handbook*. National Aeronautics and Space Administration, 2007.

[11] OMG. Omg unified modeling language (omg uml), version 2.5.1, 2017.

[12] Anis Sahbani and Jean-Claude Pascal. Simulation of Hybrid Systems Using Stateflow. In Rik Van Landeghem, editor, *Proceedings of the 14th European Simulation Multiconference on Simulation and Modelling. Enablers for a Better Quality of Life*, pages 271–275. SCS Europe, 2000.