CEAS-GNC-2022-036

CEAS EuroGNC 2022
*"Conference on Guidance, Navigation and Control"*
3-5 May 2022 @ Technische Universität Berlin, Germany

# The Virtual Flight Test Environment — A Web-Based Framework for Realistic Testing of Flight Control Laws

**Alexander Köthe** Chief Technology Officer, AlphaLink Engineering GmbH, 10625, Berlin, Germany. alexander.koethe@alphalink.aero

**Janik Hopf** Flight Control Engineer, AlphaLink Engineering GmbH, 10625, Berlin, Germany. janik.hopf@alphalink.aero

**Richard Reinfeld** Systems Engineer, AlphaLink Engineering GmbH, 10625, Berlin, Germany. richard.reinfeld@alphalink.aero

**Daniel Cracau** Chief Executive Officer, AlphaLink Engineering GmbH, 10625, Berlin, Germany. daniel.cracau@alphalink.aero

*ABSTRACT*

**With current open-source tools, flight control design and testing is tedious. The presented virtual flight test environment provides a complete toolchain for the safe testing of flight control software in a virtual flight test using the web browser. Any UAS can be integrated as a model into the test environment and virtually tested at any location. This reduces development costs through fast and safe testing of software prototypes. The new tool also supports the education of flight control engineers through i) a variety of predefined control-loop structures and ii) a direct transition from virtual to real flight test.**

**Keywords:** Flight Simulation; Digital Twin; Aircraft Control

## Nomenclature

| | | |
|---|---|---|
| $H$ | = | Altitude |
| $k$ | = | Control gains |
| $L_p$ | = | Roll damping |
| $L_\xi$ | = | Influence of the aileron on the roll moment |
| $p$ | = | Roll rate |
| $T_p$ | = | Time constant of low pass filter |
| $\Phi$ | = | Bank angle |
| $\Theta$ | = | Pitch angle |
| $\xi$ | = | Aileron deflection |

## 1 Introduction

Testing of flight control software is typically done in several steps to ensure that the designed flight controller is functional in reality [1]. Development teams are often involved in the design process - sometimes spread over several countries. In this case, it must be ensured that the same development toolchain

is available to all engineers. To make this possible, AlphaLink has developed the Virtual Flight Test Environment (VFTE). It is a web-based test environment that requires only a browser. The core of the web-based test environment is a non-linear flight dynamic model of a UAS. In addition to the movement of the aircraft, actuator and sensor dynamics are also simulated. This leads to a digital representation of the real aircraft and, thus, to a digital twin. In the web-based development environment, engineers can upload their controllers developed in Simulink and test them in a flight simulation. Atmospheric disturbances can also be simulated. Scopes enable a quick evaluation of the designed controllers. Different locations can be used as landscape sceneries. Currently, a generic scenery and the waterway cross in Magdeburg, Germany, are available. This allows testing of all cascades of a flight control system up to the navigation control before the controller is tested in the real flight test. This saves costs and the development goal can be achieved significantly faster. Figure 1 shows the web-based visual of the VFTE.

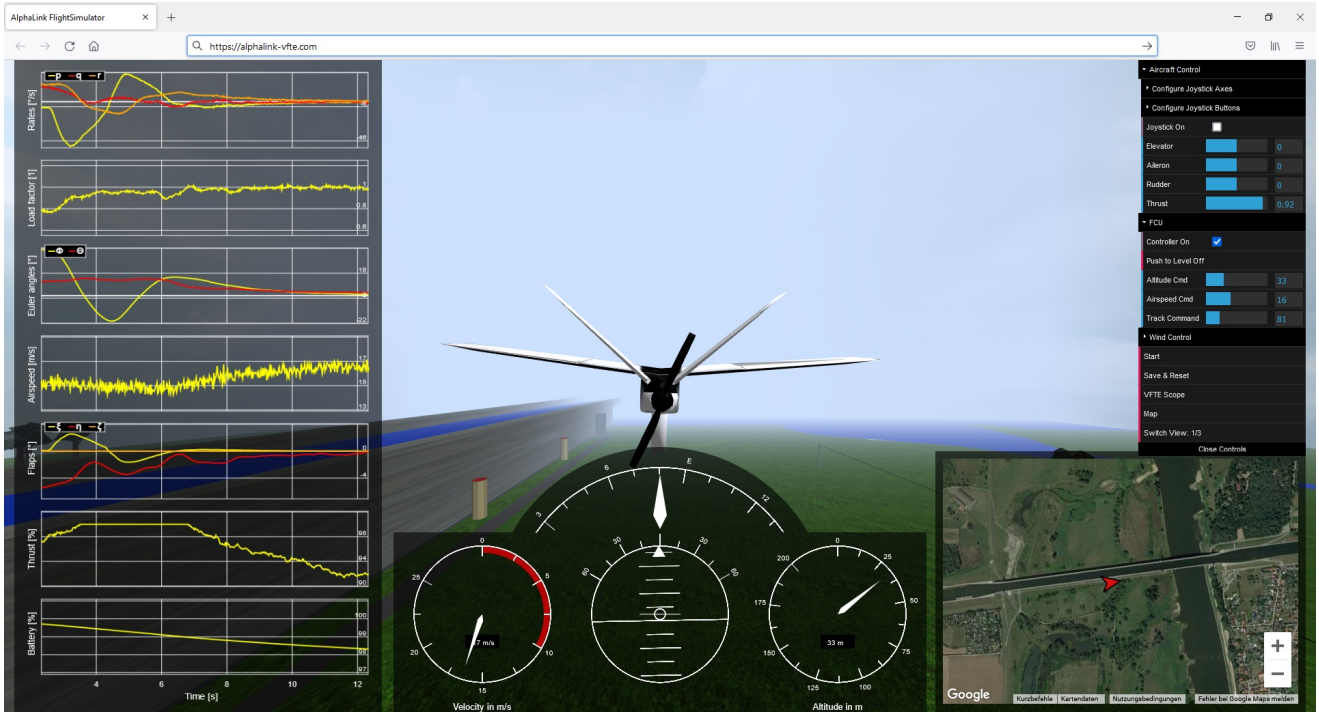


**Fig. 1 Flight Simulation in the Web Browser**

The VFTE also supports the education of flight control engineers. The *Unmanned Aircraft Experimental System* (UAXS), or Flying Lab, is the base for this (cf. Fig. 2). The UAXS is a system that comes ready to fly. Flight control system, sensors and actuators are already integrated in the aircraft. A Simulink template is provided for the integration of controllers. The Simulink model contains all available sensors as inputs and all actuators as outputs. Scientists and students can integrate

**Fig. 2 UAXS in the Configuration as Hardware-in-the-Loop Simulator**

their designed flight controllers into this template and then use them without having to program any code themselves. To make it easier to get started, twelve predefined control-loop structures are already available in the web-based VFTE. Using a linear model, students can determine the controller parameters for the existing control-loop structures and enter the parameters in the web browser. The designed controllers can then be tested in the virtual flight test. If the test is successful, a Simulink model of the flight control software (including the designed parameters) can be downloaded, which can be integrated directly on the UAXS. On the one hand, this process makes it possible to learn flight control quickly by using a web browser. On the other hand, it enables a rapid transition from simulation to real flight test. Because no additional software is required - except for the web browser - lecturers can use the VFTE directly in class and demonstrate the introduced theoretical concepts in practical applications.

This paper explains the architecture of the VFTE and shows the possibilities for development teams. It further explains how the teaching of flight control can be made more attractive with this web-based solution, so that students develop an in-depth understanding of flight control.

# 2 Structure of the VFTE

Virtual testing of flight control software is already taking place today. In most cases, a flight simulator such as *X-Plane* or *FlightGear* is used as a visual for this purpose. The Simulink model of the model and controller, which runs in real time, is connected to this visual via a network connection. This requires a high level of in-house development effort to ensure smooth and reliable data exchange. In addition, an input mask is required as a flight control unit and a connection to the controller (keyboard or joystick) is necessary. The VFTE lets users jump all these steps, requiring only a web browser. Figure 3 shows the basic architecture of the VFTE. The entire implementation is done exclusively using JavaScript and HTML.

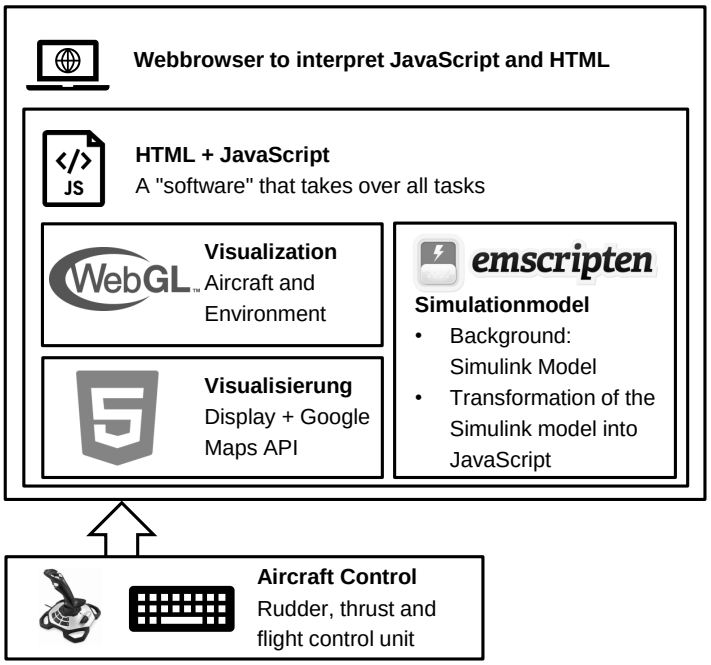


**Fig. 3 Structure of the VFTE**

## 2.1 Integration of the Flight Dynamic Model in JavaScript

The core of the flight test environment is a non-linear simulation model of the UAS [2]. It contains both the flight dynamics and the simulation of aircraft systems. The standard model comes with the following features

- full nonlinear flight dynamics including a wind and turbulence model,
- nonlinear aerodynamics including stall behaviour,
- a thrust and power model,

- a ground model that allows landing and testing of auto-land controllers, and
- identified sensor models including offset, bias, drift, and noise from the real sensors.

In general, any model like a copter can be integrated into the VFTE. This requires a Simulink model that can be converted into C++ code using the Embedded Coder from The Mathworks [3]. The MIT tool *emscripten* is used to convert the generated C++ code into JavaScript [4]. In addition to the automatically generated code, a class is written that takes the inputs of the control devices and control law software, initialises the flight dynamic model, executes the calculation step, and then provides the outputs of the calculation. With this class and a template in Simulink, arbitrary flight dynamics can be simulated in the VFTE. In addition to different unmanned fixed-wing aircraft, a copter from the company DiAvEn has already been integrated into the VFTE (cf. Fig. 4).

## 2.2 Visualisation and Aircraft Control

To visualise the UAS and the environment, the application programming interface *Three.js* is used [5]. Three.js allows the display of animated and fixed three-dimensional computer graphics in the web browser using WebGL. It is a JavaScript library allowing the visualisation of the aircraft movement since it is directly connected to the flight dynamic model. The scopes and map shown in Fig. 1 are HTML elements that are controlled by JavaScript. In the scopes, the roll rate, pitch rate, yaw rate, load factor, pitch angle, roll angle, airspeed, flaps, thrust, and battery charge level of the last 10 seconds are displayed continuously. With these scopes, the performance of the controller can be evaluated immediately when it is activated. The flight control unit is located on the right side of the browser window (cf. Fig. 1). This can be used to set reference variables for the controller or to activate or deactivate certain controller modes. In addition, the perspective can be changed on the right side. Three perspectives are available: i) the view from behind the aircraft, ii) the first-person view or iii) a view from a fixed observation position on the ground. The latter allows pilots a realistic scenario of the real flight test.

**Fig. 4 Copter Labfly in the VFTE**

The aerodynamic flaps and thrust can be controlled either via sliders in the digital control panel or through direct inputs from the keyboard or a joystick. If a flight control law is used for manual flight, the target values such as roll rate, pitch rate or load factor can also be set with the help of the joystick. The VFTE supports all joysticks and controllers that are compatible with the PC used. A short calibration of the joystick may be required during the first use; after that it will be set up automatically and is available immediately.

## 2.3 Use of Flight Control Laws

There are two ways to use flight control laws: i) one of the twelve predefined control-loop structures or ii) the user's own controller that can be uploaded into the VFTE. The predefined control law structures include a bank angle control law, a heading control law, a rate command attitude hold (roll axis) control law with and without yaw damper, an airspeed control law with elevator only, a pitch attitude control law with elevator only, an altitude control law with elevator only, a total energy control law, an airspeed and altitude control law based on the total energy control law, a rate command attitude hold control law for the pitch axis, a rate command attitude hold control law for both pitch and roll axis, and a full autopilot control law with airspeed, altitude and heading control. Those predefined control-loop structures are taken from standard literature [1, 6] and are shown in Fig. 5. A Simulink template is available for advanced users to integrate further control structures. After parameters have been entered in a certain control loop structure, they are stored in a database so that they are available for future

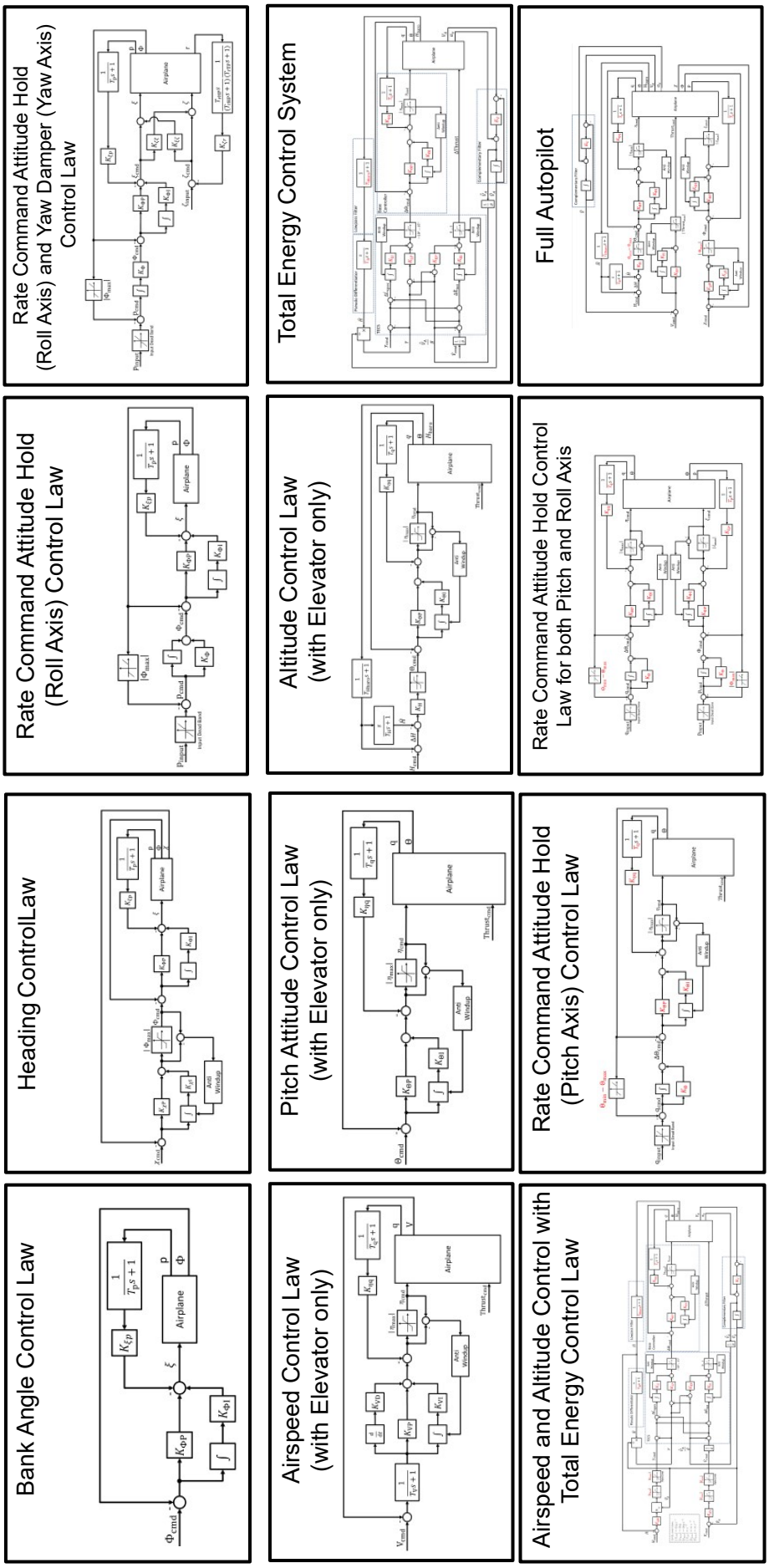


**Fig. 5 Illustration of the predefined control loop structures in the VFTE**

applications. Because the predefined control laws incorporate different structures, they require different sets of parameters, which in turn are stored separately.

The compilable flight control software can also be downloaded. There, all entered parameters are integrated. This software can be implemented directly on the UAXS for a real flight test. It should be noted that the controllers are based on a continuous model. In the background, the continuous model is converted into a discrete model using bilinear z-transformation. As a result, it is executable on the target hardware without further adjustments. The execution rate of the flight controller is 100 Hz, both in simulation and in the real aircraft.

In addition to the predefined control-loop structures, own controllers can be uploaded. For this purpose, a Simulink template is provided, in which the controllers are integrated. The controller must then be converted into C++ code using Embedded Coder and the resulting source code and header files are uploaded. On the server, the code is converted into JavaScript code so that the own controller can be tested in the VFTE. This is particularly advantageous for developers. The measurement data that the controller receives as inputs are identical in quality to real measurement values, at least for the predefined model of UAXS. They contain noise and bias. In addition, sampling rates (e.g. GPS with 5 Hz) are taken into account in the model. This allows extensive testing of the controller in the virtual flight test before the controller is tested in the real flight test. It should be noted that only discrete transfer elements can be used in the controller model.

## 2.4 Evaluation of the Virtual Flight Tests

All measurement data of the virtual flight tests are saved continuously. At the end of the flight test, the stored data is provided as an Excel spreadsheet. The data can then be further processed in *MATLAB* or in any other programme. This allows a comprehensive evaluation of the virtual flight test and provides insights for further improvements in the flight control law design.

# 3 Use Cases

The VFTE benefits for teaching and research will now be demonstrated. Two examples are discussed guiding the reader through the actual use of the VFTE.

## 3.1 Example I: Bank Angle Control Law

One of the simplest control loops in the flight control system is the wings level controller. It consists of a roll damper and a bank angle controller. This controller is suitable for both flight control and control engineering courses because the physics of the controlled system is particularly simple. This use case shows how teachers can make their classes in flight control or control engineering in general more realistic by using the VFTE.

### *3.1.1 Plant Model*

The modelling is shown in Fig. 6. The initial point of the modelling for the aircraft roll motion is the angular momentum equilibrium. This states that the sum of the external roll moments is equal to the product of the roll acceleration and the moment of inertia about the roll axis. External roll moments are caused by the ailerons and the roll motion of the aircraft. The ailerons increase the lift on one side and decrease it on the other, which creates a roll moment. If a wing moves upwards due to a rolling motion, the air flows downwards on this wing. This reduces the angle of attack and, thus, also the lift. On the side, where the wing moves downward due to the rolling motion, the angle of attack and, thus, also the lift is increased. That effect is called roll damping. This results in the first-order linear differential

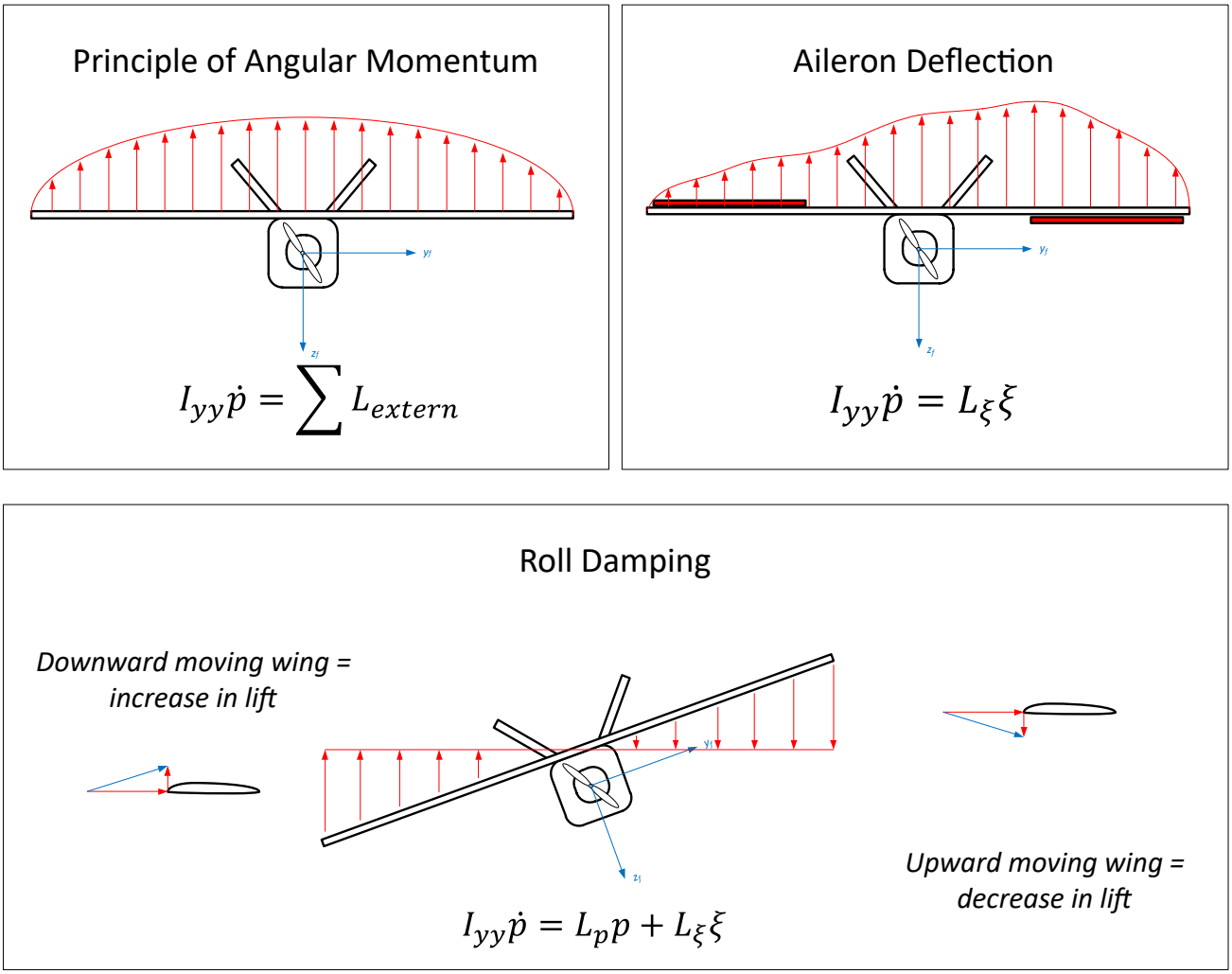


**Fig. 6 Illustration of physical effects for modelling the aircraft roll motion**

equation with

$$\dot{p}(t) = \frac{1}{I_{yy}} \left( \hat{L}_p \, p(t) + \hat{L}_\xi \, \xi(t) \right). \tag{1}$$

In terms of control engineering, the roll rate is a state variable and the aileron is a control variable. However, a simple kinematic relationship is still missing. The bank angle results from the integration of the roll rate or the derivative of the bank angle is equal to the roll rate with

$$\dot{\Phi}(t) = p(t). \tag{2}$$

This provides a model in the form of a first-order differential equation system with

$$\begin{bmatrix} \dot{p}(t) \\ \dot{\Phi}(t) \end{bmatrix} = \begin{bmatrix} L_p & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} p(t) \\ \Phi(t) \end{bmatrix} + \begin{bmatrix} L_\xi \\ 0 \end{bmatrix} \xi(t) \quad \text{with} \quad L_p = \frac{\hat{L}_p}{I_{yy}} \quad \text{and} \quad L_\xi = \frac{\hat{L}_\xi}{I_{yy}}, \tag{3}$$

which can be used for the design of the control law. The bank angle is thus another state variable. The VFTE provides a *MATLAB* file, where the quantities $L_\xi$ and $L_p$ are available. Teachers can use these values directly for their classes. A complete state space for longitudinal and lateral motion is, hence, available that can be downloaded after login.

### 3.1.2 Control Law Structure

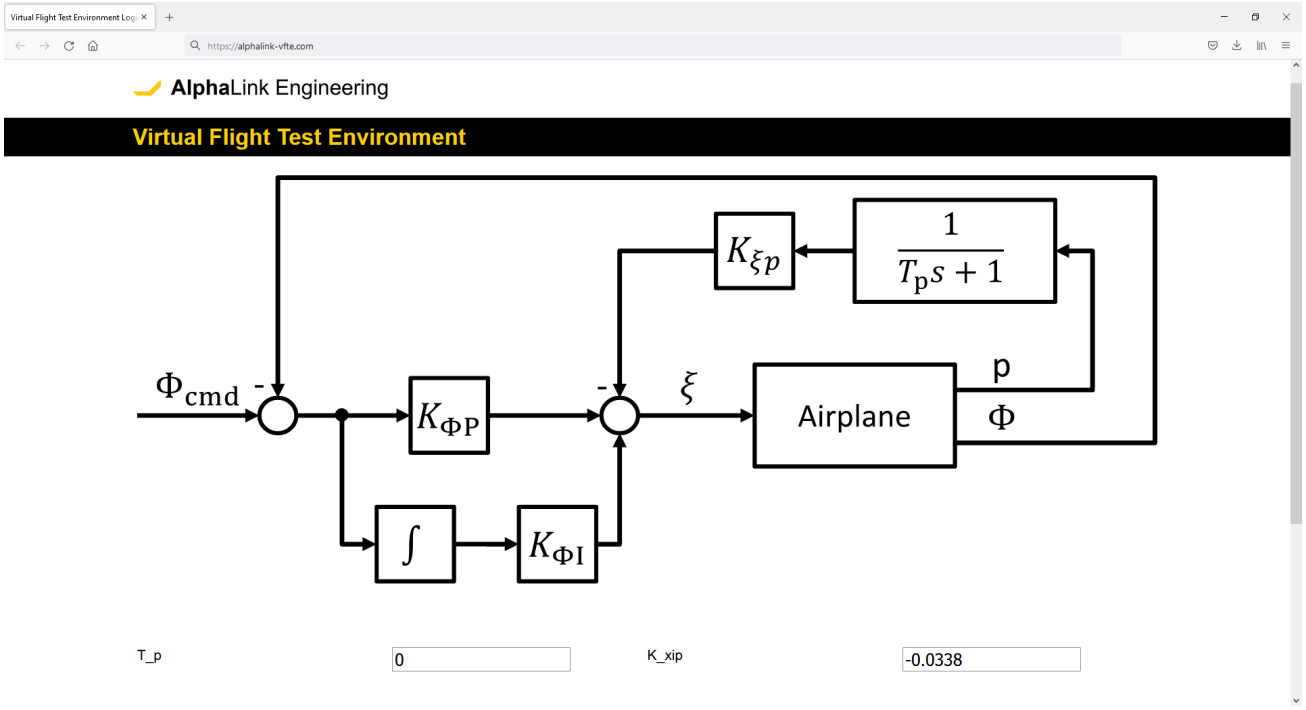


**Fig. 7 Screenshot of the parameterisation of the control loop structure for Lesson 1 (Bank angle control law) of the VFTE**

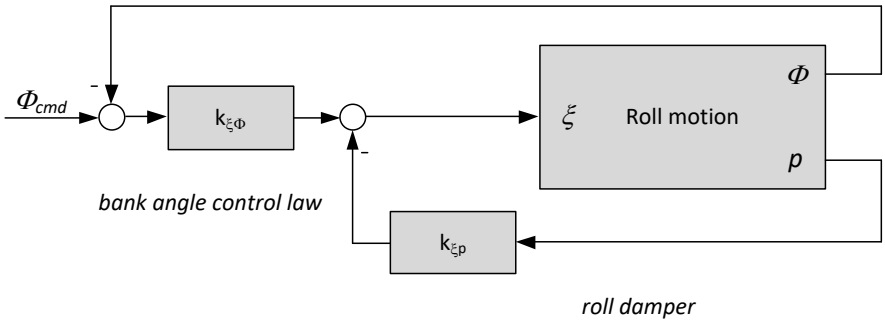


**Fig. 8 Example of a modified control loop structure for Lesson 1**

For the bank angle control law, a predefined control loop structure exists in Lesson 1 of the VFTE. The structure consists of

- a low-pass filter for filtering the roll rate,
- a roll damper as a proportional controller, and
- a PI controller for the bank angle.

Figure 7 shows the control loop structure as presented in the VFTE Lesson 1. The structure of the control loop is shown in the form of a block diagram. Individual blocks have parameters that can be entered in a mask below the block diagram. Individual elements of the control loop structure can be set to zero, so that teachers can also use slightly modified control loop structures in their classes. For example, if the low-pass filter shall not be considered, the time constant $T_p$ can be set to zero. For demonstration purposes, a control loop structure shown in Fig. 8 will be used, omitting the I controller for bank angle and the low-pass filter for roll rate from the original control loop structure. The gains can be calculated

by different methods. The simplest methods would be state-space control. This results, for example, in the parameters

$$k_{\xi p} = -0.0153 \quad \text{and} \quad k_{\xi \Phi} = -0.672. \tag{4}$$

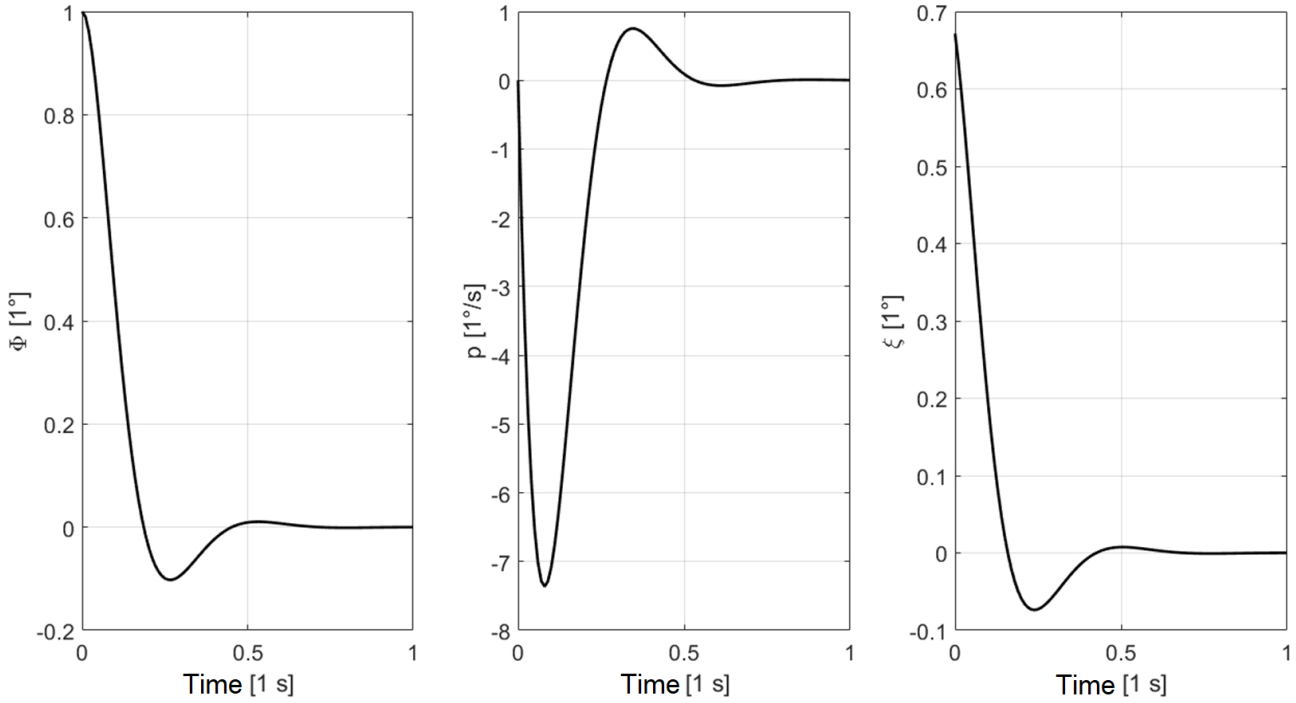


**Fig. 9 Linear simulation results for the modified control law structure of VFTE Lesson 1**

In linear simulations with the established model of the roll motion, excellent results are obtained (cf. Fig. 9. However, if the students enter these parameters into the input mask and start the simulation afterwards, an unstable oscillating behaviour results. The simulation results are shown in Fig. 10.

### *3.1.3 The Difference between Model and Reality*

The reason for the unstable behaviour of the closed loop is that no low pass filter was used and the gains were set too high. In reality, the measured values are very noisy and if the bandwidth of the closed loop is too high, this noise is amplified so that an unstable behaviour follows.

Students may not understand this effect from the teacher's conventional blackboard lecture only. They benefit from experiencing the effects in flight tests. And this is exactly what you can do with the VFTE. Through the fast, web-based interface, gains can be tested quickly in a nonlinear simulation environment without large effort. These tests can be incorporated directly into the classroom. This renders teaching not only more relevant to subsequent practical application, but is also more attractive for the participating students.

Furthermore, the example of Lesson 1 in the VFTE can also be used to explain why a PI controller is useful even if there is an integral component in the path. Also additional control variable limitations can be clarified using this example.

## 3.2 Example II: Testing New Control Law Concepts

Even if a lot of research has already been done in flight control, sometimes new control concepts arise that need to be practically validated. Before the flight test is carried out, the VFTE can be used

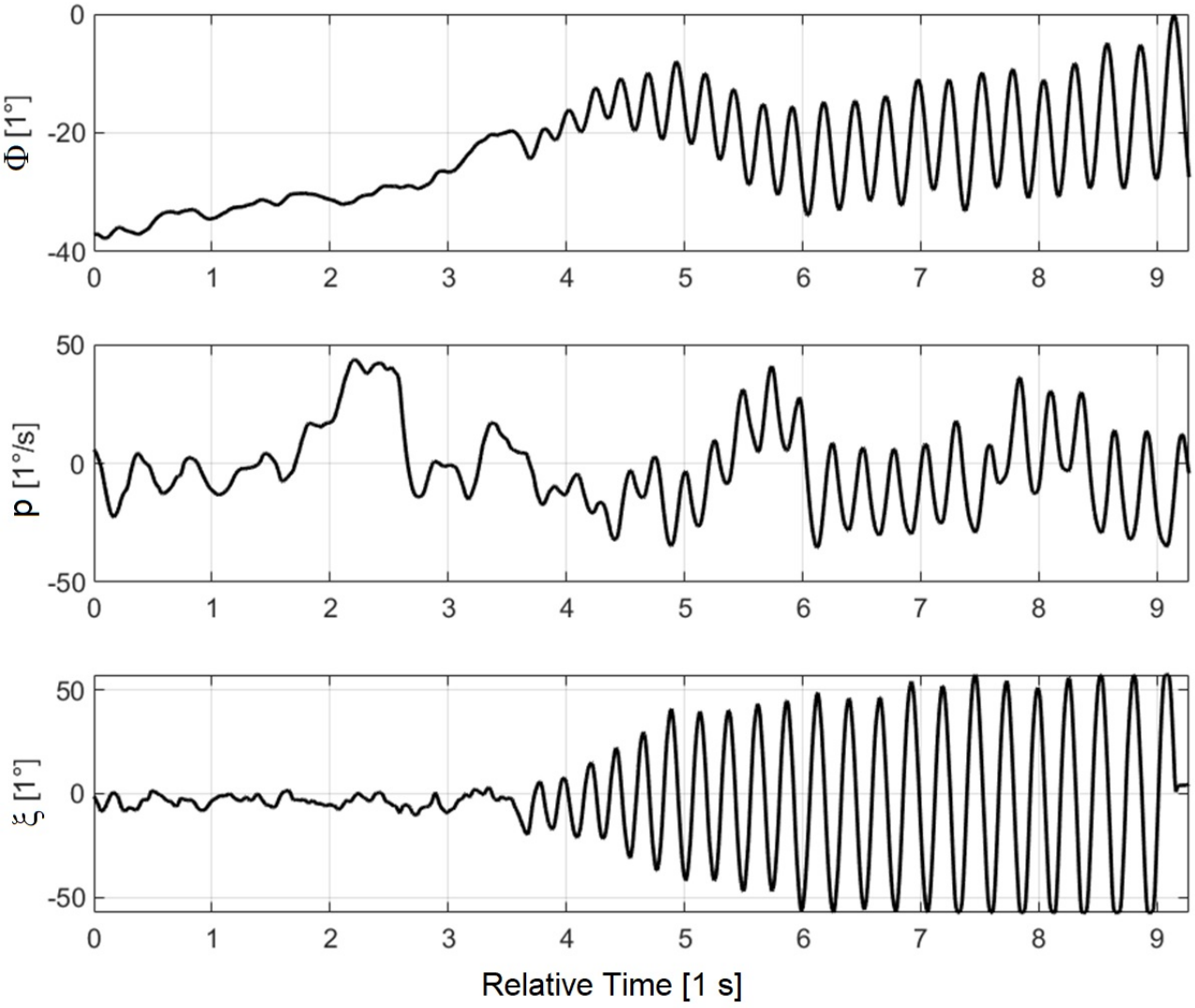


**Fig. 10 Simulation results obtained in the VFTE for the modified control law structure of Lesson 1**

to check the implementation of the idea in the flight control software and to get a realistic impression of the functionality. This feature allows researchers to make changes more quickly and, hence, carries enormous time and cost potential.

In order to test own control loop structures in the VFTE, a Simulink template is provided (cf. Fig. 11). All inputs are listed on the left side of the model. These are measurement data, such as data from the IMU, the GPS or from the barometric measurement system, as well as control inputs from the joystick and user-defined inputs like numbers and buttons. These user-defined inputs can be used, for example, to specify reference values for the flight control system. On the right side of the model are the outputs to the control surfaces. In addition, up to 20 outputs are available for customisation by the users. Those are saved in the log file during flight. They can be used, for example, to store values calculated within the model. All areas between inputs and outputs can be freely defined by the user. However, there is the restriction that only discrete blocks may be used. Continuous transfer functions are not permitted.

After the controller model has been designed, MATLAB can be used to generate C++ code from the model. This C++ code is subsequently uploaded to the VFTE. Each user can only access their own model. On the server, the controller model is converted into JavaScript code and integrated into a personalised VFTE. Thus, the VFTE is not only useful for teachers and students, but also for researchers, who want to test their controller approaches in a realistic, but nevertheless safe environment.

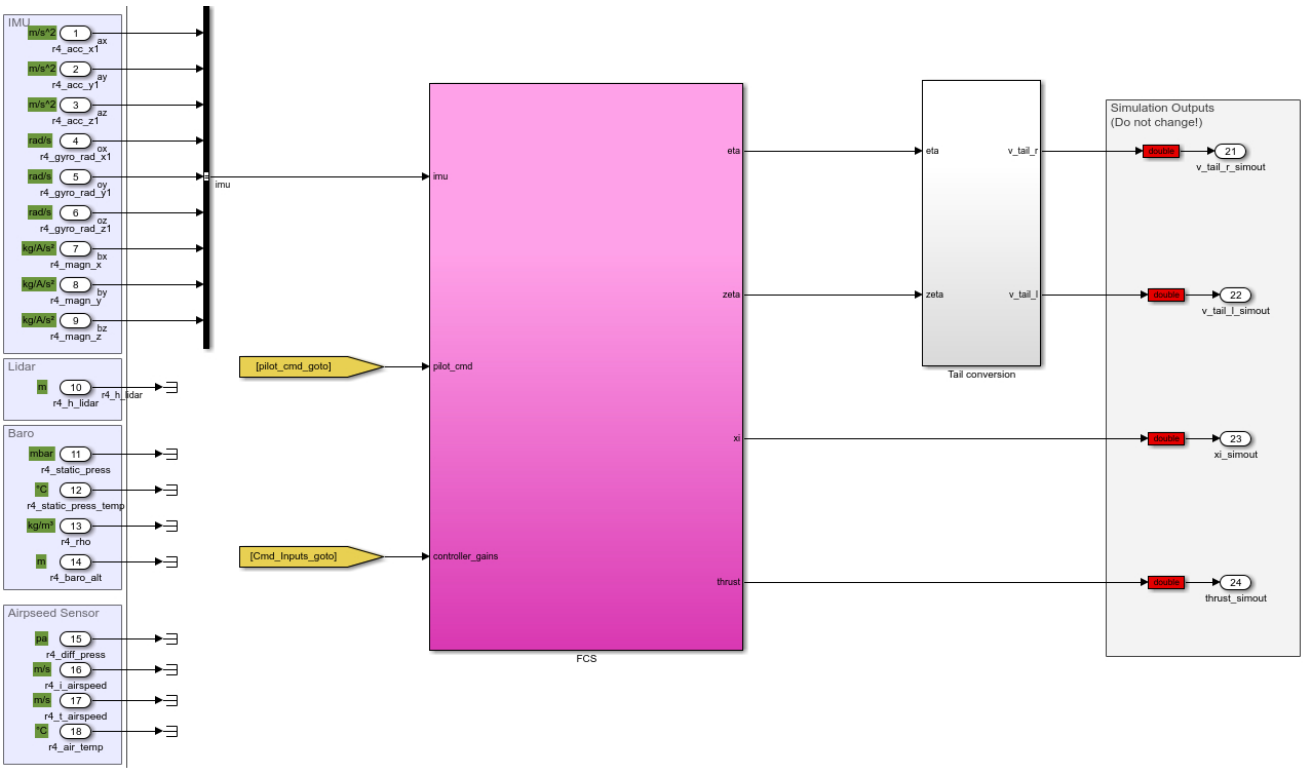


**Fig. 11 Segment of the Simulink template for the integration of custom control loop structures into the VFTE**

Finally, it should be noted that this Simulink template corresponds exactly to the template used for the physical set of the Flying Lab. This means that controllers that have previously been successfully tested in the virtual world can be brought directly into the real-world flight test. This saves development time and costs.

# 4 Use of the VFTE with a Hardware-in-the-Loop Simulator

In addition to using the VFTE for virtual flight tests, the visualisation (including map and scopes) can also be used for hardware-in-the-loop simulations. AlphaLink, in collaboration with Vector Informatik GmbH (Vector), has developed a hardware-in-the-loop (HiL) simulator for Pixhawk-based flight control systems. The HiL enables even more intensive testing before the actual flight test [7]. In such HiL testing, the movement of the aircraft, the environment, and the behaviour of sensors and actuators are replaced by a mathematical model to perform a realistic simulation. All the remaining physical components are tested with a physical aircraft in the real world.

Figure 12 shows the architecture of the HiL. The main component is the aircraft system provided by the AlphaLink UAXS. This includes the Nano Talon as aircraft and the Pixhawk 4 Mini (Pixhawk) as flight controller inside. The Pixhawk will be the desired target hardware to run the controller structure that shall be tested. The aircraft system in the HiL is exactly the same as for ordinary flight testing. Hence, it takes the RC commands as inputs and sends the flight data as outputs to the ground station, in this case the software QGroundControl. The difference between the HiL simulation add-on and the basic Flying Lab is the modified flight stack, i.e. the Pixhawk is interacting with the simulation environment instead of the real world. This is achieved through the interface component. The HiL Simulator is using the CAN bus for the communication between the hardware and the simulation loop. This CAN bus is controlled using the original hardware and software from Vector. The interface provides the hardware with the required sensor data, while transmitting the actuator commands from the hardware

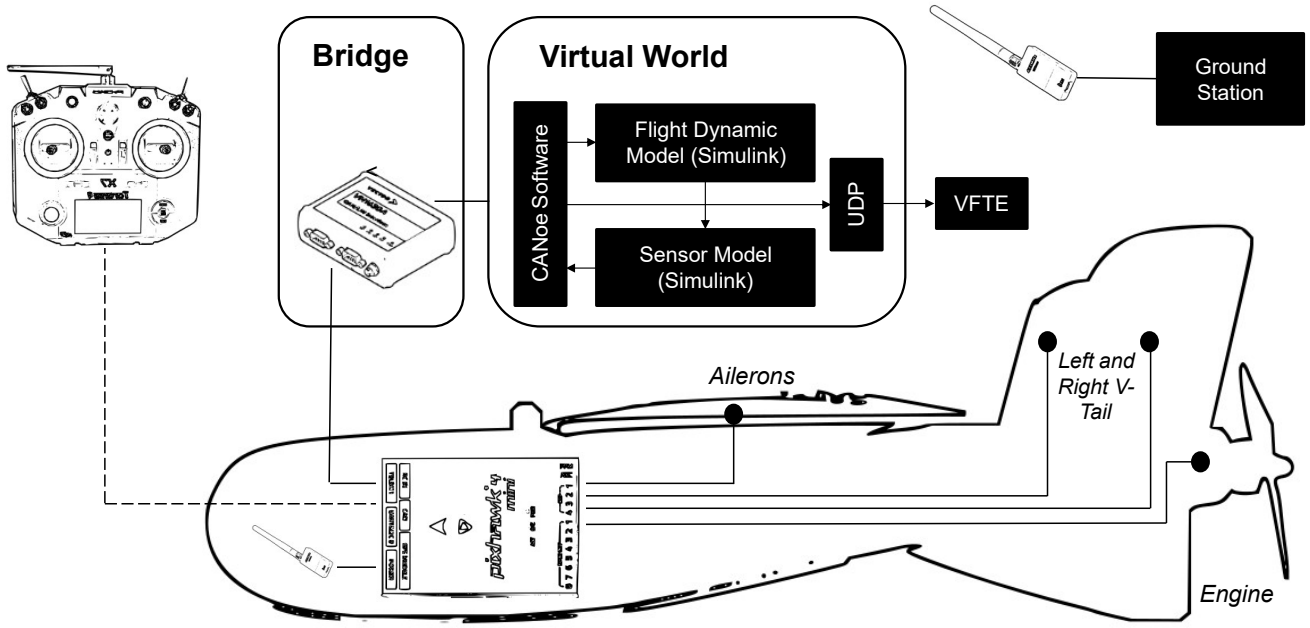


**Fig. 12 Architecture of the Hardware-in-the-Loop Simulator**

to the simulation model. The simulation model is completely identical to the flight dynamic model in the VFTE. The simulation is done using Simulink. It takes the actuator commands and the initial start conditions for the desired testing situation as inputs. With this, the model computes all dynamics ranging from the flight dynamics to the sensor and the actual actuator behaviour. The objective is providing the sensor data of the aircraft to the flight controller under nearly real-world conditions. To visualise the simulated flight of the aircraft, the simulation model is connected to the VFTE.

The advantage of this simulator is the opportunity to test the controller structure directly on the target hardware without the risk of flight accidents. The testing proceeds in a safe environment, while all remaining aspects from flight testing like RC control and ground monitoring through the virtual ground station are given. In addition, the movement of the control surface can be seen in real time at the physical aircraft model, while the resulting dynamic behaviour is observed in the VFTE. Also critical situations like sensor failure can be tested without the consequences of any physical damage. Figure 13 shows the HiL in use. A dedicated ground station completes the HiL setup. With this configuration, a lot of error cases can be simulated so that they are avoided in the real flight test. Furthermore, pilots can be trained in how to fly the UAS using the RC and how to interact with the control law being tested.

# 5 Conclusion

In this paper, a new web-based test platform for the development and testing of flight control laws was presented. The benefits for teaching, research, and development were demonstrated on the basis of three use cases. In education, the advantage for students is that they are already confronted with problems of real flight tests in the lecture room. The future flight control engineers can test control concepts and the influence of changes in control parameters by using a web browser only. Particularly nowadays, this offers great advantages especially for digital teaching. Training costs are reduced because flight tests can already be fully prepared in the virtual world. Universities can integrate their own UAS into the VFTE or use the already available and validated model. Researchers have the opportunity to test new approaches by uploading their own flight control law models in the virtual world. In addition, global collaboration is enabled through the use of a similar toolchain. Costs for research and development can be significantly

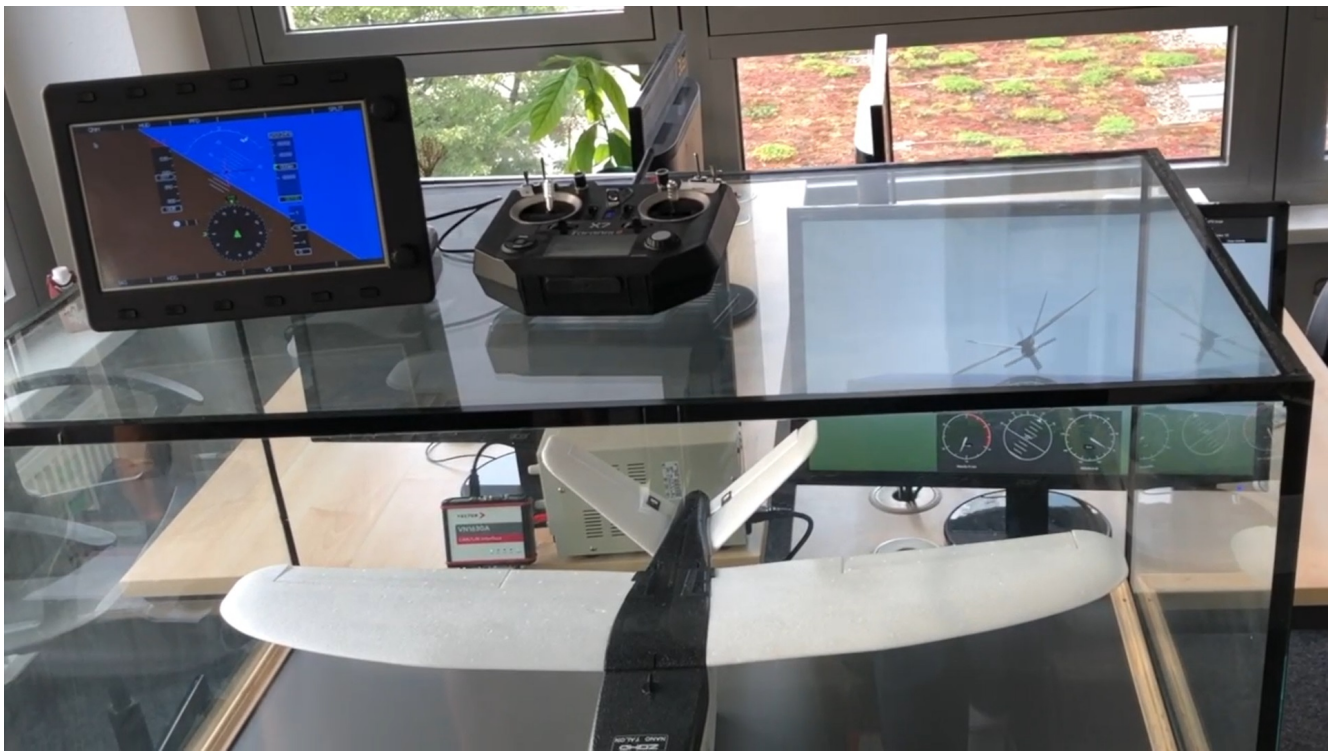

**Fig. 13 Components of the Hardware-in-the-Loop Simulator**

reduced through savings in travel expenses, personnel costs and material costs. In combination with the HiL simulator, it is not only possible to detect errors in the development of flight control laws at an early stage, but also to systematically investigate system failures and their effects on flight control laws.

The VFTE is hence designed to offer the advantage of virtual flight tests to a broad audience. It provides a shortcut to the previously tedious process of combining different tools available, e.g. open source, for flight control design and immediate testing.

## Acknowledgements

The authors would like to thank Mara Krachten, Jakob Dommaschk, Nicolai Block and Paul Wormann for their cooperation. Furthermore, the authors would like to thank Vector Informatik GmbH, in particular Arne Brehmer and Jörn Haase, for their support in realising the projects.

## References

[1] R. Brockhaus, W. Alles, and R. Luckner. *Flugregelung*. Springer-Verlag, 2013.

[2] J. Hopf, J. Dommaschk, N. Block, R. Reinfeld, M. Krachten, P. Worrmann, D. Cracau, and A. Köthe. Unmanned aircraft experimental system: The flying lab for applied flight control and flight mechanics. In *Deutscher Luft- und Raumfahrtkongress 2020*, Bonn, 2020. Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V.

[3] J. Grant. *Code Generation in Simulink*. CreateSpace Independent Publishing Platform, 2018.

[4] Alon Zakai. Emscripten: an llvm-to-javascript compiler. In *Proceedings of the ACM international conference companion on Object oriented programming systems languages and applications companion*, pages 301–312, 2011.

[5] Brian Danchilla. Three. js framework. In *Beginning WebGL for HTML5*, pages 173–203. Springer, 2012.

[6] Brian L Stevens, Frank L Lewis, and Eric N Johnson. *Aircraft control and simulation: dynamics, controls design, and autonomous systems*. John Wiley & Sons, 2015.

[7] André Kaden, Bernd Boche, and Robert Luckner. Hardware-in-the-loop flight simulator–an essential part in the development process for the automatic flight control system of a utility aircraft. In *Advances in Aerospace Guidance, Navigation and Control*, pages 585–601. Springer, 2013.