International Forum on Aeroelasticity and Structural Dynamics
IFASD 2024
17-21 June 2024, The Hague, The Netherlands

LESSONS LEARNED FROM FLIGHT TESTING ACTIVE FLUTTER SUPPRESSION TECHNOLOGIES

Julius Bartasevicius[1], Dániel Teubl[1], Thomas Seren[1], Fanglin Yu[1], Mirko Hornung[1], Balint Vanek[2], Daniel Balogh[2], Szabolcs Tóth[2], Mihály Nagy[2], Balázs Fritsch[2], Keith Soal[3], Thiemo Kier[4], Özge Süelözgen[4]

[1]Technical University of Munich, the Chair of Aircraft Design
Boltzmannstrasse 15, 85748 Garching, Germany
julius.bartasevicius@tum.de

[2]HUN-REN SZTAKI
Kende u. 13.-17., H-1111, Budapest, Hungary

[3]German Aerospace Center DLR, Institute of Aeroelasticity
Bunsenstr. 10, Göttingen, Germany

[4]German Aerospace Center DLR, Institute of System Dynamics and Control
Münchener Straße 20, Weßling Germany

Keywords: FLIPASED, active flutter suppression, aeroelasticity, UAV, flight testing

Abstract: Active flutter suppression (AFS) was demonstrated in real-life conditions during the project FLIPASED (Flight Phase Adaptive Aero-Servo-Elastic Aircraft Design Methods). It was done by passing the flutter speed with a subscale demonstrator called P-FLEX. Two different flutter suppression controllers were used in this case, both of which proved to dampen the flutter modes, allowing the demonstrator to further increase the airspeed. This article presents the background of the demonstrator, its systems, and the testing, the challenges faced during the tests and finally the important lessons learned during the process.

1 INTRODUCTION

Developing and validating an active flutter suppression system was one of the main goals of the two European projects FLEXOP (Flutter Free Flight Envelope Expansion for Economical Performance Improvement) and FLIPASED (Flight Phase Adaptive Aero-Servo-Elastic Aircraft Design Methods). The latter has successfully concluded in June 2023 by flying an unmanned 70kg, 7m wingspan conventional configuration vehicle beyond the flutter speed. This was made possible by the active suppression controller registering the symmetrical wing bending and torsional modes in real time and moving the outboard aileron to dampen the vibrations. The current paper presents the key takeaways that are valuable when preparing and conducting a flutter flight test campaign with a similar unmanned aircraft.

The paper is structured in the following manner. First, the demonstrator, its systems, and its flight test environment are presented (section 2). Then, the required upgrades of the telemetry systems are discussed in section 3, followed by the preparations regarding trajectory design, the aeroservoelastic (ASE) model updates and the flutter detector tool description (section 4). A section about planning and conducting the flight tests (section 5) is next. Finally, the selected lessons learned are summarized (section 6).

2 P-FLEX - THE FLUTTER SUPPRESSION DEMONSTRATOR

2.1 Configuration

The P-FLEX demonstrator was an unmanned, conventional configuration, remotely-piloted aircraft with a design takeoff mass of 65kg (Figure 1). It was the upgraded version of the first demonstrator called T-FLEX. The demonstrator featured a $7m$ wingspan, a $20deg$ sweptback wing with an aspect ratio of 20 and a V-tail. The aircraft was powered by a pylon-mounted miniature jet turbine (AeroDesignWorks B300F).

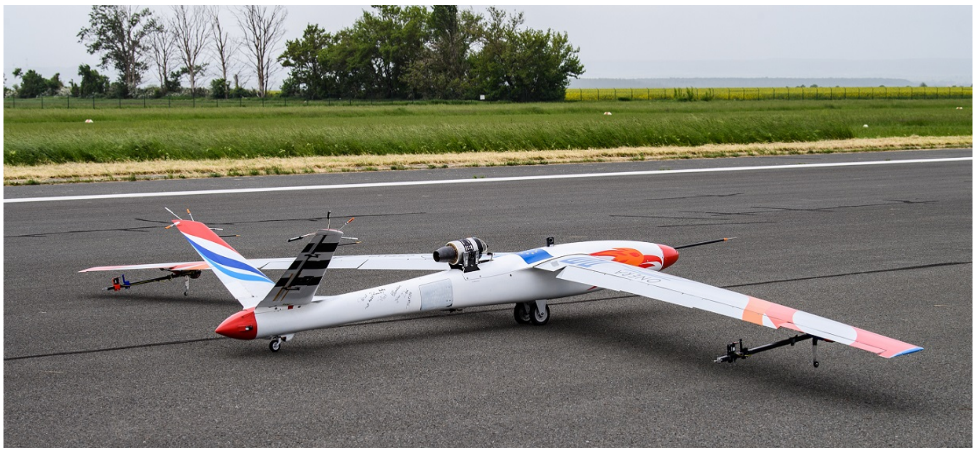

Figure 1: The P-FLEX demonstrator before takeoff. The rods extending backwards from the wing trailing-edge are the flutter tuning rods with the flutter stopper device at the end. Photo by: DLR.

The wing used with the P-FLEX was designed to flutter at a specific airspeed of $48.5m/s$. Based on the GVT results the updated flutter simulations predicted a flutter speed of $56m/s$ by tuning the first symmetric wing bending and wing torsion modes. The tuning was done by adjusting the weight and its position on the flutter tuning rod - a carbon rod facing backward from the wingtip (Figure 2). More details about the wing design can be found in [1].

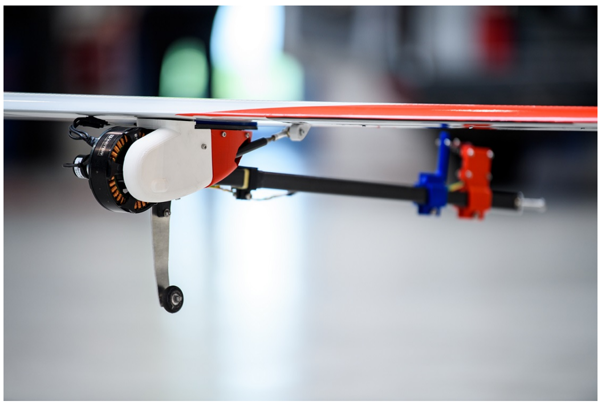

Figure 2: The flutter suppression actuator and the tuning rod on the left wing. Photo by: DLR.

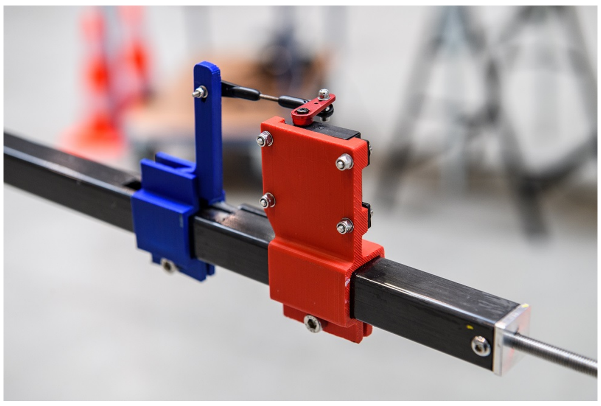

Figure 3: The flutter stopper actuation mechanism. Photo by: DLR.

The pilot flew the T-FLEX manually via external vision with rate control flight mode. The aircraft had two control links. The primary one was a Jeti DS-24 system with an additional backup receiver. The secondary control link was controlled by the backup pilot. Control signals inside the aircraft were distributed via a custom Flight Control Computer (FCC) (see section ref-sec:electronics). The autopilot was used during some tests, but never during take-off or landing. More information about the autopilot design and testing can be found in [2] and [3].

The geometry of the aircraft is summarized in table 1.

Table 1: Geometry of the P-FLEX subscale demonstrator.

Wing span, m:	7.08	Tail projected span, m:	1.42
Wing area, m^2:	2.54	Tail area, m^2:	0.43
Wing aspect ratio:	19.8	Tail aspect ratio:	4.7
Wing incidence, deg:	-0.5	Tail incidence, deg:	-4.0
Wing 0.25c sweep, deg:	19.1	Tail 0.25c sweep, deg:	10.3
Wing taper ratio:	0.50	Tail taper ratio:	0.52
Wing twist, deg:	-2	Tail dihedral, deg:	34.8
Number of wing control surfaces:	8	Number of tail control surfaces:	4
Fuselage length, m:	3.42		
Fuselage maximum height, m:	0.32		
Fuselage maximum width m:	0.30		

As mentioned before, the P-FLEX was the second iteration of the demonstrator. The first iteration, the T-FLEX, had its maiden flight in 2019, but was lost in a crash in August 2023 [4]. After the crash, some upgrades were made for the second demonstrator. The radio control system was improved by upgrading the secondary radio to a Jeti DS-16 NG and adding additional receivers. The wiring scheme was upgraded. An additional, rigid fuel tank was added to the aircraft in the landing gear area, extending the available fuel mass from $7kg$ to $10.3kg$ and, as later was proved in practice, improving the endurance of the aircraft by 72%. An automated fuel transfer system was integrated, which pumped the fuel from the secondary tank to the main tank. Therefore, if the centre of gravity (CG) of the T-FLEX could always be assumed to be at the same location (because the fuel tanks were located very close to the overall CG of the aircraft), the CG shift for the P-FLEX had to be accounted for.

2.2 Onboard electronics and software maintenance

A custom Flight Control Computer, developed by SZTAKI, was used in both demonstrators. The FCC received continuous improvements during the two projects. For example, after the first few flight campaigns it was realized that additional input and output channel handling is required, as well as enhanced computational capabilities. Therefore, a complete redesign of the control relay module called RXMUX (Figure 4) was done. The global chip shortage due to the COVID pandemic posed an additional obstacle and the limited pool of components available in the market had to be made compatible with the already existing hardware.

It was often necessary to modify the actuator mapping and control signals within the RXMUX modules during the flight test campaigns. For redundancy reasons, two of these modules were required to run the aircraft. In the first version of the RXMUX, firmware changes were applied on both modules one by one. After the redesign, a client software that communicated with the modules through an I2C bus was developed, enabling simultaneous reprogramming of both modules. Additionally, the configuration data was separated from the firmware, which allowed for quick configuration changes without needing to modify the embedded software of the modules (Figure 5). This helped to prevent code bugs during configuration modifications.

Figure 6 illustrates the further updates of the RXMUX software.

Additional redesign was done within the aircraft's Remote Control (RC) structure. This involved replacing the previous PPM communication with the digital EX Bus protocol in be-

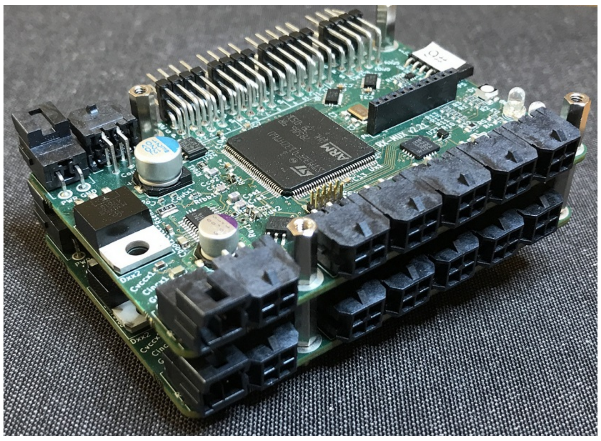

Figure 4: The redesigned RXMUX module.

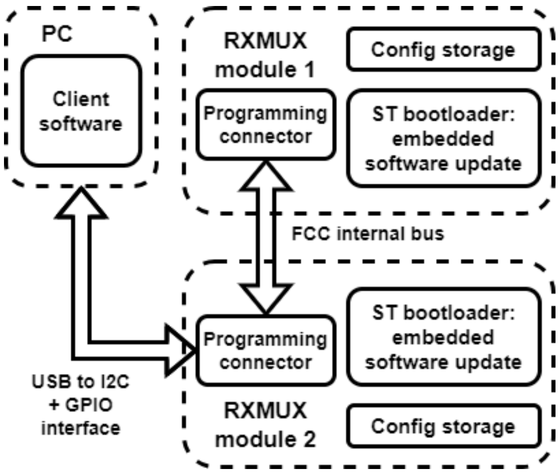

Figure 5: Easy config. update on RXMUX modules.

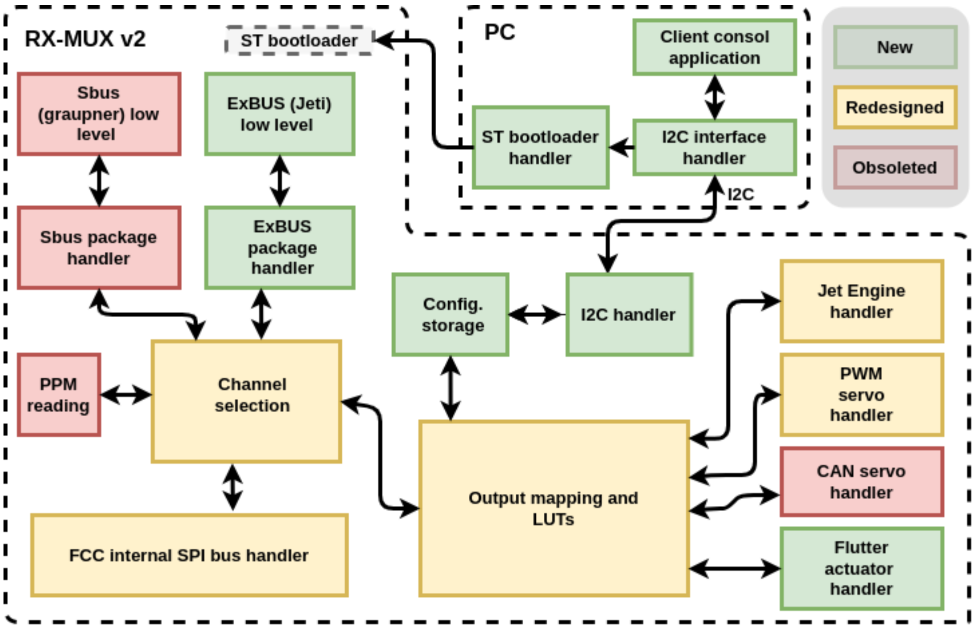

Figure 6: Evolution of the RXMUX hardware and software.

tween the RC receiver and the RXMUX for both primary and backup functions. The revised RC system increased the channel capacity and accommodated the necessary modifications in the aircraft's system layout.

The employed Hardware-in-the-Loop (HIL) test environment was built on a real-time target machine, known as Speedgoat, capable of running the aircraft's model to simulate its behavior and send signals to the FCC. HIL testing proved effective in many cases, but especially in autopilot tests. As the flight plans were developed, the most crucial test points were tested in the HIL environment. Due to the pandemic lockdowns, it was essential that the HIL could be controlled remotely. For this reason, a programmable transmitter emulator was made, meaning that no physical presence was required in the lab to run the simulations. This small add-on proved particularly useful during the flight test campaigns, because then the necessary changes to the autopilot software could be tested remotely before being applied on the flight hardware.

2.3 The flutter actuator and its performance testing

For flutter mitigation, the outboard (4th from the wing root) aileron was utilized. A motor designed for UAV propulsion systems was repurposed along with a high-resolution encoder, as typical actuators were unsuitable for the required frequency and torque. The actuator is visible in Figure 2. To ensure the actuator configuration's suitability for flutter mitigation, a test bench was built, designed to replicate conditions similar to the aerodynamic forces acting on the control surface. It featured two springs attached on both sides of the arm connected to the motor rotor. The distance from the axis of rotation and the spring forces were calibrated to generate the required torque on the motor.

Torque values, appropriate for the loads seen during operation, were selected. The estimated maximum hinge moments were around $0.44Nm$, and a higher load case was chosen to be at $1.04Nm$. Sinusoidal signals close to the flutter frequency, were then injected into the motor, with the maximum deflection set at $10°$. A point-following controller was tuned by analysing the difference between the commanded rotational position and the momentary rotational position of the motor, regardless of the shape and frequency of the signal. The results of the tests on the test bench are presented in table 2.

Max Pos. Error, °	Max Torque, Nm	Signal
0.95	0.44	sin 20 Hz
1.05	0.44	sin 25 Hz
1.68	0.44	sin 30 Hz
0.73	1.05	sin 10 Hz
1.02	1.05	sin 20 Hz
0.89	1.05	sin 25 Hz
1.78	1.05	sin 30 Hz

Table 2: Position tracking performance of the flutter actuator on test bench.

Case	Avg. Abs. Accuracy, °	Max Pos. Error, °
1	0.20	2.18
2	0.22	2.57
3	0.50	2.82
4	0.47	2.77
5	0.45	4.52
6	0.41	3.97
7	0.23	2.57
8	0.18	3.23
9	0.19	3.85
10	0.16	3.09

Table 3: Position tracking performance of the flutter actuator during active flutter control in-flight.

It was decided that the the actuator's performance and the controller's accuracy were satisfying to utilize it for active flutter control tests. The table 3 displays the performance of the flutter actuator during these tests. Each row represents a different case in which the flutter actuator was engaged in performing active flutter control. Figure 7 illustrates the nature of the flutter mitigation signal with a bigger jump.

As seen in table 3, the average accuracy was much below the approximate $\pm 1°$ accuracy, which was seen on the test bench. Some sudden changes indicated higher error values, which occurred a few times during each flutter suppression test. The maximum position errors were observed in these cases.

Proper cooling of the motor was also crucial because the controller's performance decreases with the increase of temperature. On the test bench, the motor was loaded for short periods, so cooling was not necessary. Thankfully, during the controller tests in-flight it was observed that the incoming airflow was enough to provide effective cooling to the actuator, which greatly reduced its temperature in comparison to, for example, the aircraft standing on the taxi-way.

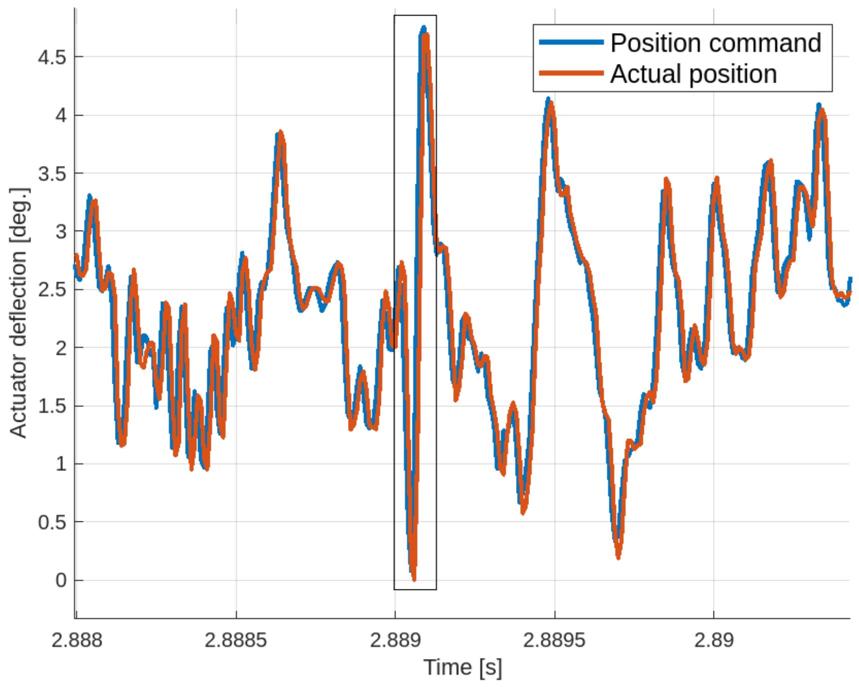

Figure 7: Flutter actuator position during active flutter control with a relatively high jump.

The flutter actuator added additional maintenance requirements to the demonstrator. Due to their high-frequency actuation, the ball-bearing rod-ends of the control linkages for the flutter suppression flaps would quickly develop free-play and had to be changed frequently. Additionally, the weight of the actuator would bend the flexible wing, which was already close to the ground during the taxi or landing. Therefore, additional wing-tip wheels had to be attached to protect the actuator. Often, these wheels would get in contact with the ground during the landing, therefore it was beneficial to make them from aluminum instead of CFRP.

2.4 Flutter stopper

In case the active flutter suppression malfunctions, an additional safety mechanism had to be implemented. Two methods were proposed during the design stage: either a mechanical device that would shift mass within the wing therefore increasing the flutter speed or a preprogrammed maneuver that would quickly dissipate airspeed. It was envisioned that a maximum drag configuration of the aircraft (airbrakes open, landing gear extended, and crow flaps deployed) could be used. However, the second idea was quickly dismissed due to the long spool-down time of the engine and flap deflection limitations at high speed.

Fast and easy to implement design for a mechanical device was proposed. As the tuning masses in the flutter rods were removable, it was decided to develop a spring loaded, controlled release mechanism that would shoot these masses forwards and within $0.5 - 1s$ secure them in the front part of the boom.

In the final version of the mechanism, a spring at the tip of the flutter rod was preloaded on the ground during the start-up procedure of the aircraft (Figure 3). A light servo motor triggered the release mechanism of the spring, which would consequently shoot the mass forwards. The mass was then held in place by a plastic capture clip. Either of the pilots could trigger the flutter stopper from their transmitter. From that point, it took around $0.125s$ to shift the weight, changing the torsional frequency of the wing from 10.7Hz up to 12.8Hz [5]. In such way, flutter would be damped naturally.

The device could only be used once per flight, as there was no mechanism to reload the device in-flight. After it was triggered and the aircraft had landed, the weights had to be pushed back using a metal rod to reload it.

2.5 Flight test environment

Flight tests were performed at DLR's National Experimental Test Center for Unmanned Aircraft Systems at Cochstedt Airport (EDBC). For the final, flutter test campaign, two circular flight boxes above a sparsely-populated area were defined (Figure 8). The flight boxes were centered on the two taxiways from which the flight test team operated and where the pilots would be. The dimensions of the boxes were limited by the maximum VLOS range of $1500m$ (which is the outer border of the contingency zone) and the size of the ground risk buffer.

Ground risk buffer size was mainly determined by the wind drift that would result if the parachute would open at the edge of the contingency zone, at the contingency altitude (in this case, the flight geography altitude plus $158m$) and at the maximum operating mass and airspeed. It was calculated using a custom software for two flight altitudes (with contingency altitudes in brackets): the minimum flight altitude of $H_{FG} = 150m$ ($H_{CV} = 308m$) and the maximum flight altitude of $H_{FG} = 300m$ ($H_{CV} = 458m$). The software was adjusted with the characteristics of the installed parachute. The simulations were performed for various horizontal wind conditions from 0 to $7m/s$. The resulting drift trajectories are tabulated in table 4.

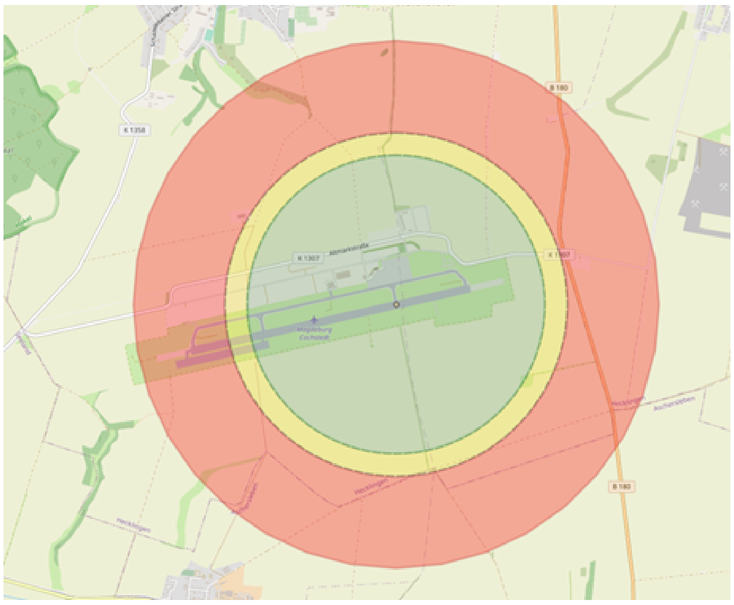

Figure 8: One of the flight boxes for the flutter campaign. Green - flight Geography, yellow - contingency zone, red - ground risk buffer.

Table 4: Drift distance after parachute release from two contingency volume heights for different wind conditions.

Wind speed, m/s	$H_{CV} = 308m$	$H_{CV} = 458m$
0	155m	155m
2	267m	326m
4	378m	497m
6	491m	669m
7	546m	754m

As a result of the simulations, wind limits had to be applied to one of the flight boxes, as otherwise the ground risk buffer would overlap with an edge of the village, not complying with the flight permit regulations. In the end, to achieve the biggest possible flight box for the flutter test flights, the borders of the ground risk buffer had to extend up to $2254m$ away from the pilot.

At the time of writing, the parameters for the flight geography can also be retrieved by using the online tool provided by the German Federal Ministry for Digital and Transport [6].

The flight test operations were continuously improved based on the previous experiences [4, 7]. Flight test crew consisted of 5 people (two pilots, an operator, an engineer and a manager). By the end of the project, 13 different people had to be trained to take up different roles at different times improving flight test team's flexibility and redundancy.

3 ADVANCES IN TELEMETRY SYSTEMS

3.1 Initial Telemetry Setup

In the T-FLEX demonstrator, three 433Mhz telemetry channels were used for monitoring and control functionalities. The first two channels were part of the original system design, and the third channel was added during the last two flight test campaigns. Each telemetry unit used a dipole antenna pair. Each channel used a different application for data reception and visualization.

The first channel was dedicated to mission control via standard MavLink protocol. It used Mission Planner (MP) with an extended user interface for autopilot control. The operator used the monitoring interface for guidance, where the actual position, orientation, altitude, airspeed and back-angle of the aircraft are displayed. An extended interface was used by the engineer to interact with the autopilot of the system.

The second channel was used to status and health monitoring of the system via a custom, in-house developed Engineering Data Link (EDL) protocol. It used a static display created in Matlab. The display showed status of the batteries, autopilot and control information about the FCC, readiness status of the custom IMUs, servo positions errors and temperatures, engine and ECU status, airspeed and the aircraft load factor.

The third channel was used to support Online Mode Analysis (OMA) application [8]. It used OpenMCT to display the predicted trends over time of certain structural bending modes of the aircraft.

Figure 9 shows the Ground Control Station with the different telemetry views. The first screen was used indicate the approximate end of flight. The second screen was used by the operator as navigation assistance for the pilots. Screen four showed live data from the OMA tool. Screen five was used to control the autopilot functionality via a custom MP interface. Screen six showed the static EDL screen.

The telemetry system had a data rate limitation. The inherent radiation characteristics of the used dipole antennas resulted in dead spots within the flight-box, where one of the telemetry channels had no reception. The available space inside the payload area of the T-FLEX meant the three receivers were in close proximity of one another causing interference. That interference caused reduced data rate or reception loss depending on the orientation of the system. The available bandwidth and transmitted data allowed only relatively low update frequency on each channel.

The EDL had $\sim$ 3Hz average data rate on all the displayed data, but that varied drastically based on the quality of the datalink. The MP had different data rates for different telemetry points.

3.2 Updated telemetry systems

For the P-FLEX demonstrator, the telemetry systems were updated. The updates are described in the following subsections.

3.2.1 Radio modules

The 433MHz modules and their dipole antennas were replaced due to their orientation limitation and maximum achievable data rate. The updated telemetry system used RFD 868Mhz telemetry modules with 2 dipole antennas for each module. The 433MHz and 868MHz system were field tested in a mock-up configuration using a Px4 autopilot system. During that test, the 868Mhz system outperformed the 433Mhz system in data rate, distance and orientation as well.

To reduce interference between the two telemetry modules, the EDL module was placed in the payload area, and the mission control module was placed under the belly of the aircraft, behind the landing gear.

After the updates, the data rate on the EDL link was increased to $\sim 8-10$ Hz average data rate on all values. The data rate for the MP link didn't change due to hard-coded time values.

3.2.2 Mission Planner

Mavlink parameters were used to control the autopilot functionalities in-flight. To make this control more user-friendly during the flight, a custom GUI was developed. The last iteration of the autopilot interface GUI can be seen in figure 10.

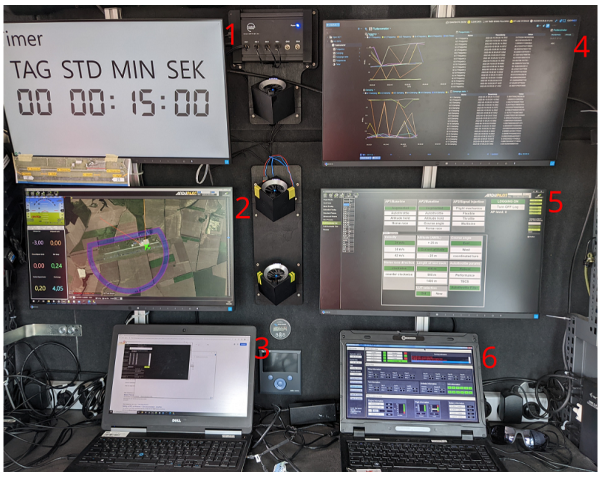

Figure 9: Different telemetry views used for T-FLEX.

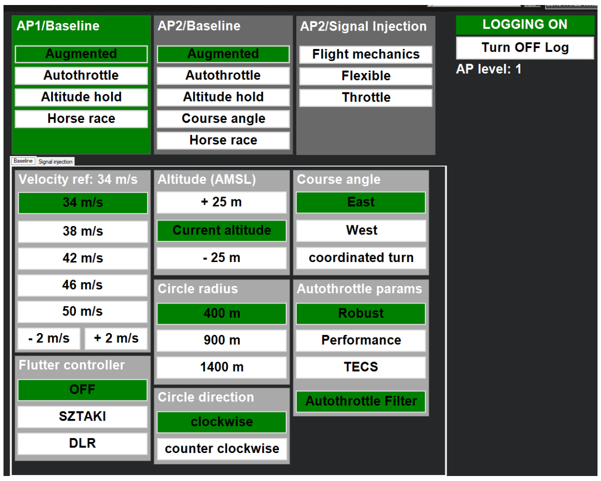

Figure 10: Mission Planner autopilot interface.

There are three main parts of the user interface. First one is the three function enabling panels at the top with AP1/AP2/Baseline/Signal Injection labels, that correspond to the three different autopilot switch positions on the transmitter. The second one is the logging on/off switch and an AP level indicator. The third one is the large tab control panel with baseline and signal injection tabs. The background of the panels show which autopilot state is active (manual / autopilot 1 / autopilot 2). The buttons on the baseline panels can toggle the appropriate autopilot function. For example in the figure, augmented mode is enabled in both AP states and the AP1 mode is enabled.

3.2.3 OpenMCT

OpenMCT (Open Source Mission Control Software) [9] is an open source telemetry software developed by NASA. It provides an easily customizable GUI running in a web browser and relies on Node.js as its back-end. The configuration used during this project is based upon an Open MCT workshop held by M. Weber [10].

The dynamic properties of OpenMCT allowed the visualization of system's information on 2D graphs withing a predefined sliding window. Along with that, the test-team was able to reconfigure the OpenMCT display between flights, thus preparing mission specific visualization along with the health monitoring functionalities. Moreover, since EDL and OMA messages were coming through the same radio module, the test team was able to put both EDL and OMA specific graphs on the same telemetry screen.

Figure 11 shows the GCS view after the updates. Screen one shows a live video stream of the aircraft using a GPS based object tracking system. Screen two shows the view used for navigation by the operator. Screen three shows status information about the aircraft along with parameters used for fast feedback during different mission segments like system identification or flutter testing. Screen four shows the autopilot control panel.

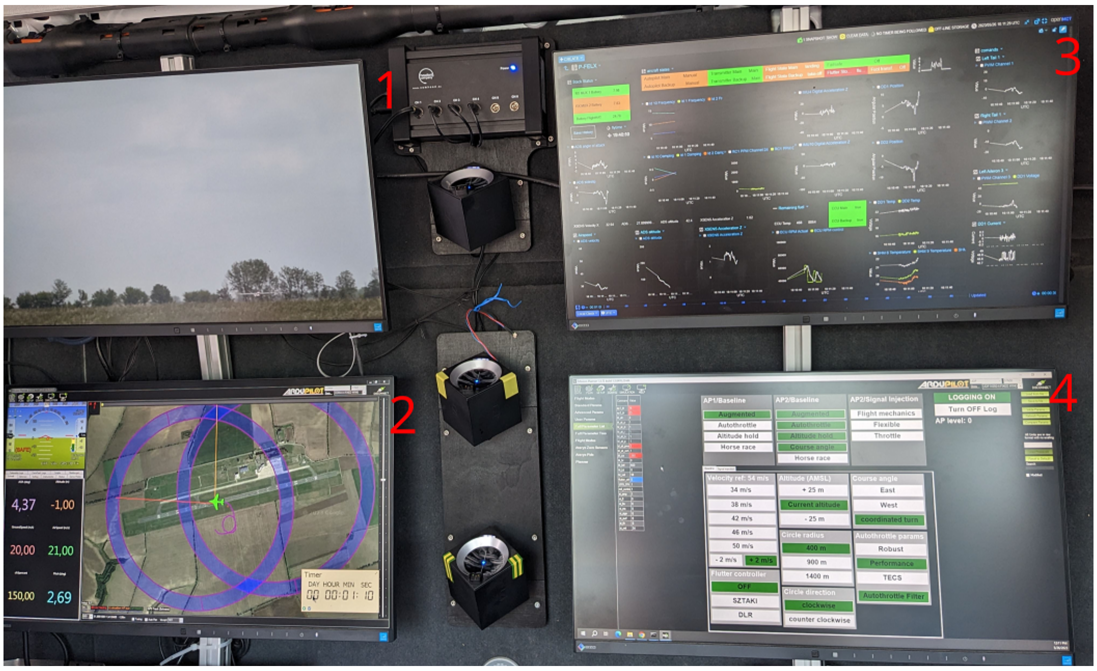

Figure 11: GCS with all the telemetry screens.

4 PREPARATIONS FOR THE FLUTTER SUPPRESSION FLIGHT TESTS

4.1 Flight trajectory design

When analysing aeroelastic systems, current state of the art techniques rely on linear time invariance for identifying system parameters and ultimately critical damping ratios for flutter prediction. The flight envelope expansion proceeds with discrete test points as shown in figure 12. The aircraft is flown at constant speed and altitude for a sufficiently long duration (usually 20+ minutes) until the dynamic behaviour is identified and deemed to be safe. The aircraft then either lands or proceeds to the next test point.

The current methods developed by DLR [12], have proposed real time in-flight modal parameter

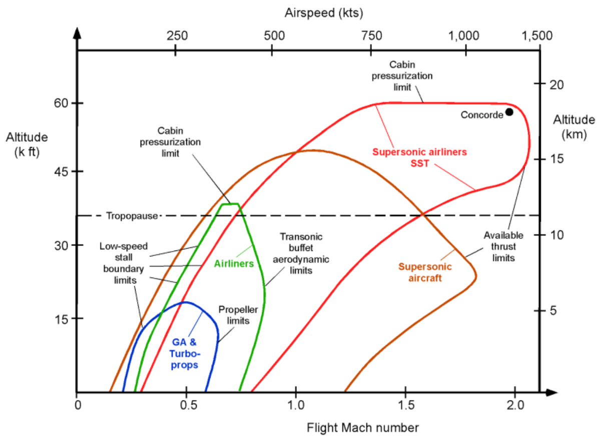

Figure 12: Flight envelope expansion for different aircraft in terms of their achievable altitudes and airspeeds [11].

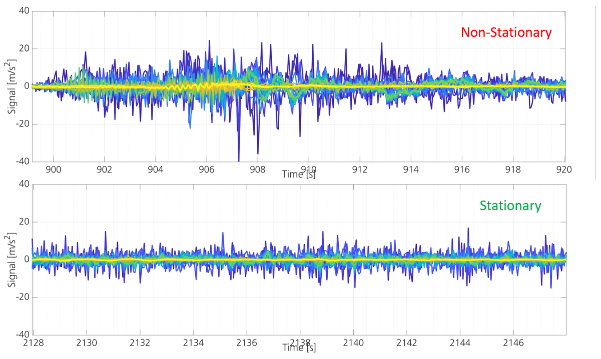

Figure 13: Non-stationary and stationary flight vibration signals.

identification with fast and efficient algorithms. The challenge of the FLIPASED project was to determine how long was 'long enough' for the system to be considered time invariant and the signals stationary.

The initial trajectory for acquiring this stationary test point in a confined flight box (see Section 2.5) was a horse race track pattern. During the pattern, the time for a single straight stationary test point was limited to approx. 8 - 20 seconds. The aircraft then needed to break and turn before flying the next part of the horse race track. Therefore, the trajectory was changed to a constant bank angle turn that closely followed the flight geography border. Flown by an autopilot, the aircraft performed a simple constant radius turn. When necessary, the pilot would manually correct for the wind drift as commanded by the Flight Test Operator, who followed the trajectories and the flight box borders during the flight tests.

As shown in figure 13, the horse race track flights resulted in signals as shown on top, while the constant bank angle circles resulted in signals as shown at the bottom. Both plots have the same Y axes scaling. Here it can be seen that the constant bank angle circles can be considered far more stationary than the horse race track pattern. Furthermore, the constant bank angle circles increased the time on 'stationary' test point. During the flight test campaigns it was found that by optimizing the Stochastic Subspace Identification SSI algorithm accurate modal parameters could be identified and tracked with data buffer lengths of 30 seconds. A sliding overlap window provided new estimates every 2 seconds from the online monitoring system developed by DLR [8], running on the secondary onboard computer.

4.2 Flutter detector module

In order to safely proceed with the flutter tests, an online flutter detection algorithm was required. This would allow the test team to confirm the functioning of the flutter suppression algorithms in-flight, and make "go/no-go" decisions. The initial plan was to implement the *flutterometer* proposed by Lind [13]. This was quickly abandoned due to real-time, on-board implementation aspects. Two further approaches were pursued, an operational modal analysis algorithm as mentioned in section 4.1, and a rapid, filtering based approach described below.

The online, real-time flutter detection algorithm is tuned for the anticipated flutter frequency of 8.75 Hz, with a reaction time of less than 1 sec. The main algorithmic components are shown in figure 14.

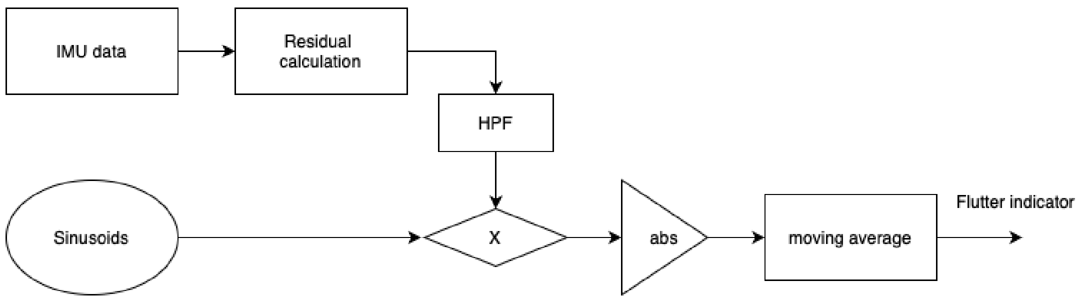

Figure 14: Flutter detector block diagram.

The algorithm has two main lines, focusing on detecting the occurrence of symmetric and asymmetric oscillatory motion of the wings, using the built-in inertial sensors of the wing and the fuselage.

The logical flow of the detector is the following. Angular roll and yaw rates at the CG (p, r) and wing torsional rate from the wingtips $(\omega^6_{y,L}, \omega^6_{y,R})$ are used. Two residuals, similar to the measurement used by the flutter controller [14] corresponding to the symmetric and asymmetric wingtip relative torsional rotation are formed, taking into account the wing sweepback angle (17.6 degrees at the location of the IMUs):

$$r_{sym} = [0, -2 \cdot \cos(17.6/180 \cdot \pi), 1, 1] \cdot [p, r, \omega^6_{y,L}, \omega^6_{y,R}]^T \tag{1}$$

$$r_{asym} = [-2 \cdot \sin(17.6/180 \cdot \pi), 0, 1, -1] \cdot [p, r, \omega^6_{y,L}, \omega^6_{y,R}]^T \tag{2}$$

Residuals (r_{sym}, r_{asym}) are high-pass filtered first to eliminate offsets and biases, with band-pass frequency of 0.1 rad/s. The high-passed residuals are multiplied by $1 + 12$ sinusoidal signals of 8.75 Hz frequency, starting form the actual to increasing number of delays up to 12 delays with sampling time of $t_s = 1/200$ sec. This corresponds to 0 to 189 degrees phase shift of these generated sinusoidal signals, at an anticipated flutter time constant of $1/f_f = 0.1143$ sec. GNSS receiver tracking loops use similar techniques to find the larges correlation between a self generated replica signal and the measured one. There is no need to cover the entire 360 degrees of phase shifts, since the replica sinusoidal signals and the residual are multiplied and the absolute value is taken.

In the next step to form a flutter indicator signal a moving average of 100 samples, equal to 0.5 sec, is taken. Lastly a maximum vale of the 13 concurrent signals, corresponding to the highest correlation phase shift, is taken and this signal from both the symmetric and the asymmetric channels are sent via telemetry to the engineering display of the GCS where they are monitored together with a longer latency operational modal analysis results.

Some results of using the detector are presented in section 6

4.3 Updating the flutter speed predictions

In consideration of the upcoming critical flutter tests, it was essential to ascertain the precision and reliability of the flutter speed predictions derived from updated theoretical model. The structural dynamics modal model was updated by using the eigenfrequencies and damping values that were identified in the GVT. Further, the GVT-identified modes shapes were taken and mapped to the structural grids of the condensed model. By employing this approach, it is possible to update the structural modal model, which is used directly in the flutter calculations and controller design, without requiring any modification to the full FE model [15].

The updated modal data was then incorporated into the p-method and pk-method, two classical flutter methods. By approximating the unsteady aerodynamics via rational functions, the p-method determines the eigenvalues of the system matrix as a function of flight speed. The eigenvalue problem in the frequency domain can be solved using the pk-method, which involves iteratively adjusting the interpolated complex valued generalized aerodynamic forces to achieve the desired reduced frequency.

The GVT-updated model showed remarkable behavior, as the antisymmetric flutter mechanism was eliminated and the symmetric flutter delayed to about 55 to $56m/s$. Scaling the mode shape to 80% of its identified magnitude was employed to modify the modal mass of the symmetric torsion mode in order to assess the robustness of these results against potential inaccuracies in the identified modal mass. The estimated $55.5m/s$ flutter velocity was confirmed. Additional independent computations by different partners validated the flutter velocity of approximately $55m/s$ to $56m/s$.

Using data from flight tests of the fixed-wing P-FLEX UAV, a significant number of flight test data have been examined for post-flight system identification with respect to the determination of flutter boundary. For system identification an automatically running robust Stochastic Subspace Identification (rSSI) method is used, which only needs the structural dynamic response data of the aircraft due to non-deterministic natural and/or operational excitations which are provided by atmospheric turbulence and/or pilot control inputs. The outputs of the method are the stochastic system matrices and consequently the flight modal parameters.

The results of the updated and verified model are discussed in section 5.3.

5 BUILD UP AND CONDUCTING THE FLUTTER TESTS

This goal of this section is to describe the planning of the test campaigns and how did those plans develop as the campaigns progressed in the last year of the FLIPASED project, 2023. To better understand the process and logic behind the planning, it is important to start by explaining the status quo at the beginning of that year.

The previous demonstrator that was used by the project (the T-FLEX), crashed in August 2022. At that point, the geometrically identical demonstrator was flying with a rigid, non-fluttering wing. Its system bugs were identified and either fixed. The autopilot software was able to keep the demonstrator flying in a predefined course, but further tuning of the airspeed-tracking module was necessary.

With the crash of the T-FLEX demonstrator, some minor design changes were done to the second iteration, the P-FLEX. Even without the changes, the flight testing had to start from early stages of the program to gain trust and solve the inevitable system problems in the second demonstrator. This program was developed building onto the previous experience with the T-FLEX and is described below.

The details of the accident with the first demonstrator were described by Bartasevicius [4].

5.1 The initial plan

When preparing the flight test program for the final year of the project, the following had to be considered:

- Even though the wing flutter speed in a flutter-safe configuration (with the flutter weight in the front of the tuning rod) is higher than the usual flight speed, extra care has to be taken not to excite the flutter unintentionally.
- The rebuilt demonstrator will only pass the ground vibration testing end of February.
- Uncertainty remains about the timeline for receiving the flight permit from the German Civil Aviation authority (LBA).
- Due to the cold weather temperatures, no flying before March should be planned.
- As the project ends in end of June, the flight tests should finish in May to allow for some final reporting to be done.
- The demonstrator has new systems on board, that were only tested on the ground. These systems will have to be proven functional in flight:
  - The fuel shift system,
  - The rebuilt and upgraded flight control computer,
  - The flutter stopper mechanism,
  - A new secondary transmitter.
  - The high-speed flutter actuator,

With this and the overall project targets in mind, the separate goals were noted down. It was decided to split the goals among two two-week-long flight test campaigns. Based on previous experience, two weeks was an optimum time between the logistics required to get the flight team and equipment to the test site and the time left for testing and solving the problems.

The first one was planned for March and the second one would be planned for May.

5.2 Conducting the tests

The goals mentioned above are displayed in Figure 15. In the figure, each flight is colored according to the goals for the flight (see the colorbars next to the goals), and differences between the planned and actual flight test sequences can be identified.

Only had two flights performed during the first campaign. Even though 9 potential flight days were planned for that campaign, the bad weather has disturbed the schedule heavily. Due to snow and wind the complete first week was unflyable.

The second week has again started with wind above the allowable limits, and only the last two days had flyable weather, where two flight test were conducted. Nevertheless, the maiden flight was performed and majority of the system checks were finished. Also, some of the autopilot functionalities were tested.

For the second campaign, a rest week was planned in between the two flight test weeks. The rest week could be used either for analysis of the interim results, for necessary upgrades or for repairs. Again, high crosswinds and rain has disturbed the first week of the campaign. Even so, five flights were made, covering majority of flight dynamic and performance checks and autopilot tests. The first week has finished with the flight test 30.

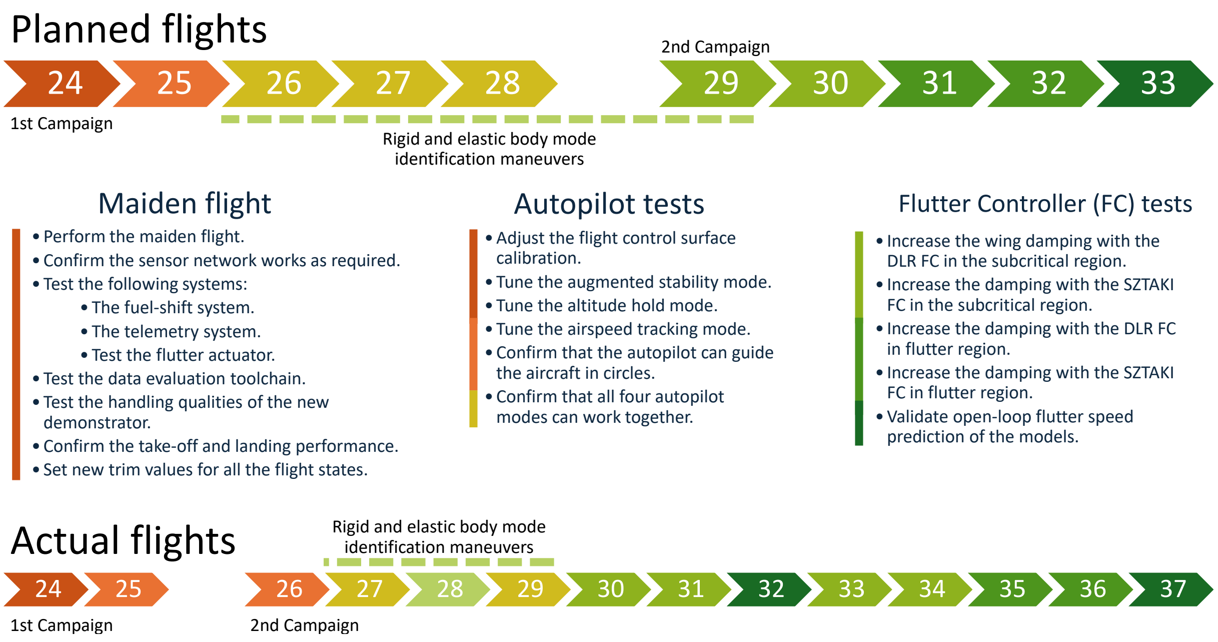
Planned flights
2nd Campaign
24
25
26
27
28
29
30
31
32
33
1st Campaign
Rigid and elastic body mode identification maneuvers
Maiden flight
• Perform the maiden flight.
• Confirm the sensor network works as required.
• Test the following systems:
• The fuel-shift system.
• The telemetry system.
• Test the flutter actuator.
• Test the data evaluation toolchain.
• Test the handling qualities of the new demonstrator.
• Confirm the take-off and landing performance.
• Set new trim values for all the flight states.
Autopilot tests
• Adjust the flight control surface calibration.
• Tune the augmented stability mode.
• Tune the altitude hold mode.
• Tune the airspeed tracking mode.
• Confirm that the autopilot can guide the aircraft in circles.
• Confirm that all four autopilot modes can work together.
Flutter Controller (FC) tests
• Increase the wing damping with the DLR FC in the subcritical region.
• Increase the damping with the SZTAKI FC in the subcritical region.
• Increase the damping with the DLR FC in flutter region.
• Increase the damping with the SZTAKI FC in flutter region.
• Validate open-loop flutter speed prediction of the models.
Actual flights
Rigid and elastic body mode identification maneuvers
24
25
26
27
28
29
30
31
32
33
34
35
36
37
1st Campaign
2nd Campaign

Figure 15: The planned and actual flight tests. The color coding of the separate flight corresponds to the main goals of that flight, as identified by the colorbars next to the goals. The difference between the two test campaigns is shown by the separation in the flight sequence.

5.3 Flutter controller tests

After the rest week, additional flights were scheduled to confirm the open-loop flutter speed of $56m/s$. Steady circling test points without the flutter controller were gradually done from $48m/s$ up to $54m/s$ during flight test 32. At that airspeed, steady oscillations of the wing were visible. Identified and predicted flutter speeds derived from the updated model were then compared (Figure 16). After a good match between the two datasets was confirmed, the flutter controller tests could continue.

The following build-up sequence was proposed to test the functionality of the controller in the sub-critical region:

1. Test for controller saturation by engaging the flutter suppression controller for a duration of $5s$ in a straight flight together with the other autopilot modes. Evaluate the aircraft response.
2. With the controller engaged, perform small control manoeuvres during a straight flight.
3. If the first two points are passed, fly two automated circles in the sub-critical region (starting at $42m/s$) with the controller on and then one circle with the controller off for comparison.
4. Increase the airspeed command by $2m/s$ until $48m/s$ below the critical flutter speed and repeat the previous step.

Both SZTAKI and DLR controllers have passed the above sequences, and the testing moved onto the critical flutter region. The testing was done in the following manner:

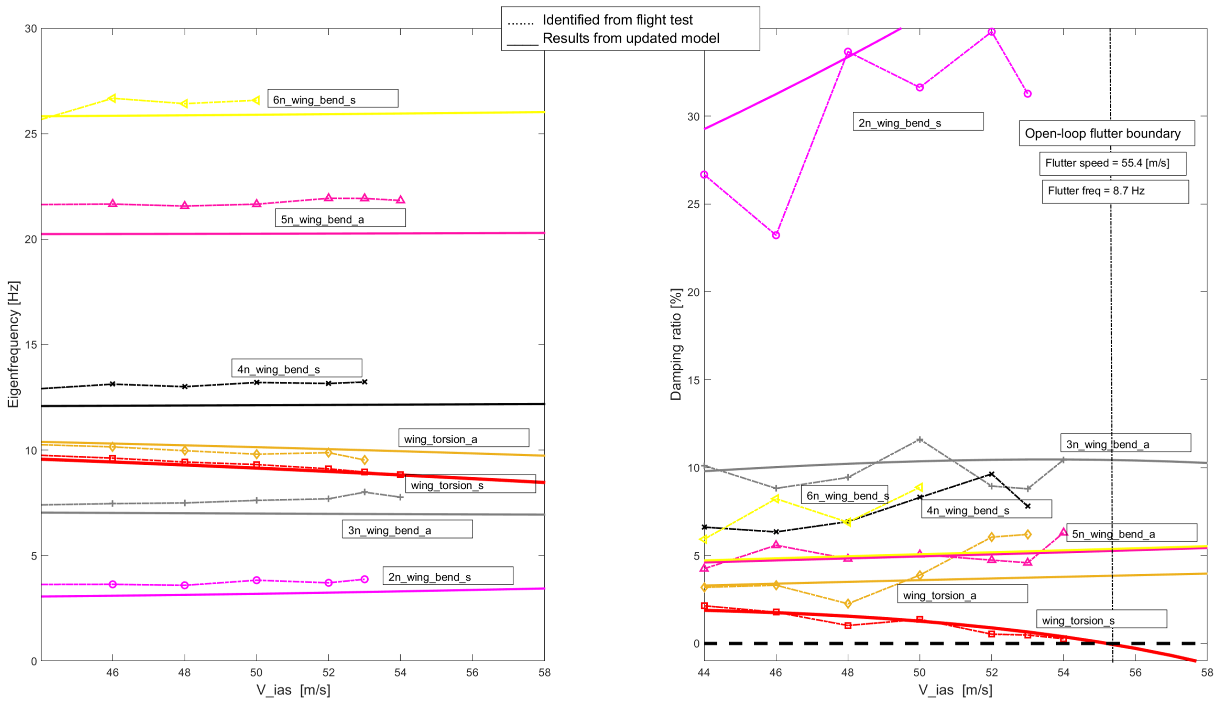

Figure 16: Comparison of flight modal parameters obtained from updated ASE model and identified from FT32 (open-loop)

1. Starting at $50m/s$, perform circle test points and increase the airspeed by $2m/s$.
2. At $54m/s$, for setting the baseline, disengage the flutter controller and perform a circle without it.
3. After, keep increasing the airspeed by $1m/s$ with the flutter controller engaged until $59m/s$.
4. If successful, switch to a straight flight and perform the following sequence: flutter controller off $\rightarrow$ wait 3 seconds $\rightarrow$ engage the flutter stopper. This sequence was intended to evaluate the flutter development characteristics beyond the first critical speed and was trained by the crew on the ground.

This procedure was followed with both controllers. During the flight the ground crew followed the telemetry and evaluated for any signs of flutter.

Figure 17 shows the flutter indicator values during a critical flight test (FT35), during which the DLR flutter controller was being tested. The symmetric indicator signal exceeds 1.5 for a short time during a test leg with $54m/s$ when the flutter controller is switched off. Another short duration exceeding 1.5 is when windgust pushes the demonstrator above $60m/s$ for a short time with flutter controller on. It is clearly noticeable that the symmetric indicator takes larger values, while the asymmetric indicator rarely goes beyond 1.0. The overall trend of the signal peaks between 1400 and 1800 sec also correlates well with increasing velocity, reaching the closed-loop limit of flutter.

Figure 18 shows yet another critical flight test (FT36). The flutter indicator values during the performance validation of the SZTAKI flutter controller do not go beyond 1.5. The value is

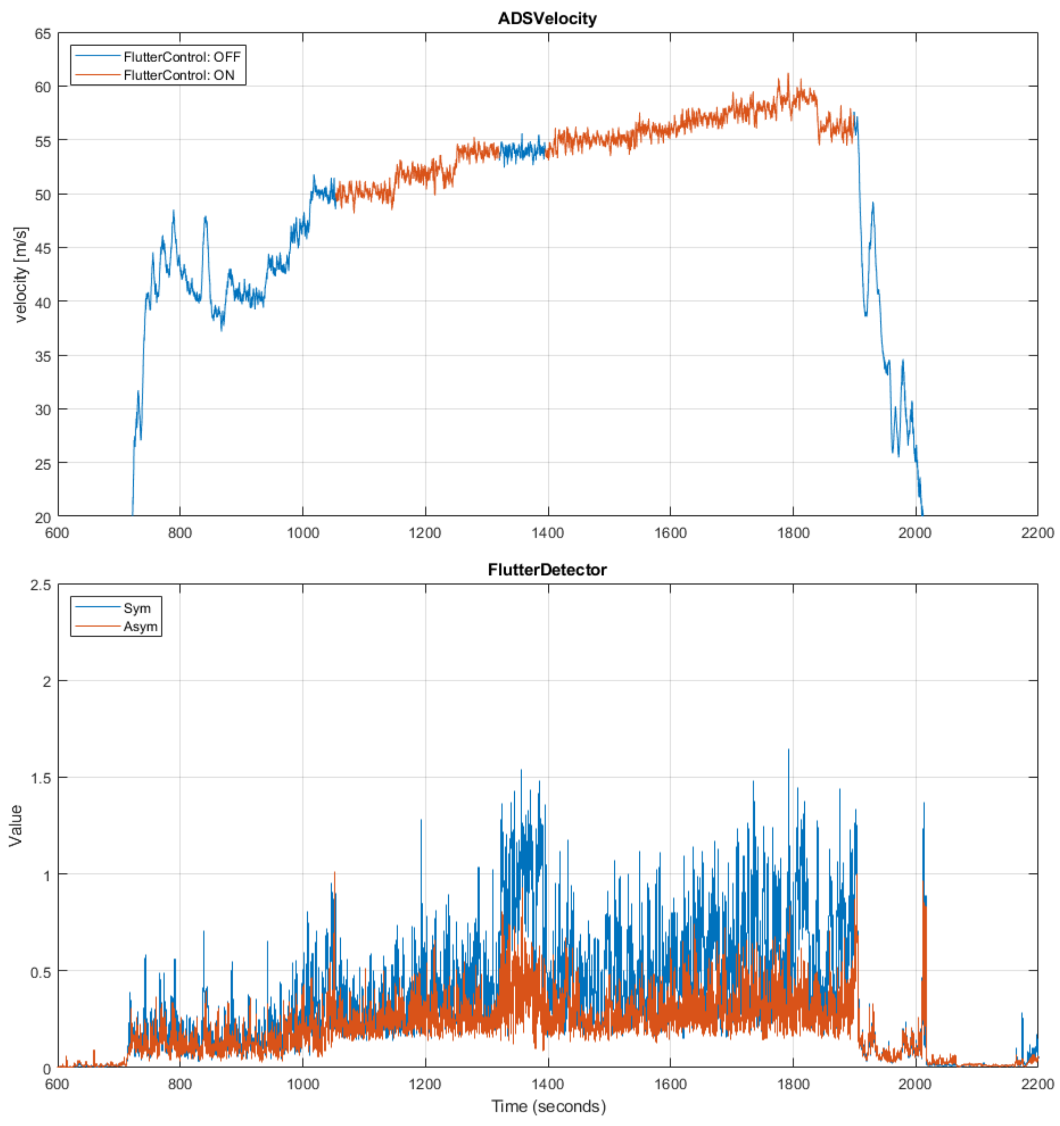

Figure 17: Flutter detector signal in FT35 (DLR Flutter Control test).

only reached in the last part of the flight test, when the flutter controller is disengaged at $58m/s$, before the flutter stopper is engaged. The open-loop part between $1250 - 1350$ sec also shows higher values, clearly indicating that the flutter controller has significant impact on the torsional behavior of the wing.

Since both FT35 and FT36 were performed on the same day with the same demonstrator configuration the only difference between the flutter controller off parts could be due to atmospheric conditions. This means more excitation were present during FT35. The increasing flutter indicator trend with velocity increase using the DLR controller is not so visible with the SZTAKI controller. This might be due to the fact that the indicator and the performance objective of the SZTAKI controller are similar, hence in this metric the SZTAKI controller performs better. Based on these flights a sensible engineering decision would be to set a threshold of $1.5 - 1.6$ for automatic flutter warning. This signal might be used in the future to trigger the flutter stopper, to automatically eliminate the onset of flutter.

The flutter boundary expansion enabled by the DLR AFS controller of the closed-loop system has been verified via post-flight modal identification. Here the flight modal damping parameters

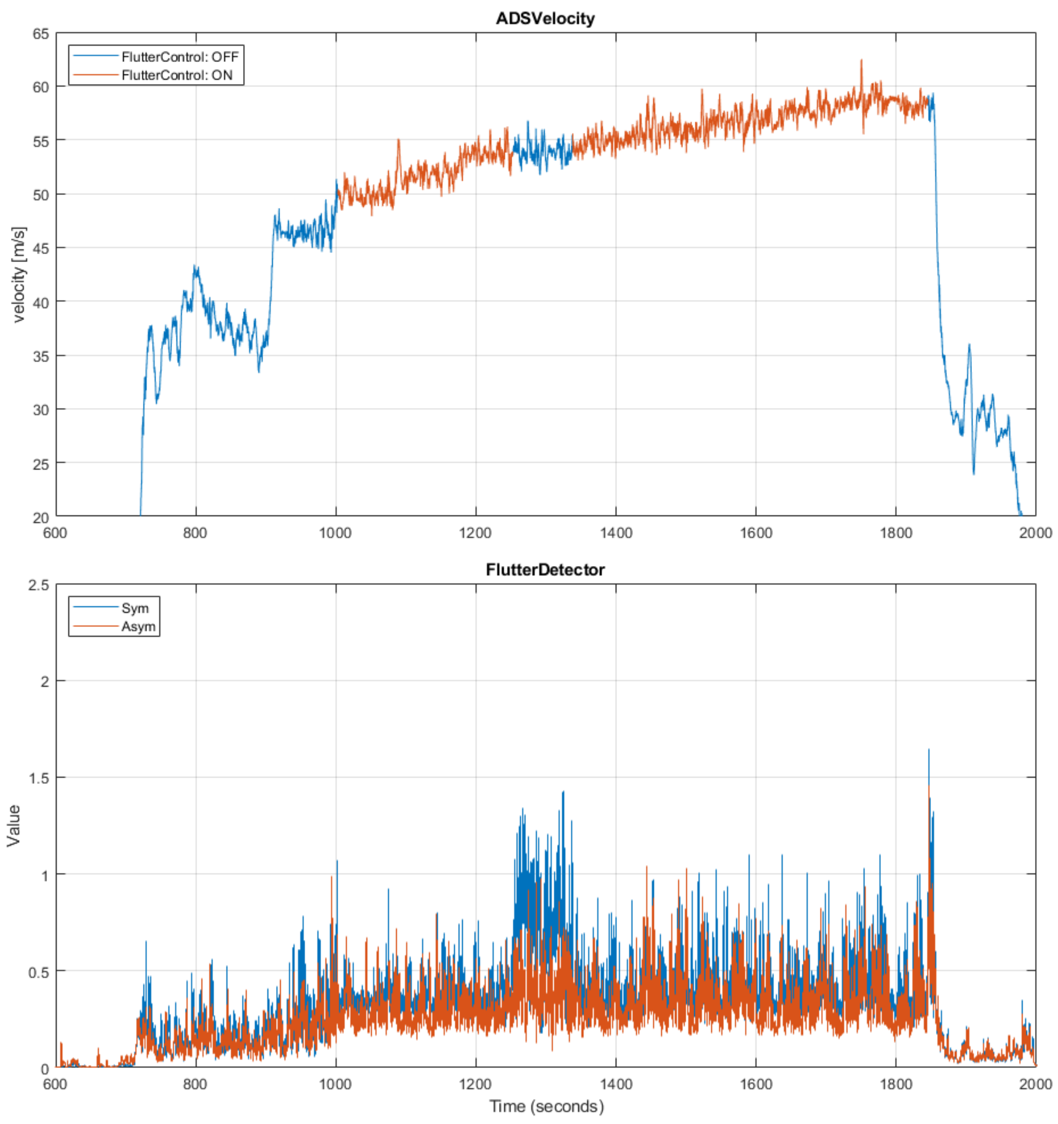

Figure 18: Flutter detector signal during FT36 (SZTAKI Flutter Control test).

identified from FT32 (open-loop) and those extracted from FT35 (closed-loop with AFS controller) are compared (Figure 19). It is evident that activating the AFS controller increases the damping of the critical flutter mode (1st wing torsion), thereby allowing the aircraft to operate at speeds beyond its open-loop flutter speed.

5.4 Confirming the open-loop flutter speed

The final flight of the project was dedicated to confirm the flutter speed by experiment. The aircraft was to be accelerated to $54m/s$, the last open-loop test point, and then the airspeed would be increased by $1m/s$ until flutter could be observed by the crew.

The flight took place in high turbulence conditions and wind speeds of $5m/s$. Nothing unexpected was noticed when the aircraft accelerated to $54m/s$, so the airspeed was gradually increased. Surprisingly, the aircraft reached $59m/s$, way beyond the expected flutter speed. Then, the airspeed was reduced to $56m/s$, the flutter controller was turned on and the aircraft was again accelerated up to $61m/s$.

At that point it was noticed that the right flutter rod is hanging loose from the wing and that

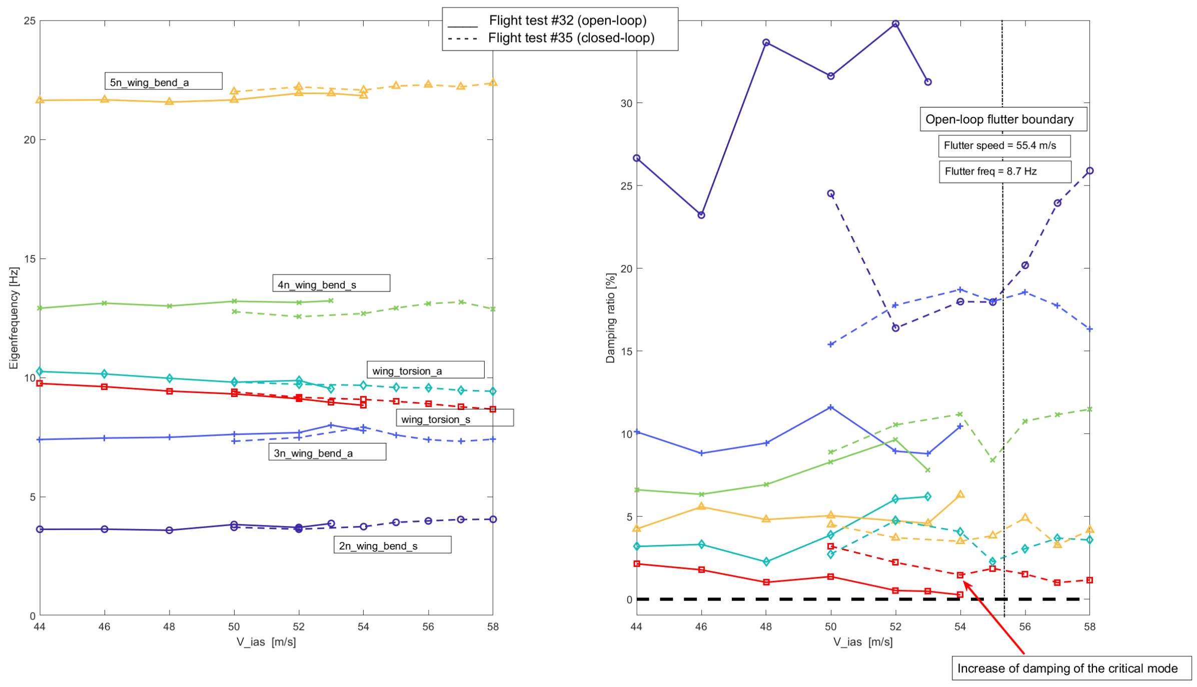

Figure 19: Comparison of flight modal parameters identified from FT32 (open-loop) and FT35 (closed-loop)

the left one is missing completely (Figure 20). The flight was aborted and the pilots went for landing.

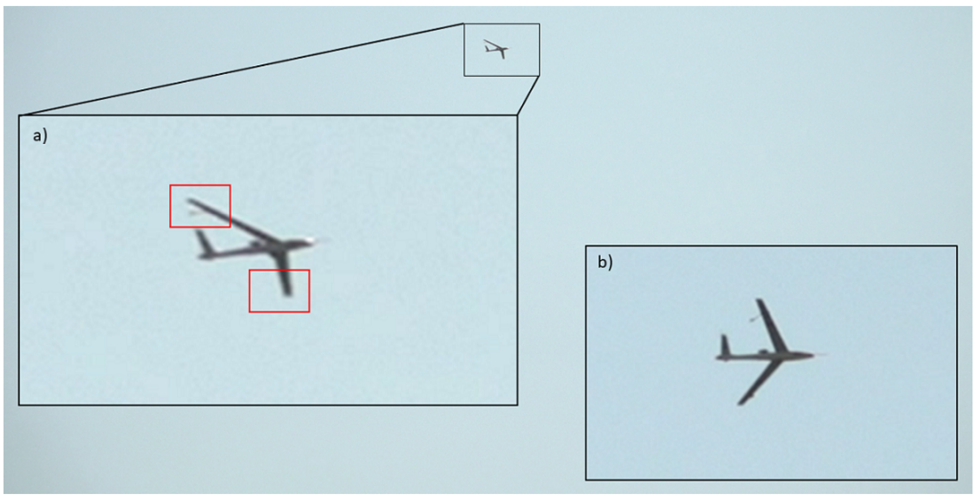

Figure 20: The zoomed in view from the tracker camera. The inlet a) shows the left flutter rod missing after 6 minutes of flight. Inlet b) shows the right rod hanging around 14 minutes into the flight.

During the analysis of the onboard video data it became clear that the flutter rod mount got damaged after the aircraft reached $54m/s$. At that moment the aircraft hit a gust which induced significant vibrations (Figure 21, around 13:49:14), and the airspeed reached $56.1m/s$. The vibrations were dampened out the first time, but the second time they increased, and as the airspeed reached $55.9m/s$, the flutter rod mount got damaged (Figure 21, around 13:49:26). The right rod was still rigidly mounted on the wing. In this asymmetric configuration, the aircraft could reach airspeed beyond the calculated flutter speed, with the loosely connected left flutter rod acting as a damper. In another two minutes, after another big gust was encountered, the right flutter rod got damaged and the left one broke off completely.

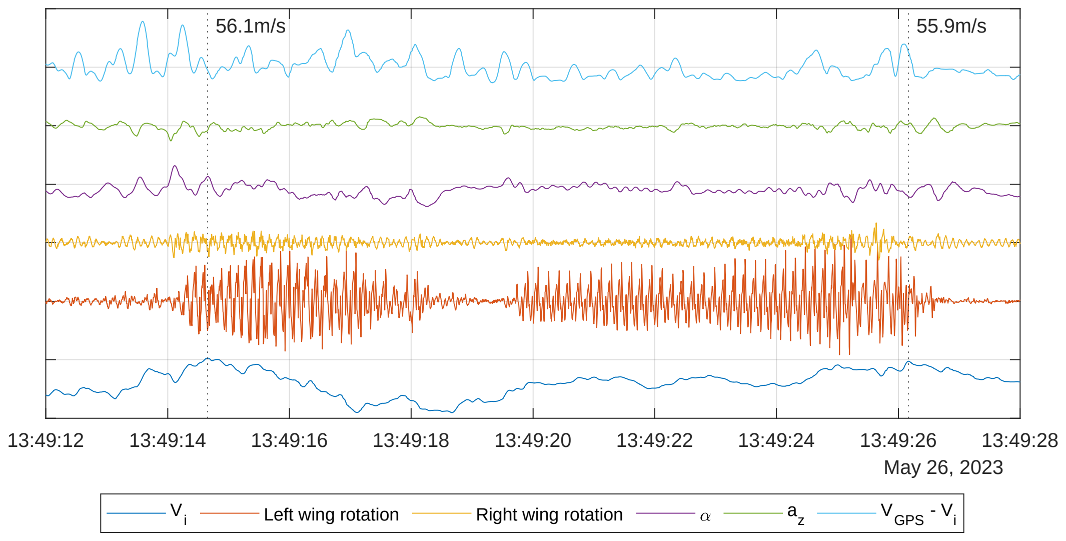

Figure 21: The 16 seconds of highest oscillations of the left wing. All variables here are normalized and offset vertically to each other for better visibility.

The flutter detector signal for the flight is presented in Figure 22. A value over 2.4 is visible on the symmetric flutter indicator at 750 sec and the asymmetric indicator also goes beyond 2.0 at the event when the flutter tuning rod breaks off from the wing. In the subsequent part until 1080 sec high peaks, reaching 1.0 are still visible with one single flutter rod on the right wing. After it breaks off, the flutter detector values even beyond $60m/s$ do not go beyond 0.3.

The question was raised why did the same not happen on flight test 32, where the aircraft flew at $54m/s$ for 90s. In the end it was attributed to the turbulent conditions of the day of the last flight, where significantly more turbulence was present than during the flight test 32.

6 CONCLUSIONS AND LESSONS LEARNED

This article describes some of the challenges that can be expected during flight testing of active flutter suppression technologies on a subscale demonstrator. The required upgrades for the electronics and telemetry systems are presented. The differences between the planned and actual flight test schedules are underlined, proving that the weather conditions can be very influential if not enough flexibility is in place. The build-up sequences for testing and evaluating the effectiveness of flutter controllers in-flight are shared.

During the actual flight test, a few factors made it hard to use the flutter detector tool to its explained capability. Firstly, frequent data drops in the EDL channel made the flight test team ”blind”. Secondly, the development status of the tool made it difficult to apply the tool correctly. Finally, the team did not have enough confidence to correlate the flutter detector signal with the actual flutter events. Finally, it is clear that the current indicator is tailored to one particular flutter frequency, so it cannot be used as a generic flutter indicator. Future evolution of the algorithm could be a combination of the operational modal analysis method and a modal property matching varying frequency detector.

In addition to the above, the following lessons learned could be identified:

- The additional fuel available after the demonstrator rebuild proved extremely useful during the energy-demanding flutter suppression tests. The minimum reserve fuel was de-

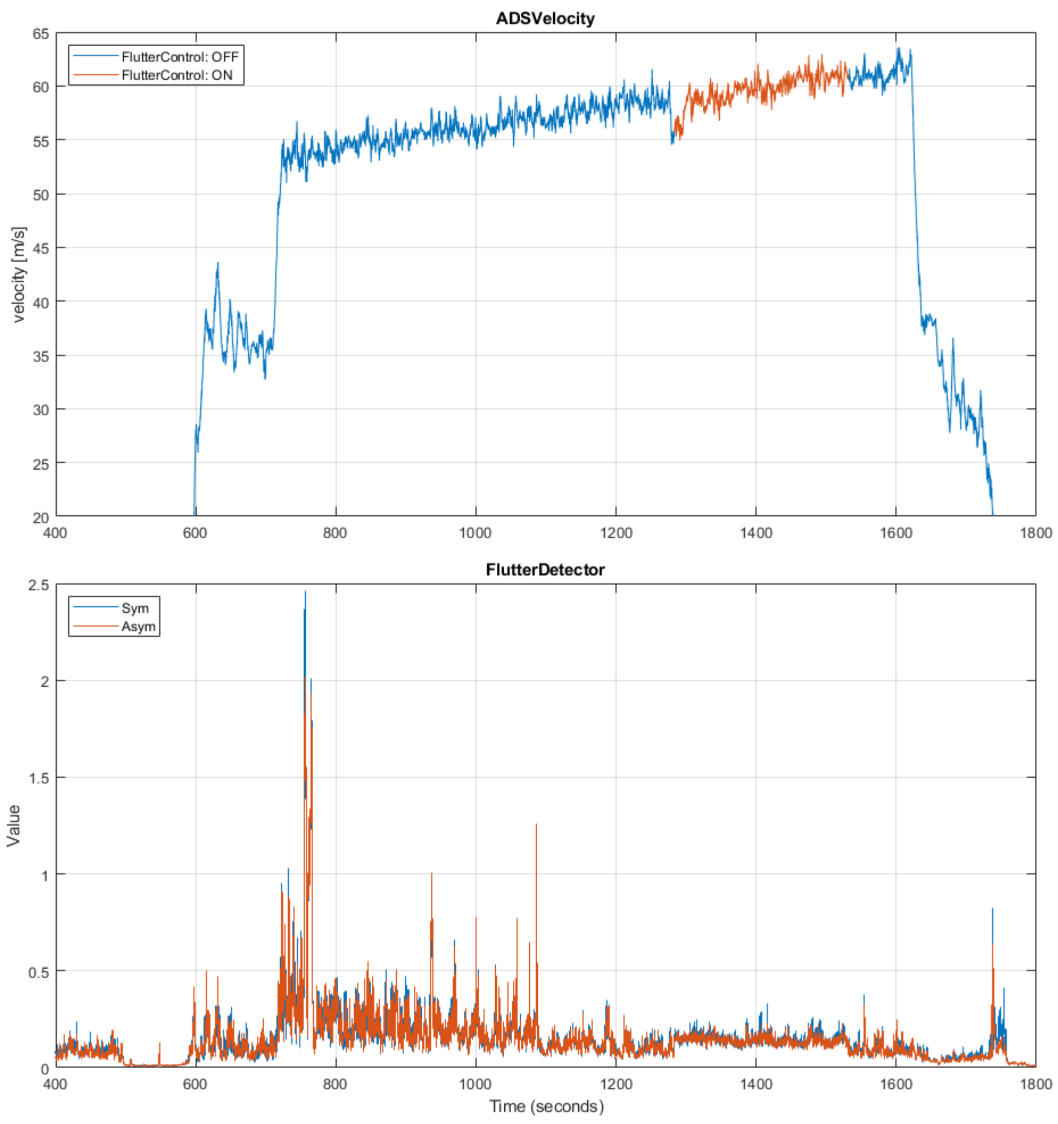

Figure 22: Flutter detector signal in FT37 (Ultimate flutter flight test).

fined as $0.75kg$ which was calculated for two go-arounds. Due to the high thrust requirements for the flutter test sequences, we usually landed with 10% fuel (around $1kg$) left. This underlines the importance of keeping enough fuel reserves during preliminary design (section 2).

- High frequency flutter suppression actuator will degrade the nearby mechanical components, such as rod linkages or bearings, faster (section 2.3).
- Having a flutter stopper mechanism on the aircraft can be useful if correct automated trigger sequence is implemented. For example, by using the flutter indicator signal (sections 2.4 and 4.2).
- Making a remotely controlled HIL platform can be useful during test campaigns, where software changes can be expected (section 2.2).
- If only a confined airspace is available, flying in big, steady circles is a plausible solution for flights that require high speeds or long test legs (section 4.1).
- In order to identify flutter in-flight, the crew must know not only how would it look like visually, but also on the available telemetry data. In our case, we were not confident enough to identify the flutter event in-flight, even though it was visible from the flutter detector telemetry data.

The flight test data of the project is freely available [16].

7 REFERENCES

[1] Meddaikar, Y. M., Dillinger, J., Klimmek, T., et al. (2019). Aircraft Aeroservoelastic Modelling of the FLEXOP Unmanned Flying Demonstrator. In *AIAA Scitech 2019 Forum*, January. Reston, Virginia: American Institute of Aeronautics and Astronautics. ISBN 978-1-62410-578-4. doi:10.2514/6.2019-1815.

[2] Luspay, T., Baár, T., Teubl, D., et al. (2019). Flight control design for a highly flexible flutter demonstrator. *AIAA Scitech 2019 Forum*, (January). doi:10.2514/6.2019-1817.

[3] Pusch, M., Ossmann, D., and Luspay, T. (2019). Structured control design for a highly flexible flutter demonstrator. *Aerospace*, 6(3). ISSN 2226-4310. doi:10.3390/aerospace6030027.

[4] Bartasevicius, J. and Hornung, M. (2023). Flight testing for flutter - operational design and lessons learned. In *SFTE 2023 International Symposium*. Annapolis: The Society of Flight Test Engineers, pp. 1–20.

[5] Yu, F., Teubl, D., Tóth, S., et al. (2023). D3.9 Advanced wing integration and ground test completed. Tech. rep., FLIPASED Deliverable.

[6] MapTool-dipul by Bundesministerium für Digitales und Verkehr. `https://maptool-dipul.dfs.de/`, accessed on 2024-05-24.

[7] Bartasevicius, J., Koeberle, S. J., Teubl, D., et al. (2021). Flight Testing of 65Kg T-Flex Subscale Demonstrator. In *32nd Congress of the International Council of the Aeronautical Sciences, ICAS 2021*. Shanghai, China. ISBN 9783932182914, pp. 1–16.

[8] Soal, K., Volkmar, R., Thiem, C., et al. *Flight Vibration Testing of the T-FLEX UAV using Online Modal Analysis*. doi:10.2514/6.2023-0373.

[9] Open MCT open mission control technologies. `https://nasa.github.io/openmct/`, accessed on 2024-04-23.

[10] Weber, K. S. J., Marius (2021). Ft-03: Open mct workshop. Workshop at the AIAA AVIATION Conference 2021.

[11] Gordon Leishman, J. (2024). *Introduction to Aerospace Flight Vehicles*. Eagle Pubs. doi:10.15394.

[12] Sinske, J., Govers, Y., Jelicic, G., et al. (2017). Flight testing using fast online aeroelastic identification techniques with dlr research aircraft halo. In *Proc. of the International Forum on Aeroelasticity and Structural Dynamics*.

[13] Lind, R., Voracek, D. F., Truax, R., et al. (2003). A flight test to demonstrate flutter and evaluate the flutterometer. *The Aeronautical Journal*, 107(1076), 577–588. doi:10.1017/S0001924000013798.

[14] Patartics, B., Lipták, G., Luspay, T., et al. (2021). Application of structured robust synthesis for flexible aircraft flutter suppression. *IEEE Transactions on Control Systems Technology*, 30(1), 311–325.

[15] Wüstenhagen, M., Ackermann, L., and Bartasevicius, J. (2021). Validation and Update of an Aeroservoelastic Model based on Flight Test Data.

[16] Bartasevicius, J., Teubl, D., Seren, T., et al. (2023). FliPASED Flight test data. doi: https://hdl.handle.net/21.15109/CONCORDA/PU4R32.

COPYRIGHT STATEMENT

ACKNOWLEDGMENTS

The work presented has been conducted within the framework of projects FLEXOP (grant agreement No. 636307) and FLiPASED (grant agreement No. 815058) funded from the European Union's Horizon 2020 research and innovation program.