FliPASED
WWW.FLIPASED.EU

D3.4 Sensor Concept Advanced Wing Finalized

Keith Soal (DLR), Yasser Meddaikar (DLR), Julius Bartasevicius (TUM), Sebastian Köberle (TUM), Dániel Teubl (TUM), Szabolcs Tóth (SZTAKI), Mihály Nagy (SZTAKI), László Gyulai (SZTAKI)

GA number: 815058
Project acronym: FLIPASED
Project title: FLIGHT PHASE ADAPTIVE AEROSERVO-ELASTIC AIRCRAFT DESIGN METHODS
Funding Scheme: H2020 **ID:** MG-3-1-2018
Latest version of Annex I: 1.1 released on 12/04/2019
Start date of project: 01/09/2019 **Duration:** 40 Months

Lead Beneficiary for this deliverable: SZTAKI
Last modified: 31/12/2021 **Status:** Delivered
Due date: 30/11/2020

Project co-ordinator name and organisation: Bálint Vanek, SZTAKI
Tel. and email: **+36 1 279 6113** vanek@sztaki.hu
Project website: www.flipased.eu

Dissemination Level		
PU	Public	X
CO	Confidential, only for members of the consortium (including the Commission Services)	

“This document is part of a project that has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 815058.”

Glossary

ADS Aeroprobe Air Data Sensor with μADC Air Data Computer. 5, 56

CAN Controller Area Network. 6, 8, 10, 47, 48

ECU Engine Control Unit for T-FLEX Jet Engine (BF B300F). 5, 7, 8, 49

EDL Engineering Data Link. 8, 9, 49

FCC Flight Control Computer . 7, 8, 12, 49, 50, 56

GCS Ground Control Station . 8, 13, 49

GVT Ground Vibration Test. 10

IMU Inertial Measurement Unit. 10

OBC-II On-Board Computer II, a companion for FCC . 5, 12, 13, 49

PWM Pulse Width Modulation . 6

RX-MUX Part of FCC, name is originated from multiplexing receiver signal, with the autopilot signals. It handles ECU, RC servos, and DirectDrive actuators.. 6–8, 49, 50, 56

SHM Servo Health Monitoring device . 5, 6, 47–49

T-FLEX Demonstrator Aircraft used in FLEXOP and FLiPASED project. 6, 12, 13, 35

TMS Thrust Measurement System. 13, 19, 56

Table of contents

List of Figures

1 Executive Summary

This document describes the design and implementation of the sensor layout and actuation system on the T-FLEX demonstrator, with emphasis on the configuration with the -3 wing. Work on the sensor concept of the new wing is based on the experience gained during previous test flights and based on the errors occurred during earlier development and integration. Not only new sensors were utilized, but the performance of the current ones were analyzed and improved to improve performance. Backward-compatibility must be maintained at as many parts as possible, therefore the on-board avionics is capable of reproducing former flight configurations to be able to produce results that are comparable.

At the beginning of the deliverable an overview of the sensor routes is presented in Section 2.

In the following parts, the components of the sensor and actuator system are described. Almost every subsystem got some modifications compared with the previous setup used in FLEXOP. These are:

- The number of the flutter IMUs were increased and the measured signals were refined with new measurement mode for better modal analysis Section 3
- The onboard computers and other electronics were modified for the new components, integration of them was made: OBC-II in Section 4
- xSens and ADS was relocated to achieve better signal-to-noise ratio (Section 5)
- Feedback sensors on actuators were improved: on the jet engine, the fuel flow by the ECU (Subsection 2.3) and the thrust measurement became more precise (Section 6).
- Cameras were mounted on fuselage for monitoring wing shape (Section 7)
- Actuators for the wing ailerons with SHM sensors were investigated in Section 8, and an improvement was suggested on them. DirectDrive actuator integration was continued.

With the improvements in sensor system, and involving new sensors, the aircraft models and controllers became more accurate. The sections above lead to the new wing -3 which will be the main platform for the new avionic system. Section 9 summarizes the design objectives of the new wing and the decisions which have to be made in the design process. These include the following engineering tasks:

- Actuator number was selected
- Actuator feedback systems were compared, improvements in actuator systems were considered
- Sensors for the wing shape measurement and modal analysis were developed and integrated for observing the behavior of the new wings

2 Sensor placement, data routes and processing

The T-FLEX demonstrator aircraft will be used with 4 different wings for different purposes. Each wing has different configurations of flaps, actuators and sensors. The -0, -1 and -2 wings have 4 flaps, the -3 wing will be manufactured with 9 flaps (see Section 9). The number of the flaps defines the number of actuators and the purpose of the wings determines the types of the actuators.

The -0 wings are conventional wings, which are used for initial flights and test all the system of the aircraft. The -2 wings are lighter wings with higher flutter speed than -1 wings. The -0 and -2 wings have the same actuator configuration, those use the HBL599 PWM controlled servos for all four flaps. These actuators are require an external diagnostic sensor, which called Servo Health Monitoring (SHM).

The -1 wing designed with low flutter speed to test the active flutter control, which requires a higher speed and higher torque actuator on the outer flaps. We call this actuator Direct Drive and an own controller and encoder are fitted into it, so it is capable to give high accuracy position feedback and diagnostic information. For the other flaps we use the HBL599 PWM controlled servos with SHM units like for the -0 and -2 wings.

According to the main concept for the -3 wings we will use servos which communicate through CAN bus and have the own sensors which give back position and other required diagnostic information. The possible advantages are discussed in Section 8, and candidates for the improved servo actuators are also presented.

The system uses 6 IMUs per wing and two in the tail for wing shape estimation and modal analysis. 1 shows the IMU placement in the wings. The IMU configuration is the same in all 4 wings.

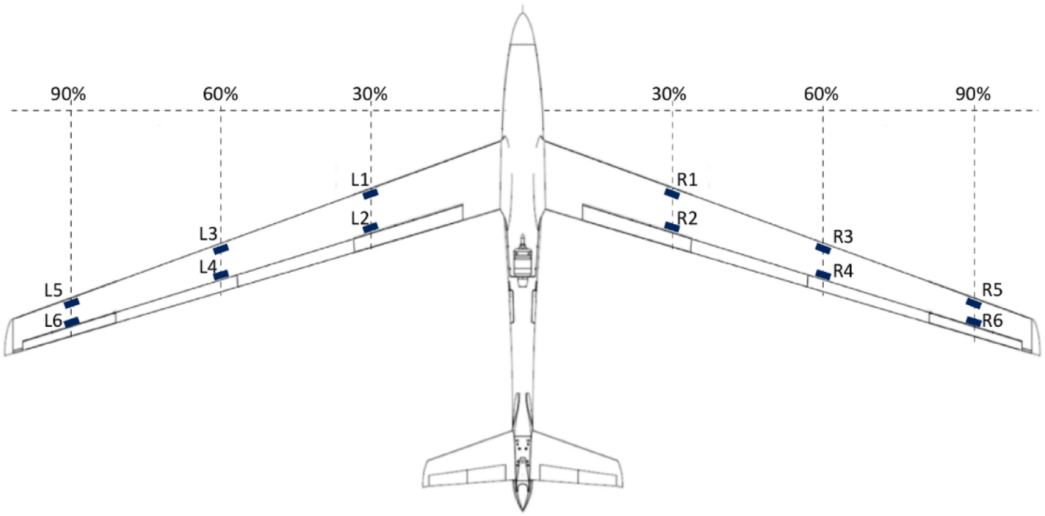


Figure 1: Flutter IMU placement in wings

The CAN actuators and the Direct Drive with their sensors give their feedback and diagnostic information to the RX-MUX module of the FCC. The flutter IMUs and the Servo Health Monitoring (SHM) systems are connected to the interface panel of the Flight Control Computer which called flightHAT.

For navigation purposes the demonstrator aircraft uses GNSS/INS and air data and flow measurement

sensor systems. To measure fuel consumption, a fuel flow measurement system placed. These are all connected to the interface panel of the Flight Control Computer (FCC). The jet engine (ECU) with it's sensors connected to the RX-MUX module.

- Data route, signal types, frequencies

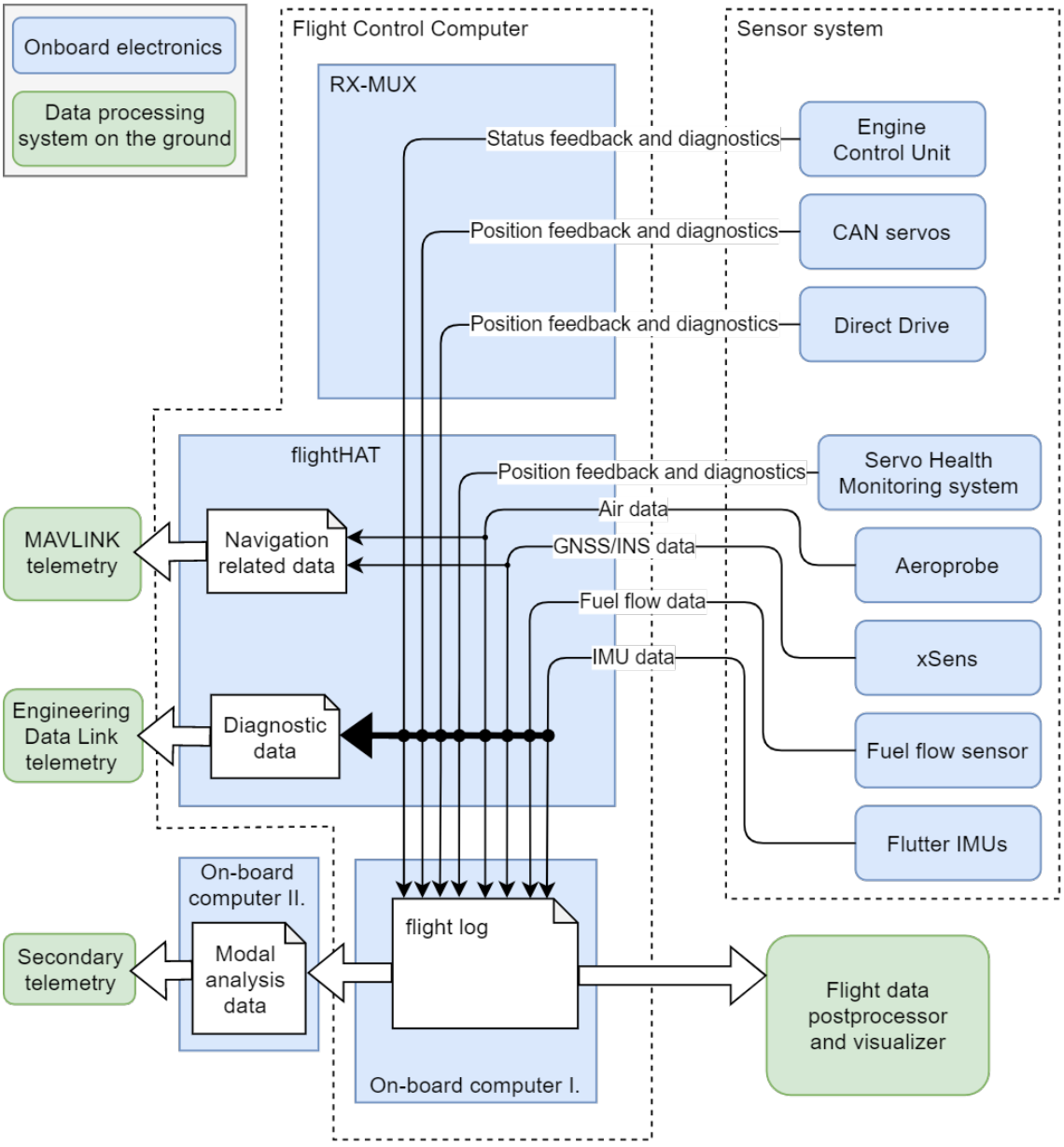


Figure 2: Sensor dataroutes

2.1 RX-MUX improvements

The FCC system is improved further for the new wing sensor and actuator system. The whole concept of the on-board electronics development for the -3 wing aims to substitute as many analog signals with digital ones as possible. Therefore noisy environment is not affecting the actuator reference signals and other communication. However, the following protocols increase the necessary computational power, and this can be fulfilled by redesigning the RX-MUX unit with a more powerful STM32F4 controller instead of the currently used PIC16 one. Since the flightHAT panel also uses a similar microcontroller, the FCC becomes more unified in terms of hardware and software development environments.

The following main features of the new RX-MUX board are related to sensor and actuator handling:

1. As mentioned previously in this section, CAN servos are more trustworthy than the currently used HBL599 ones, due to their digital protocol instead of the currently used with analog PWM reference (see Section 8).
2. Currently the RXMUX unit receives the commands from the pilot(s) via analog signals. The receivers for the Jeti and Graupner RC transmitter units using PPM signals (Pulse Position Modulation). Replacing them with their digital equivalents (exBUS for JETI, SBus for Graupner) is prefered. We expect more robust communication and these digital protocols have the advantage of bidirectional communication to notify the pilot on the screen of the transmitter.
3. The communication with DirectDrive actuator via CANopen protocol cannot be handled properly with the current controller, as discussed in Subsection 8.2.

The development is in progress harmonized with the new -3 wing manufacturing process discussed in Section 9.

2.2 Telemetry and post-processing

A new addition regarding the telemetry of the aircraft is that the autopilot commands which are sent from the custom Mission Planner interface are now logged on the FCC. This is very important, because otherwise these signals are not saved and the inspection of the behaviour of the autopilot after flights is impossible. For example, it is crucial to see if the command message sent from the GCS to set the airspeed reference from 34 m/s to 38 m/s has really arrived onto the aircraft or not.

2.3 Fuel flow measurement changes

The previous concept of the fuel flow measurement system stated, that the value of fuel flow (grams per seconds) has to be sent to the EDL via telemetry, and then there it is integrated to get the remaining fuel on the aircraft. This idea is only applicable if there are no lost messages, but of course this is an ideal situation. To prevent miscalculations due to packet drops, the fuel flow measurement data has to be scaled and integrated on the flightHAT where the fuel flow sensor is connected. So the current data flow of the fuel measurement is the following.

1. The raw measurement value from the ECU arrives on the flightHAT.
2. The raw measurement data is scaled according to measurements by TUM, the end value is the fuel flow in grams/sec.

3. The fuel flow value is added to a counter, so the value of the counter is the consumed fuel.
4. The consumed fuel value is sent to EDL via telemetry and the remaining fuel is visualized in the EDL user interface.

In summary, the amount of the sensors and actuators increased in the new concept. The amount of the logged data and parameters are also extended: this helps to validate aircraft models and controllers with flight testing. Extra on-board computational power is provided, but the objective was not to increase burden of the development. Later in this document, the advantages and the possibilities are discussed in depth.

3 Flutter IMU configuration

Based on results from the Ground Vibration Test (GVT) it was decided to re-configure the IMUs as well as instrument the empennage and fuselage with additional IMUs, as shown in Figure 3. This configuration increases the number of observable modes from 6 to 14 in the bandwidth to 30 Hz.

One IMU in each tail measuring acceleration in three directions will enable the identification of 4 V-tail modes in the bandwidth 17 Hz - 33 Hz. The IMUs have been located as far from the root as possible. The sensors are oriented in the local coordinate system to assist for practical installation purposes, and can be decomposed into the global co-ordinate system using Euler angles. For the purpose of modal analysis no angular rates are required on the IMUs in the empennage or fuselage, thus saving bandwidth on the CAN bus.

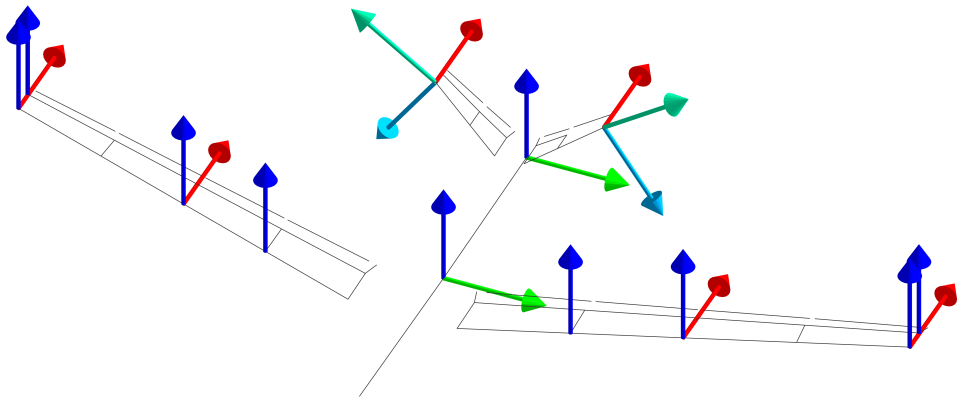


Figure 3: IMU re-configuration and additional instrumentation.

The IMUs in the wings were re-configured to include in plane measurements. Since they are glued into the wings and not accessible from outside, we took advantage of the bootloader on the flutterIMU boards to reprogram them via CAN bus. The new software is backward compatible with previous FCC software, therefore previous measurements are also reproducible. Not only the new wing will use this new mode, but the existing wings (-0, -1, -2) were also reprogrammed successfully. This allows the identification and tracking of the highly non-linear in-plane wing bending mode shown in Figure 4 and Figure 5.

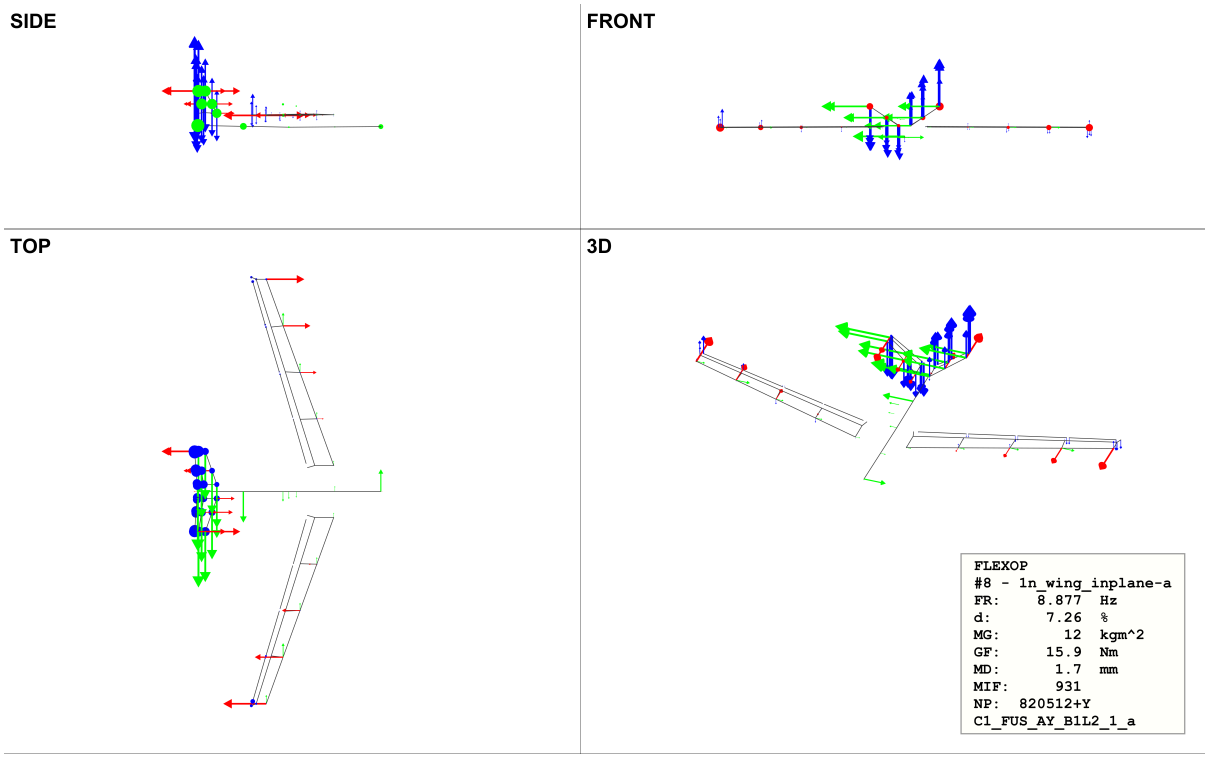


Figure 4: Four view plot of 1n wing in-plane bending.

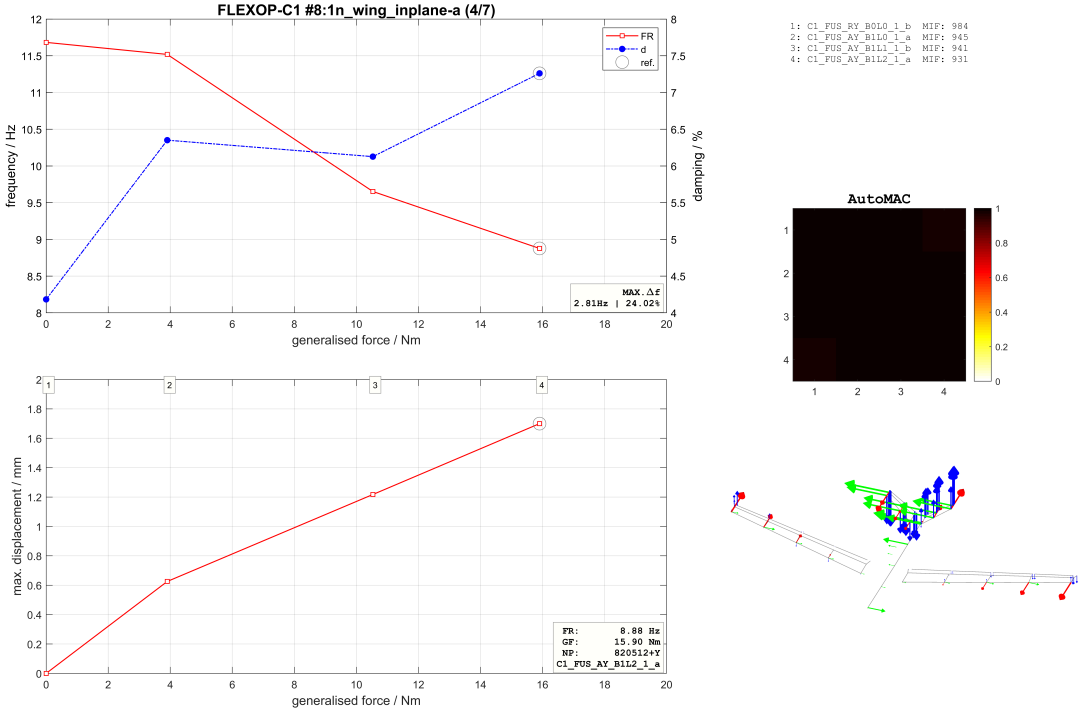


Figure 5: Non-linearity plot of 1n wing in-plane bending.

4 Onboard computer II and It's telemetry

Originally, this platform was introduced only for hosting a 'flutter sensing' or 'flutterometer' functionality developed for online flutter mode analysis - see [13]. The need for a second on-board computer(OBC-II) arised, to minimize the influence on development and computational usage of the main on board computer, the Flight Control Computer (FCC). With the separation of the functionalities, the two system can be developed, tested and integrated mostly without influencing one another, on top of that no hardware resources need to be shared.

Currently, the OBC-II has three main functionality. Firstly, hosting a modal analysis algorithm, which can track the structural modes of the T-FLEX during operation. Secondly, supporting additional sensors and telemetry as well as data storage capacity. The telemetry link works independently of the two telemetry links used to control and monitor of the mission, which allows us a highly customize interface to monitor any data, which is not transmitted over the two main links. Currently, similar antenna types are used for this link as the ones used for the two main links, but this interface is easily replaceable to support higher bandwidth solutions [13]. Thirdly, it acts as a central logging and telemetry interface for any custom sensors introduced to the T-FLEX. For example, the data from the previously developed thrust measurement system (described in section 6) logged on this platform. The overall interface structure of the OBC-II can be seen in figure 6.

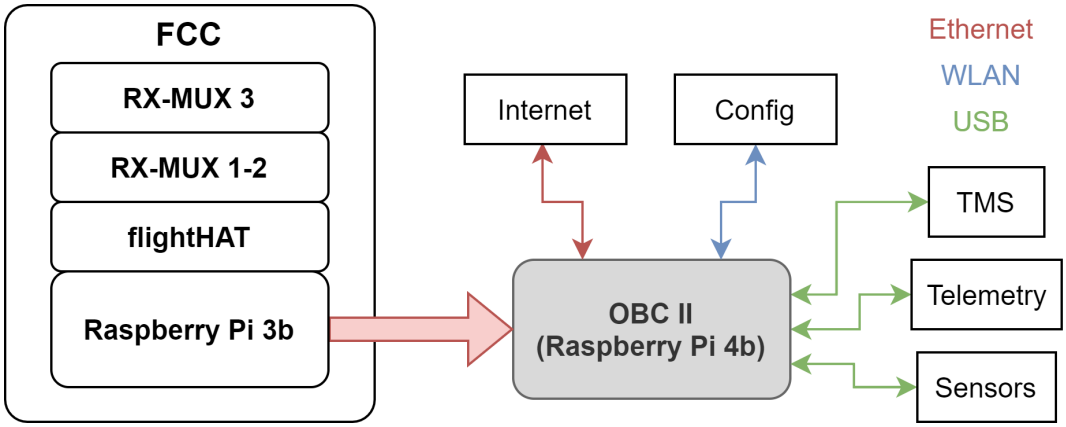


Figure 6: Interface structure of the OBC-II

4.1 System setup and main components

Currently, the system has 3 network interfaces, each with its unique purpose. The on-board Ethernet port of the Raspberry Pi itself is used in a fixed network configuration to provide high-bandwidth data communication between the FCC and the OBC-II. The second is the on-board WLAN interface, which is used as a configurable access point setup. The purpose of the WLAN is the provision of a wireless connection to any device, independently of the used operation system or hardware ports available on hardware side. The third interface is an additional usb-ethernet adapter, which is configured to support simple local networks with DHCP option to share internet access to the OBC-II itself. That is required to easily do updates - system or configuration - without reconfiguration of any other ports.

On the system setup side, a minimal auto-startup toolchain is created, which runs all the pre-configured applications during boot at a dedicated CPU core. This toolchain uses a custom systemd service, which allows to use each application as a service and dedicates them to a user assigned cpu core. To achieve readability, all the custom user messages of the services are pointed to the normal systemlog, which permanently stores the related information.

To ensure that update of old applications and the deployment of new applications is smooth, the custom applications and the toolchain are linked into one single git-repository. That allows easy identification of the currently used application and toolchain version between development and real target systems.

4.2 Current configuration

Currently there are two main functions configured on the platform, on top of the communication with the FCC itself. One is the online modal analysis tool, and the second one is the logging part of the thrust measurement system.

The online modal analysis tool, developed earlier to measure and indicate the different structural modes of the T-FLEX, is written in python and deployed on two different cores of the OBC-II. The result of the algorithm - eigenvalues and damping ratios associated with different structural modes - are transmitted down to the GCS via the dedicated telemetry link of the OBC-II.

The second currently deployed application is the logging part of the thrust-measurement systems (TMS). The application - at the current state - only stores the incoming data from electrical part of the TMS itself, along with local time data. The logging side is written in python. A simple configuration of the udev - universal device manager of the linux kernel - is used to create a unique name for the arduino platform, which does the low level data sampling and transmission of the data via USB.

4.3 Expansion possibilities

In the future, the telemetry channel is planned to replaced with a 5 Ghz wifi on-board antenna, and a highly directional ground antenna. The latter will be equipped on a tracker system, which will provide the necessary orientation during operation. With that system, the practical bandwidth for communication can be raised up to 500 Mbps.
To date, the tracker has been successfully field-tested and is currently configured for more rapid, stand-alone setup on the flight field and for integration in the complex network system necessary for operating the T-FLEX demonstrator in the Ground Control Station. The integration will allow controlling the tracker from the NASA Open MCT framework that is displaying the telemetry data streamed from the OBC-II, such creating a single point of access for operators. The GUI is depicted in Fig. 7.

The next steps will address the configuration of the network devices in order to stream the data to the respective stakeholders.

Due to the implemented auto-start tool-chain, implementation of new sensors are easily possible just by following the given templates in the software side. Using the standard usb ports as inputs for custom sensors serves as a clear adjustable hardware interface. With this, any sensor which can communicate through a usb interface can be connected to the OBC-II.

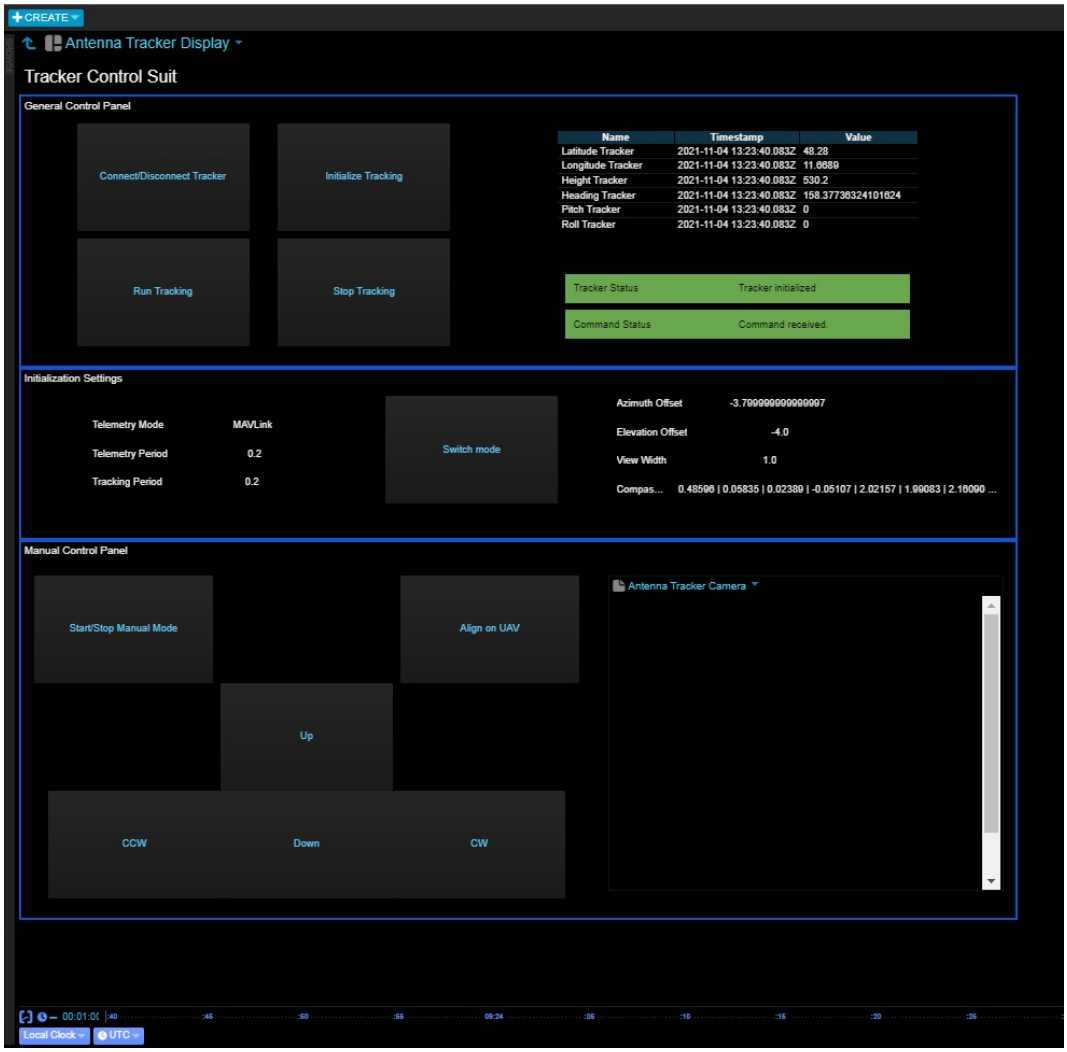


Figure 7: Interface to the tracker implemented in NASA's Open MCT framework.

5 Air data sensor and IMU placement and configuration

During the flight test data analysis phase (with the flight test data from 2019) it was noticed that the angle of attack signal is corrupted with noise which is not visible in angle of sideslip. Figure 8 shows this difference in signal noise during flight path reconstruction of a flight segment. Angle of attack sensor appears to have a visible additional noise to it, which does not exist in the angle of sideslip. The sensor was checked in the wind tunnel and it was clear that the problem is not with the sensor itself as the spectral densities in both angles were the same (Figure 9, right). It was therefore postulated that maybe the mounting of the sensor is not rigid in the longitudinal plane.

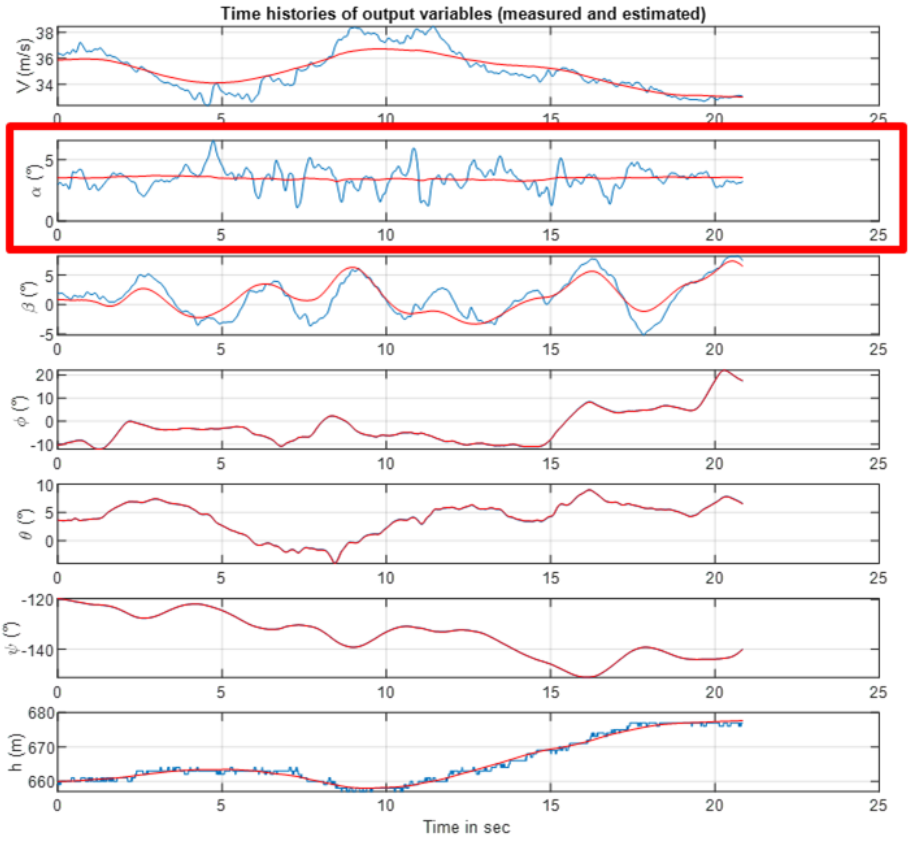


Figure 8: Data compatibility analysis (also known as flight path reconstruction). Ideally, the measured and estimated signals should match. Blue- measured signal, red- estimated signal. Clear difference in noise levels between angle of attack (alpha) and angle of sideslip (beta) can be seen.

The mounting of the pitot boom was therefore investigated. The mounting of sensor was done in a way that the air data boom would go through the nose section of the fuselage and then would be mounted on the payload rack board at the root (Figure 10 and Figure 11). It was then realised that the payload

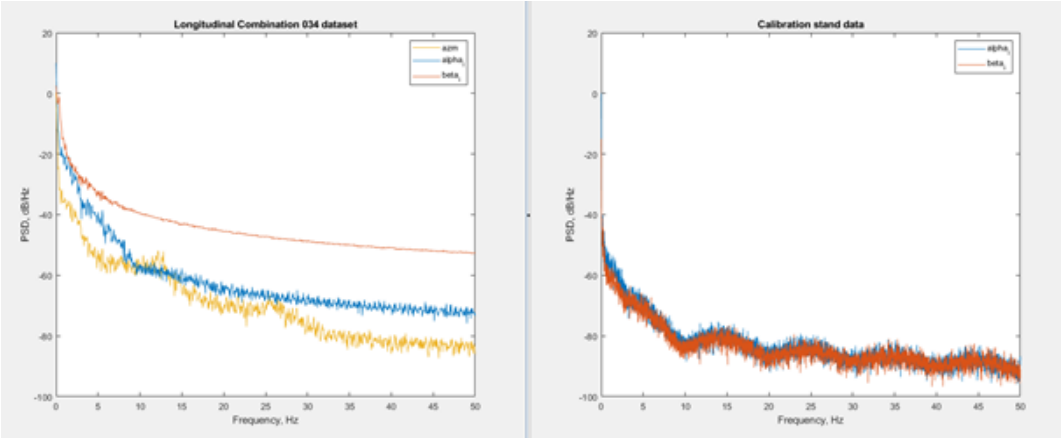


Figure 9: Angle of attack and angle of sideslip signal comparison from in-flight data (left) and wind-tunnel data (right).

rack, which is a 3mm glass fibre board with many equipment mounted on it, would move vertically during manoeuvres and in this way would move the root mount of the boom as well. Considering that the middle point of the boom, which goes through the fuselage, acts as a rotation point, the actual sensor head therefore gets deflected (Figure 12. It was also recognised, that the main IMU sensor is also mounted on the flexible glass-fibre board. Therefore relocation of both main sensors (xSens and Aeroprobe) has to be done.

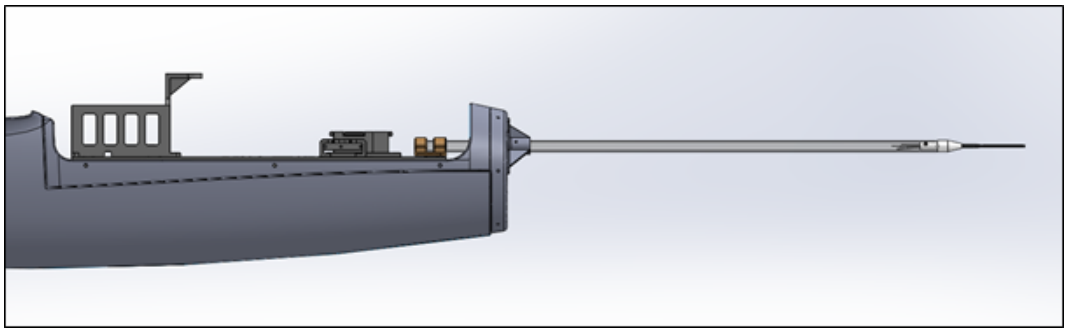


Figure 10: Air-data boom mount.

The air-data boom mount was upgraded by designing a new, rigid structure from carbon-fibre sandwich in the nose section of the fuselage (Figure 13). The purpose of the structure was to decouple the air-data boom mount from the rest of the payload rack and increase the stiffness of the point where the boom intersects the fuselage (the front wall). Solution was implemented.

In addition, the main IMU sensor xSens was relocated onto a stiff mounting point next to the fuel tanks. This reduces the coupling in between the rotational and longitudinal motion measurements,

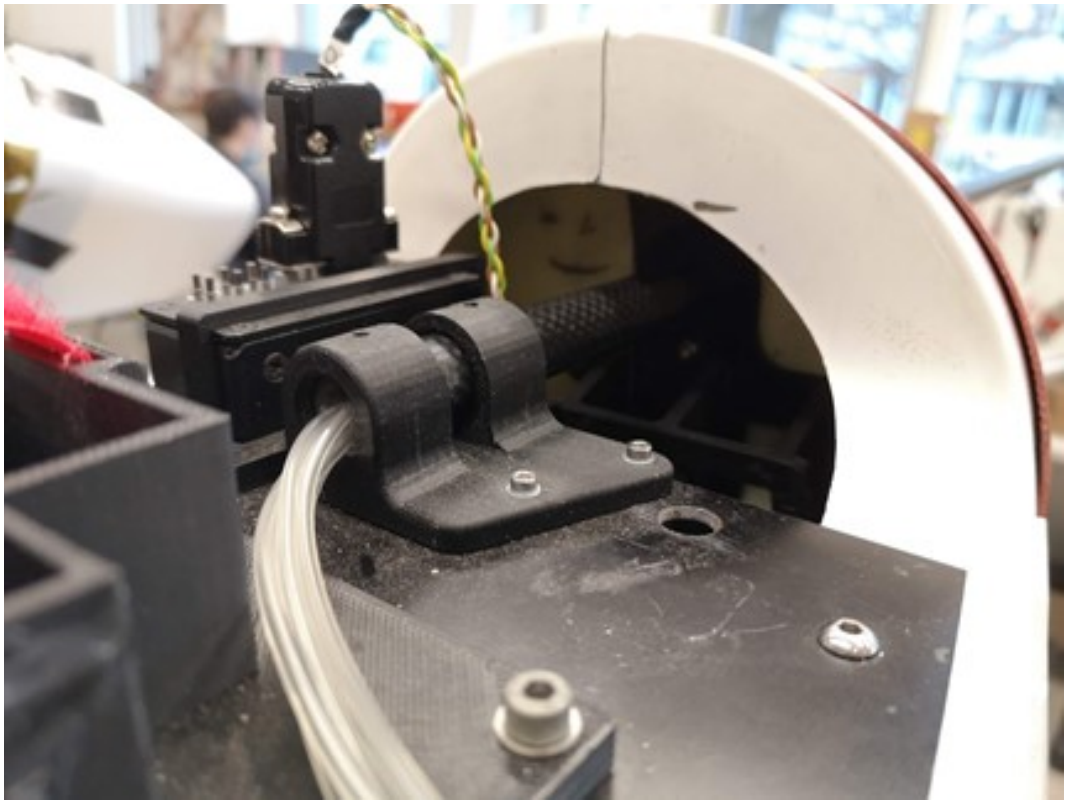


Figure 11: Air-data boom mount at the root.

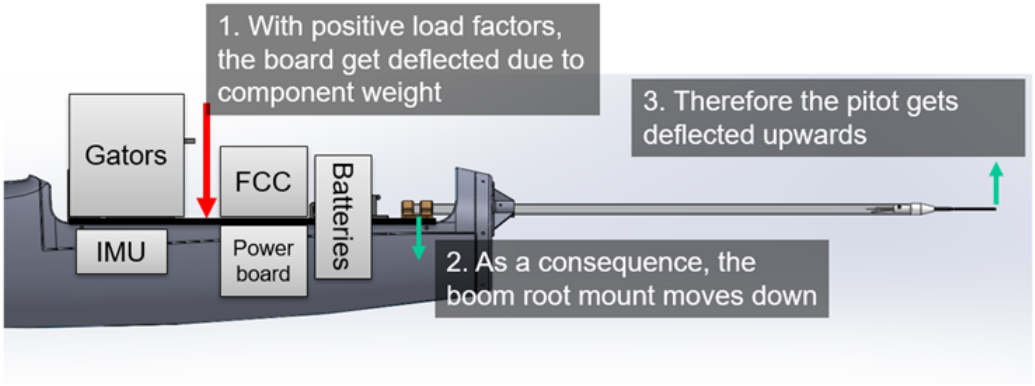


Figure 12: Air-data boom flexibility mechanism.

consequently reducing the possible errors.

Two test flights have been completed with the new sensor setup. The sensor error analysis of the new setup is not yet completed.

Furthermore, implementation of a temperature sensor on-board is currently in progress. Outside temperature influences the ambient conditions, which are required to accurately analyse the data. Currently, the METAR information is used from the local weather station. But as it only gives a single tempera-

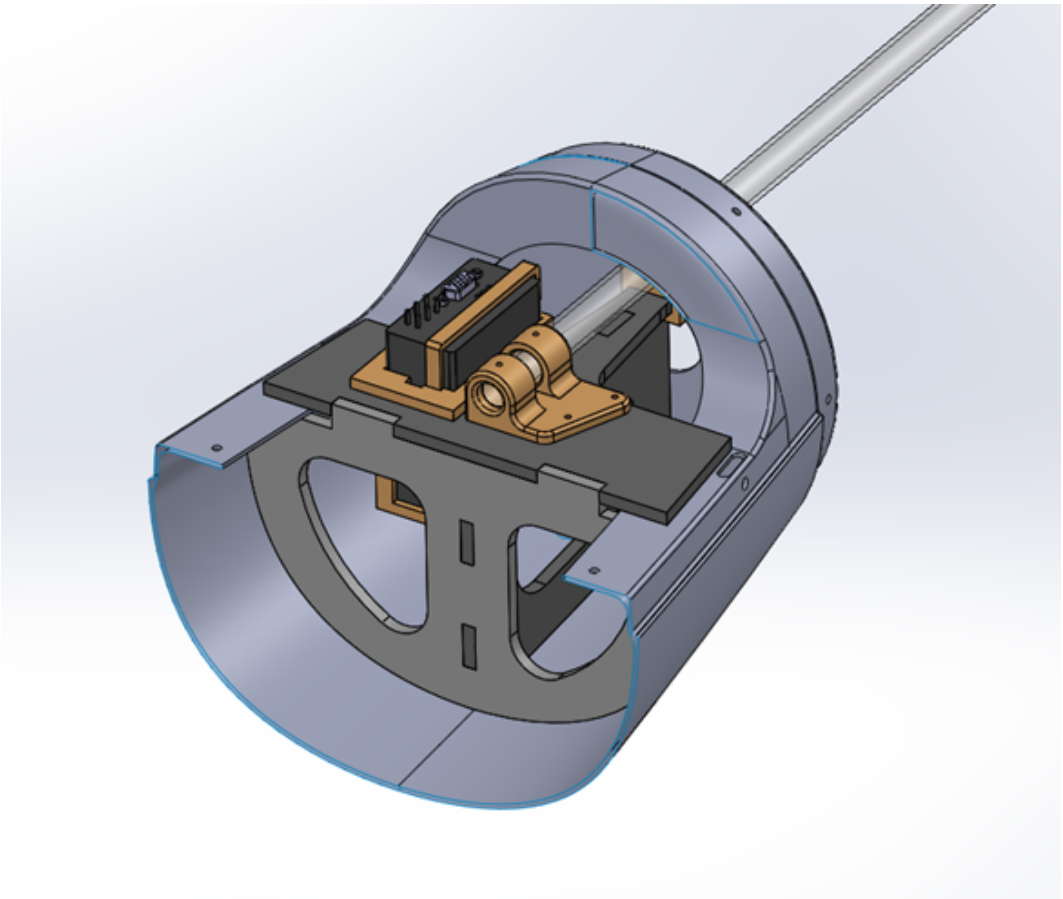


Figure 13: Upgraded air-data boom mount in the nose section of the fuselage.

ture point for the whole flight, it is expected that a continuous temperature measurement could improve the data quality. This can be especially important considering the wide range of temperatures that the aircraft is operated in ($+5$ to $+30$ degrees Celsius).

6 Thrust measurement system

In order to accurately derive drag of the demonstrator in-flight, thrust measurement system TMS was designed, tested and built. Very good accuracy was observed during calibration and system has already been tested in-flight. It is also expected that the measurements can be used to retrospectively correct the thrust measurements for previous T-FLEX flights.

All areas of system design were thoroughly described in a conference article that will be presented in the AIAA SciTech 2022 conference in January, 2022. The draft of the paper is included in the following pages.

Design and testing of an in-flight thrust measurement system for a pylon-mounted miniature jet engine

Julius Bartasevicius*, Pedro Alexandre Tonet Fleig†, Annina Metzner‡ and Mirko Hornung§
Technical University of Munich, Boltzmannstrasse 15, 85748 Garching

Optimisation of aircraft's performance often requires careful consideration of aerodynamic drag. However, direct measurement of drag of a flying vehicle is not feasible. Therefore, in order to measure the change in drag for different configurations of a flying aircraft, in-flight thrust measurement is necessary, which can consequently be used to derive drag. For this reason, design of an in-flight thrust measurement system for a pylon-mounted jet engine is presented. The system is based on trunnion thrust method. The design process is described and includes a review of state of the art as well as measurement error considerations. Calibration methods are presented. During the calibration, the thrust root-mean-square error of $0.64N$ was observed. The system was flight tested and proved to work reliably in real-life conditions. Finally, the flight test data was used to generate a thrust model based on engine parameters.

I. Introduction

It is not feasible to directly measure aerodynamic drag in-flight. Therefore, in order to quantify any changes in drag on a flying aircraft, knowledge about the other forces acting on it, namely lift, weight and thrust, is required. This work presents the design of a thrust measurement system for a miniature jet engine within the framework of the project FLiPASED (Flight Phase Adaptive Aero-Servo-Elastic Aircraft Design Methods, `https://cordis.europa.eu/project/id/815058`).

One of the goals of the project FLiPASED is to reduce the induced drag in-flight. This will be done by optimising control surface deflections, which in turn changes the load shape distribution over the wings. The reduction of induced drag will be shown in-flight with the help of a subscale flight demonstrator (SFD) T-FLEX which has been developed for the predecessor project, FLEXOP (Flutter Free Flight Envelope Expansion for Economical Performance Improvement, `https://flexop.eu/news`). Within the FLEXOP project a 65kg take-off weight, 7m wingspan swept wing unmanned aircraft was designed and built.

As the change of the induced drag is expected to be within 2-5 percent [1], accurate knowledge about the thrust provided by the engine is required. Consequently, to supplement the thrust data from an already available engine model, additional measurements in-flight are required. This is achieved by implementing a load-cell based thrust measurement system that can measure the applied thrust with accuracy of 2 percent. The correct functioning of the system is key for validating the drag reduction measures within the FLiPASED project.

Within this work a review of existing methods for thrust measurements with focus on applications for subscale demonstrators will be presented (section II). System requirements and design process applied to T-FLEX SFD will be described (section III), as well as calibration procedures (section IV) and the in-flight measurement results (section V).

II. State of the art

The existing thrust measurement methods are discussed here. Section II.E focuses specifically on methods that have been applied to measure thrust of miniature engines commonly used to power unmanned vehicles.

A. Gas generator method

The gas generator method uses the measurements taken inside the engine (mass flow, pressure or temperature) to derive the thrust. These are taken at various stations of the engine (commonly nozzle and inlet) and allow gross thrust derivation. MIDAP Study Group [2]. The main disadvantage of this method is that it requires multiple sensors installed

*Research Assistant, Institute of Aircraft Design, Boltzmannstrasse 15, 85748 Garching, Germany, julius.bartasevicius@tum.de
†Alumnus TUM M.Sc. Aerospace, Institute of Aircraft Design, Boltzmannstrasse 15, 85748 Garching, Germany, pedroafleig@outlook.com
‡B.Sc. Aerospace, Institute of Aircraft Design, Boltzmannstrasse 15, 85748 Garching, Germany, annina.metzner@tum.de
§Professor, Head of the Institute of Aircraft Design, Boltzmannstrasse 15, 85748 Garching, Germany, mirko.hornung@tum.de

within the engine. While this might not be a problem for power plants used on manned aircraft, implementation of additional sensors in the miniature engines becomes more complex.

B. Brochure method

The brochure method is considered the simplest thrust measurement technique in terms of required measurements and installed equipment [2]. It can be implemented, for example, using only the rotational speed of one of the shafts as an input. However, this method requires extensive tests and calibration, usually performed by the engine's manufacturer to create data tables that correlate certain parameters with the generated thrust. Therefore, the brochure method, while reliable and accurate when sufficient data and additional engine parameters are available for validation, is less suitable for a small-scale engine. This is mainly due to the following reasons: Firstly, there is usually very limited test data provided by the engine's manufacturer. Secondly, due to non-existent certification requirements for such engines, higher deviations between two engines of the same type can be expected.

C. Swinging probe method

The swinging probe method uses calibrated sensors traversing the engine's exhaust nozzle to perform total and static pressure, total temperature, and flow direction measurements [2]. In some cases, multiple sensors are placed in rakes to obtain better results, as shown by Davidson [3] in an application that was able to measure the net thrust. This configuration of the probes has the advantage of not requiring provisions for their installation in the engine's core, because all sensory is placed on the outer side. However, as the sensors and mountings are within the high pressure and temperature area behind the engine, the complexity and cost of the measurement system is high.

D. Trunnion thrust method

The most direct method for measuring thrust requires no pressure, temperature, or airflow sensory. Instead, the trunnion thrust method directly measures the force imposed by the propulsion unit on its mounting structure [2]. This makes the technique particularly attractive when using thrust for drag computations, because the force vector created by the propulsion unit can be measured directly.

However, aircraft engine attachments are complex structures. These are often statically indeterminate and have more than the minimum number of support points to ensure higher safety level. Moreover, they not only contain load bearing parts, but also wires, pipes, and hoses which can affect the load measurement performed on the supports. Accordingly, the trunnion thrust method is considered unfeasible for large aircraft [2]. Nevertheless, experiments have been successfully performed. For instance, the tests performed in Conners and Sims [4] were able to obtain thrust values in a supersonic aircraft implementing direct measurements. Muhammad et al. [5] has developed a system with this technique for a propeller-driven aircraft and obtained values that were in accordance with other thrust measurement methods tested. This work also emphasized the importance of Finite-Element Method (FEM) analysis to ensure an optimal placement and geometry of load cells and strain gauges.

E. Thrust measurement of miniature engines

Most of the methods mentioned above were developed for manned aeroplanes with engines that went through extensive certification procedures and tests. In contrast, in the unmanned aviation many propulsion components are high-grade radio-controlled model engines with minimal performance data sheets and low or non-existent standards for certification. Therefore, interested users have to perform the engine performance analysis on their own. Some of such examples are given below.

Martinez [6] investigated methods to measure the thrust of a miniature jet engine integrated within an airframe. It included an extensive analysis to determine the best cell geometry and strain gauge positioning to eliminate the influence of lateral and vertical forces that could affect in-flight readings. The final design was a pair of horizontal metal holders instrumented with strain-gauges. Unfortunately after installation it was discovered that the thermal expansion of the engine influenced the measurements too much.

Simavilla [7] has also designed a system based on load cell force measurement. Its sensor placement and attachment reduced the heat-induced effects. Moreover, this design highlighted the requirement of a well-defined and compatible load path for the chosen sensor configuration. As a result, the presented solution used a hinged assembly in one of the engine attachment points and achieved 1% measurement accuracy for a $90N$ power-plant.

Bronz et al. [8] measured thrust by Gaussian fusion of two methods: a pre-calibrated motor in the wind tunnel (based on an airspeed estimate) and a direct force measurement sensor. The direct force measurement was done with a thin-film force sensor which was integrated within the motor mount. This sensor proved sensitive to the vibrations generated by the propeller, which significantly increased signal noise.

Sartori and Yu [9] investigated thrust measurement for a quadrocopter equipped with brushless electric motors and plastic propellers. They identified the problems within the usual Blade Element Theory approach and proposed an approach which, by experimentally measuring additional propeller parameters, improves the thrust estimation.

Bergmann et al. [10] presents an on-board thrust measurement system applied on an electrically-powered propeller UAV. The concept uses a load-cell placed in between the airframe and the electric motor. Tension and torque are measured by the load-cell. While the authors note that the accuracy of the concept was demonstrated under lab conditions, high deviation of measured in-flight thrust is reported, which is attributed to the fluctuations in propeller speed.

III. System design

A. Description of the demonstrator

The T-FLEX technology demonstrator is a jet-engine-powered UAV with 65kg take-off mass and 7.1m wingspan (Fig. 1). The UAV is flown manually by pilot via external vision. Rate control flight mode is used, where surface deflections are directly linked to the joystick positions on the transmitter. The autopilot is used only during some test sequences, but not during take-offs or landings.

Fig. 1 T-FLEX Subscale flight demonstrator during landing phase.

The aircraft is equipped with integrated measurement equipment. Air data (aerodynamic angles, airspeed and pressures), position (GPS coordinates) and inertial parameters (accelerations, attitude angles) are being logged on the aircraft. In addition, wings are equipped with multiple inertial measurement units spaced along the wingspan for vibration measurement.

The geometry of the aircraft is summarized in Table 1.

For further background of flight test operations of the T-FLEX demonstrator, please consult Bartasevicius et al. [11].

Table 1 Geometry of FLEXOP UAV.

Wing span, m:	7.07	Tail projected span, m:	1.27
Wing area, m^2:	2.53	Tail area, m^2:	0.39
Wing aspect ratio:	19.74	Tail aspect ratio:	4.2
Wing incidence, deg:	-0.52	Tail incidence, deg:	-4.33
Wing 0.25c sweep, deg:	18.36	Tail 0.25c sweep, deg:	19.83
Wing taper ratio:	0.5	Tail taper ratio:	0.52
Wing twist, deg:	-2	Tail dihedral, deg:	35
Number of wing control surfaces:	8	Number of tail control surfaces:	4
Fuselage length, m:	3.42		
Fuselage maximum height, m:	0.315		
Fuselage maximum width m:	0.3		

B. Jet engine description

The main requirements while designing the propulsion system for T-FLEX were high acceleration, low vibration and precise speed tracking [12]. Taking these requirements into account, a jet engine paired with a fast-response airbrake system [13] was selected. The jet engine is a BF B300F turbine with 300N maximum thrust capability. The engine was mounted on a pylon above the fuselage with the fuel tank located directly below it with the intent to keep the same centre of gravity throughout the flight.

The engine is round in shape and is secured to the aircraft via a steel-cage (Fig. 2). The cage is mounted on four aluminium holders attached to the main propulsion rack structure made out of carbon plates.

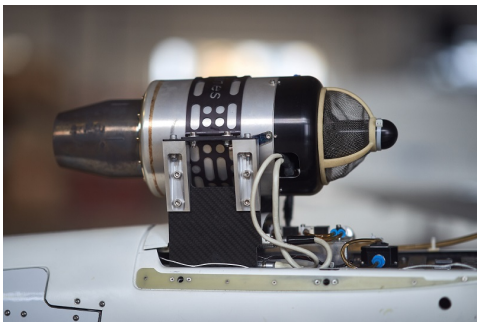


Fig. 2 Mounting of the BF B300F turbine on the aircraft.

C. Design requirements

The intentions of designing a thrust measurement system were to improve the available data for flight model identification and to allow quantify the drag reduction when active wing shape control is used. The expected overall drag reduction is in the order of 2-5 percent, which for cruise flight results in $1-3N$. The lower limit of 2% was taken as the required accuracy for the thrust measurement system.

The system was also required to have a measurement range over the whole available thrust envelope $(0-300N)$ and for the complete duration of flight (minimum of 30 minutes). Due to the slow dynamics of the jet turbine and therefore no fast changes in thrust values, a minimum sample rate of 50Hz was chosen.

Environmental conditions also had to be taken into account. Temperature, altitude and pressure as well as weather-induced conditions such as wind and rain were to be expected. Additionally, it had to be possible to compensate the measurements for off-level flight condition. Measurement of net thrust was required.

Table 2 summarizes the requirements.

Table 2 Summary of design requirements for the thrust measurement system.

Sub-Requirement	Value
Range of Measurement	$0 \leq T \leq 300$ N
Precision of Measurement	$\pm 2\%$
Duration of Measurement	≥ 30 minutes
Sample Rate	≥ 50 Hz

D. Design process

The design process started with a study of the applicability and feasibility of the previously reviewed thrust measurement methods.

While the gas generator and swinging probe methods were assumed to have the best potential for accuracy, these would need the most complex instrumentation. For the gas generator method, pressure and temperature probes would have to be installed within the engine itself. Thus, engine frame disassembly and modification would be required. This was not deemed possible with the available resources. In comparison, the implementation of the swinging probe (or rake) method, which uses sensors outside the engine, would be possible, but the high cost of temperature-resistant components was considered a disadvantage.

As the turbine was mounted on a pylon with four aluminium supports connected to two carbon frames, the trunnion thrust method was chosen. This convenient engine mount would allow for more degrees of freedom to implement the force sensor, and the symmetry of the assembly would make the load paths clearly defined. Additionally, no engine modification would be necessary, and the sensor was expected to not significantly influence the aerodynamics of the aircraft.

The chosen method was implemented in two concepts described in the following sections.

1. Initial concept

The initial concept was based on substituting half of the engine's aluminium support brackets with load cells (Fig. 3)[14]. The idea behind the concept was to achieve sufficiently low stiffness on the load cells for higher measurement accuracy while the bulk of the load would be carried through the original attachments. Thus, FEM analysis was performed to demonstrate that the forces at the sensors would be within their rated range and that the system would have sufficient stiffness for safety.

The system was tested with the calibration methods described in section IV. During the calibration, it became clear that the system did not deliver satisfactory accuracy and reliability. Two main issues were identified.

Firstly, the high temperatures from the jet engine had an influence on the load cell performance despite their rating for temperature compensation. Such effect was, similarly to the design in Martinez [6], due the thermal expansion of the cage that led to additional lateral forces interfering with the measurements. Stiffening the attachment points by additional longitudinal elements, as well as including heat-isolating spacers in between the heated steel cage and the load cells improved the results.

Secondly, the concept did not provide repeatable measurements. During calibration, it was observed that the measurement system had hysteresis loops. It was contemplated that the load paths would change when high thrust values are applied. Therefore, further improvements of the system were needed.

2. Final concept

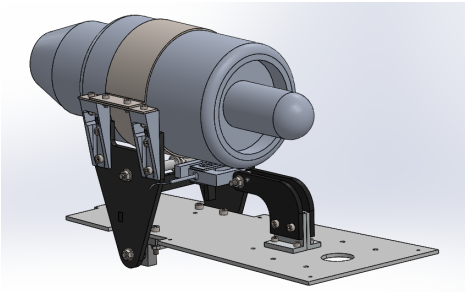
The issues encountered with the initial concept were addressed with two core approaches. First, the point of measurement was centralized and placed in the aircraft's symmetry plane. This decoupled the effects of the engine's cage thermal expansion from the measurements. Second, the structure was made statically determinate in that plane to reduce the number of unknown support reactions and to provide a clear definition of the loads measured.

The simplest solution for such structure would be a single-point, clamp-style attachment with the load cell placed between the engine mount and the rest of the aircraft. However, this configuration would have both the reaction moment over the aircraft's pitch axis and the reaction force in the longitudinal axis influencing the measured output. Therefore,

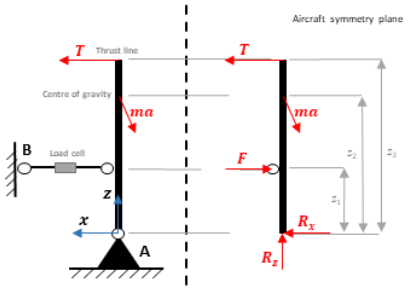
Fig. 3 Previously developed system with half of the usual aluminium support brackets (right) replaced by load cells (left) [14].

to accurately obtain the thrust from the measurements, a multi-axial load cell would be required. It was decided that such a requirement would make the load-cell and its measurement corrections too complex. Therefore, another solution was required.

A solution with two attachment points and a simple single-axis load cell was adopted (Fig. 4a and Fig. 4b). It is, in part, similar to the design presented by Simavilla [7] as the engine mount was also assembled on a freely rotating hinge. The main difference is that the found solution would use a single centrally placed load cell instead of two mounted on the sides of the engine. Moreover, the sensor would be installed within a rod with heim joints on both ends. This constitutes a support that transmits reactions only in the longitudinal direction, corresponding to the free body diagram shown in Fig. 4b.



(a) CAD model of the final concept of the thrust measurement system.



(b) Free-body diagram of the final concept in an accelerating state.

Fig. 4 Final version of the thrust measurement system.

Two moments had to be balanced by the load cell to retain equilibrium. The main moment is due to the thrust vector created by the engine. The secondary moment that was considered was the one created by the accelerating or decelerating hinged assembly. Therefore, the moment balance equations are:

$$\sum M_A = 0 \tag{1}$$

$$-Fz_1 - ma_x z_2 + ma_z x_2 + Tz_3 = 0 \tag{2}$$

Here, F is the force measured by the load cell, m is the mass of the hinged assembly (the mass of fuel was ignored), a_x and a_z are the vertical and longitudinal accelerations of the hinged assembly and T is the thrust. z_1, z_2 and z_3 are vertical distances from the hinge point A and x_2 is the horizontal distance of the CG from hinge line. As its value is very small, it is ignored in Fig. 4b.

It was assumed that the deflections of the load cell are small and that the engine assembly and the aircraft fuselage can be treated as a single rigid body. Therefore, the accelerations measured by the main inertial measurement unit were transformed into the coordinate system of the CG of the hinged assembly and used for further corrections.

To further verify the rigid body assumption and to verify the direction of thrust vector, deflections of a loaded final system around the hinge axis were measured with a 3D scanner. Initially, the deflection angle at a maximum load was found to be about three degrees. It was also observed that part of this deflection comes from the deformation of the base plate. Consequently, the base plate was reinforced with longitudinal stringers. The improvement of the structure reduced the deflection angle to one degree at maximum load. As most of the measurements would take place during steady flight with medium thrust setting, no further corrections were made for the deformation.

3. Selection of the load cell

Choosing the load cell was an integral point for the mechanical design of the system. The design variables like load cell capacity, the location of the hinge axis, the distance of the thrust line from the hinge axis and location of the load cell all influence each other. Therefore, an approach to manipulate these variables with the overall goal to reduce possible system error was adopted.

Due to their good accuracy and low drift characteristics, strain-gauge based load cells were assumed the most appropriate for the design. Initially, bending beam, s-beam and single point load cells were considered. In the end, mainly due to their shape and mounting possibilities within our existing propulsion rack, s-bend load cell was chosen. Options from four manufacturers were analysed, focusing on features such as thread size, thermal compensation, load rating and linearity. Table 3 lists some of the analyzed models and features.

Table 3 Load cell options

Model	Non-Linearity	Temperature	Thread	Capacities up to 1kN
InterfaceForce SSM	0.05 % F_{nom}	$-15 - 65°C$	M6	200,500,700,1000 N
HBM S2M	0.02 % F_{nom}	$-10 - 45°C$	M8	10,20,50,100, 200,500,1000 N
Althen / TE FN9620	0.05 % F_{nom}	$-10 - 45°C$	M12	500, 1000 N
ME-Systeme KD80se	0.05 % F_{nom}	$-10 - 70°C$	M8	100,200,500,1000N

The model InterfaceForce SSM[15] offered most rating options between 0.5 and 1kN, which was the estimated required range. It was also among the two with the highest temperature compensation range while maintaining similar linearity characteristics to most others. Also, the smaller thread size allowed for smaller and lighter components which helped reduce overall structure weight.

To estimate errors of different configurations, a design algorithm was used. The following design inputs were taken into account:

- Allowable distance between the hinge axis and the thrust line
- Allowable distance between the load cell mounting point and the thrust line
- Accuracy data for the chosen load cell model as provided by the manufacturer
- List of available capacities of the load cell of the chosen model
- Desired safety margin between maximum force and load cell capacity
- Maximum thrust force value that the system should measure
- Thrust force value at which error estimation shall be performed
- Maximum expected temperature of the load cell during operation

• Minimum expected temperature of the load cell during operation

As a result, the measurement error was calculated:

$$\begin{aligned} < \epsilon_{abs} >= (< \epsilon_{nl} > + < \epsilon_{hyst} > + < \epsilon_{T0} >< \delta T >) < R > \\ +(< \epsilon_{rep} > + < \epsilon_{cr} > + < \epsilon_{T1} >< \delta T >) < F > \end{aligned} \tag{3}$$

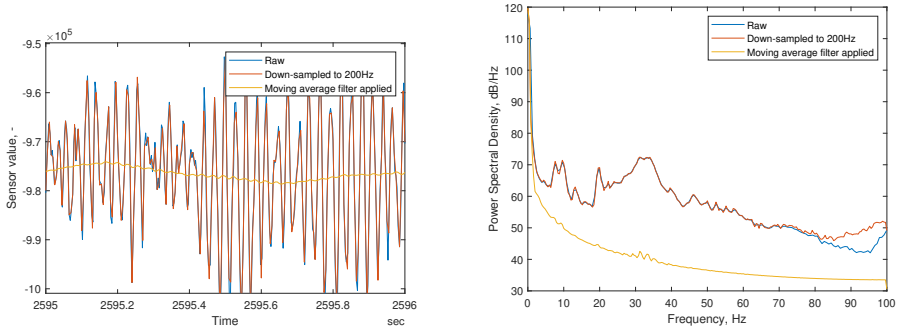
Here ϵ are errors due to non-linearity (ϵ_{nl}), hysteresis (ϵ_{hyst}), temperature influence on zero-load value (ϵ_{T0}) and output value (ϵ_{T1}), repeatability (ϵ_{rep}) and creep(ϵ_{cr}). δT is the temperature difference, R is the rating of the load cell and F is the applied load. Resulting value ϵ_{abs} is the absolute measurement error.

In the end, a load cell with $700N$ rating was chosen. The resulting positioning offered enough clearance for all existing components and cabling, reduced the maximum forces on the rest of the structure and allowed easier assembly than if the load cell was placed closer to the engine. Additionally, the gap in between the engine and the load cell reduced the possibility of temperature influence from the engine.

4. Logging system and signal processing

The data acquisition system consists a model NAU7802 amplifier integrated with an analog-to-digital converter (ADC). This device provides an excitation voltage of 4.5V to the load cell and has a maximum sampling rate of 320 samples per second. The amplifier is connected via I2C bus to Arduino Nano, which is used to convert the data from I2C to USB and then sent to a Raspberry Pi where it is logged. The created log file is not synchronized to the main flight log automatically, therefore synchronization is done manually by cross-correlating the aircraft pitch angle and measurements by the load cell during a pre-flight calibration.

The raw signal logged from the load cell had to be processed. First, the data had to be down-sampled using spline interpolation to fit the flight log from the main on-board computer which had a 200Hz sampling rate. Then, building on the assumption that the engine thrust has a slow dynamics, the moving average filter was applied with a window size of 100 samples. This process is visually presented in Fig. 5 in both time and frequency domains. At the time of writing the nature of the oscillations of the raw signal are not yet investigated.



(a) Signal processing applied on flight data during a steady flight segment.

(b) Welch's power spectral density estimate graph of the complete flight during the signal processing steps.

Fig. 5 Process of processing the raw signal from the load cell.

The smoothing was also performed on the accelerations, which were required to extract the thrust. At that point the sensor signal was converted into physical units.

For further information on thrust measurement system design, please consult Fleig [16] and Metzner [17].

IV. Calibration of the system

A calibration was needed to convert the raw sensor measurement into physical units. Calibration with weights was used as the main laboratory calibration environment. Furthermore, two methods to check and confirm a valid calibration

were applied: calibration in the static propulsion test stand with a running engine and checking the calibration just before the flight by tilting the aircraft in engine-off condition.

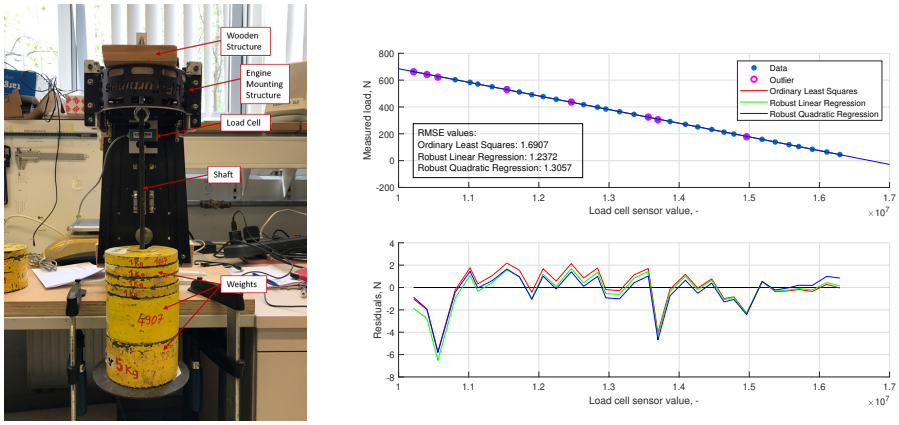
A. Calibration with weights

For the calibration with weight blocks the system was mounted vertically. A wooden plate with a hook was installed as shown in Fig. 6a. The weight blocks were hung on the hook and weight force, acting at the thrust line of the engine, was measured. In order not to damage the load cell, weights from $0N$ to $350N$ were used.

Weight of the engine mount assembly was taken into account during calibration. The weight force was assumed to act straight down.

The weights in steps of $1kg$ or $5kg$ were applied in a random fashion. It was allowed for system to stabilize for two minutes and then a measurement of twenty seconds was taken.

Results were analyzed using Eq. 2 and plotted (Fig. 6b). Three fit models were tested: ordinary least squares, robust linear regression (both of form $y(x) = bx + c$) and robust quadratic regression ($y(x) = ax^2 + bx + c$). Here x is the sensor value and y is the load, measured by the load cell in Newtons. In robust regression methods the outliers were identified using a bisquare weighting function and excluded from fit. This was assumed a valid step as no discontinuities in the calibration were expected.



(a) Thrust measurement system during calibration with weights. **(b) Calibration curve of the load cell (top). Residuals for the different fit models (bottom).**

Fig. 6 Calibration of the thrust measurement system with weights.

All three methods provided accurate results, with robust linear fit having the lowest root-mean-square error of $RMSE = 1.24N$. Even though the 8 identified outliers were hardly visible in the calibration curve, the residuals of those test points vary deviate more than the norm, ranging between $2N$ and $6N$. No considerable benefit of using a quadratic fit was found. The coefficients of the different models are presented in Table 4.

Table 4 Comparison of three fitting methods to extract the calibration curve for the system. Coefficients of $y(x) = ax^2 + bx + c$ are summarised. Here x is the sensor value and y is the load, measured by the load cell in Newtons.

	RMSE, N	a	b	c	Outliers
Ordinary Least Squares:	1.69	0	-1.017e-04	1702	Included
Robust Linear Regression:	1.24	0	-1.019e-04	1705	Excluded
Robust Quadratic Regression	1.31	-1.347e-13	-9.826e-05	1681	Excluded

Note that due to the difference of moment arms in between the load cell and the thrust line, the calibration errors for the load cell would have to be almost halved, resulting in $RMSE = 0.64N$.

B. Calibration check on a static test stand

The calibration of the system was further validated with static thrust tests. The goal was to confirm the reliability of the of the calibration and investigate any temperature or vibration effects that may rise due to the jet engine.

The thrust measurement system was mounted on the turbine test stand of the Institute of Aircraft Design (Fig. 7). The stand is a Kistler dynamometer of model 9366cc with amplifiers of model 5073A. It is operated via Labview software. A thrust schedule similar to the flight profile was applied. The recorded data had to be manually correlated in time, as no synchronization in between the two logging systems was available.

A six minute test was done with a profile similar to flight. The test stand was tared just before the measurements. Note that the test on the test stand was done before mounting the longitudinal stringers to stiffen the base, whereas the calibration curve applied to the measurements was extracted from measurements with the stringers.

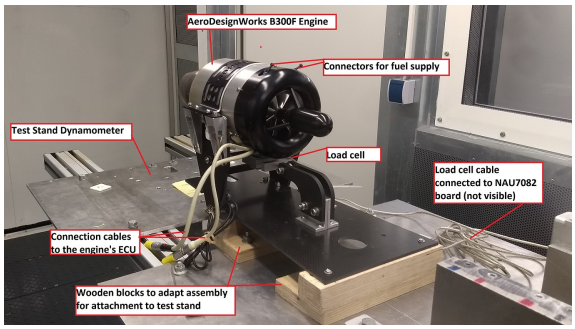


Fig. 7 Thrust measurement system mounted on a static propulsion test stand during calibration check.

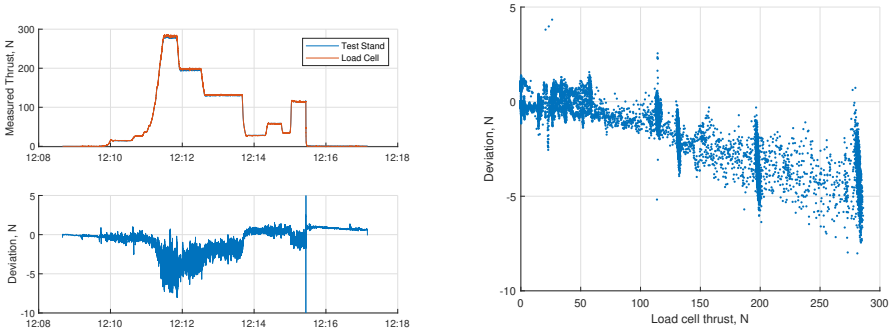
Good correlation between the two measurement methods was observed (Fig. 8a). All changes of throttle were well recorded. However, the deviation between the two measurements varied higher than expected. For low and medium throttle settings the deviation stayed within the $2N$ (Fig. 8b). For high throttle settings the deviation increased to $8N$. The overall error in measured thrust varied within 3 percent.

The higher than expected error was attributed to multiple factors. Firstly, the static load stand was known to have high drift. This can be noticed in Fig. 8a, where after the measurement the thrust value, logged by the static stand, was not zero. The measurements were not corrected for the drift effects. Secondly, even though care was taken to mount the engine as level as possible, there was no system in place to confirm that the thrust axis is parallel to the axis of the measurement stand. The vertical measurements of the stand were not taken into account. This could have resulted in noticeable deviations, as the base of the engine stand was not yet reinforced and would bend. Additionally, temperature of the engine or the load cell was not measured during the test run. Therefore, it was not possible to tell if the deviation at higher thrust levels arise due to temperature. However, as the goal of the measurements with the static test stand was not to update the calibration, but to confirm it, the resulting deviations were interpreted as acceptable.

C. On-site calibration check

As the aircraft gets assembled and disassembled multiple times during a flight season [11] it was desired to have a confirmation that the system is still calibrated (especially the constant offset) right before each flight. It was speculated, that while the curve slope of the load cell would not change (as this would mean damage to the load cell), the constant offset might change slightly due to the surrounding features of the assembly. Therefore, another procedure for on-site calibration check was developed.

The procedure included tilting and rotating the assembled aircraft and use self-weight of the engine assembly together with the inertial measurements of the aircraft to compare the applied and measured load. The aircraft would be tilted nose down and nose up to apply longitudinal load on the load cell. It was then banked left and right to confirm the



(a) Test profile and deviation in between the two methods.

(b) Deviation in between the two methods for different applied thrust.

Fig. 8 Comparison of the thrust measurement received via calibrated thrust measurement system and a static thrust measurement stand.

negligible (or measurable) influence of the roll angle on the measurement system. During the procedure, same equation 2 was used to calculate the applied load.

The applied and measured loads are displayed in Fig. 9. The sum of the two signals, which in this case should be equal to zero, has a maximum amplitude of $2.6N$. This, converted into an error of measured thrust as described in section IV.A would result in $1.3N$.

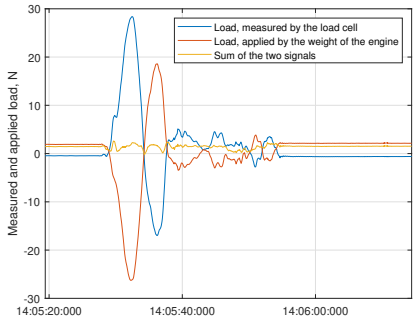


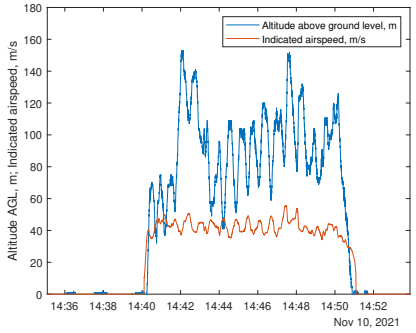
Fig. 9 Applied and measured load during the on-site calibration check. The sum of the signals displays the effect of how the thrust measurement system compensates for changes in acceleration of the engine assembly.

The method has to be taken with care. Due to the dimensions of the demonstrator, the pitch angle is limited to roughly $0.4rad$, resulting in maximum applied load of around $30N$. In comparison, $100 - 150N$ range was assumed for steady level flight during a mission. Additionally, the load cell has a maximum load capacity of $700N$. Consequently it was assumed that the errors during the tilting procedure are of acceptable magnitude.

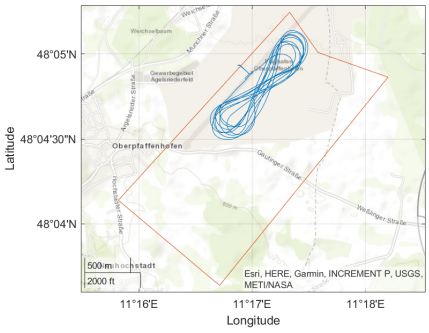
V. In-flight results

The system has been tested in-flight. Up to date, two test flights were done with the system. Results from one of them will are presented.

The flight profile with altitude and velocity is shown in Fig. 10a. The flight trajectory can be found in Fig. 10b. The flight was done on 10th of November, 2021 at Special Airport Oberpfaffenhofen (EDMO) in Germany. The main goal of the flight was to further extend the functionality of the autopilot, for which further details about the engine were needed. Therefore, many manually executed engine step inputs were done. Additionally, two test legs with different airbrake settings were tested, as well as different flap settings for take-off and landing.



(a) Flight profile.



(b) Deviation in between the two methods for different applied thrust.

Fig. 10 Comparison of the thrust measurement received via calibrated thrust measurement system and a static thrust measurement stand.

Thrust was logged throughout the whole flight, including the engine start-up phase. Measured and modelled thrust is compared in Figure 11. For this comparison, thrust model based on engine revolutions, Mach number and altitude, developed during a different project was used [18]. Even though the two methods agree well at very low thrust values, an almost constant offset of $10N$ is seen during the rest of the flight. One reason for this might be that the engine model does not take the ambient temperature into account.

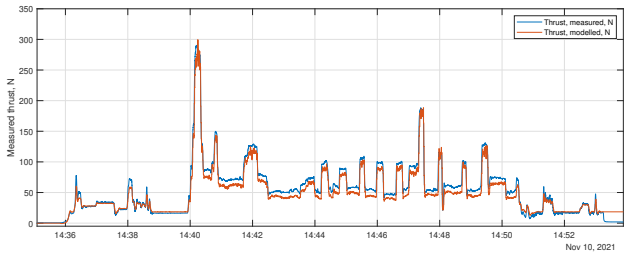


Fig. 11 Thrust measured during flight. For comparison, thrust modeled with respect to engine revolutions, Mach number and altitude is added[18].

An extract of two throttle step inputs is shown in Figure 12. The lower sampling frequency of the engine revolutions, in comparison to the rest of the flight variables, can be noted. However, the thrust measurement system does follow changes in the engine spool speed well.

After reviewing the measured thrust, some trends of the system could not yet be explained. During the moments of high yaw rates, the system tend to have jumps in logged thrust, as can be seen in Figure 12. Even though the yaw rate is accounted for when changing the coordinate system of accelerations from the aircraft to the engine mount assembly, there still seems to be an unexplained component that influences the final measurement. Another unexplained increase

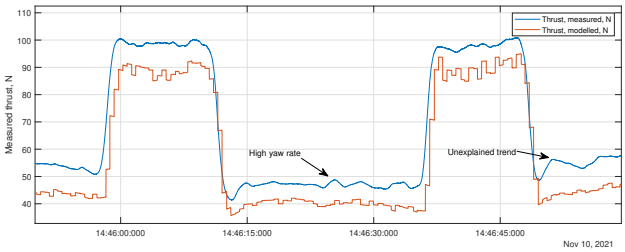


Fig. 12 Measured thrust during an throttle step input.

in measured thrust is also marked. Both of these trends seem to appear only in highly unsteady motion. Further investigation for the cause will follow.

VI. Thrust modelling

It was desired to retrospectively model thrust for the T-FLEX flights before the thrust measurement system was installed. For this reason, an update to the engine model was made, building on the data collected during the flight on 10th of November, 2021.

Data from half of the in-air time was extracted to derive the engine model. It was postulated that a thrust model based on the available engine throttle setting (actual vs maximum revolutions of the engine), ambient pressure, stagnation temperature and density ratios, burner temperature ratio and the Mach number could be derived:

$$F_T = f(\delta F_T, p/p_0, T_{stag}/T_0, \rho/\rho_0, T_{burner}/T_{stag}, M) \tag{4}$$

Linear stepwise regression was used to derive the engine model from the sample data. The method adds each of the possible terms based on their statistical significance to explain the modelled variable. If in the end the significance of a term is too low, the term is removed. Quadratic dependencies, as well as dependencies on multiplicands of lower-order terms was allowed. Normalized thrust was used as the response variable. The resulting model is described in Eq. 5.

$$F_T/F_{max} = \beta_1 + \beta_2 \delta F_T * T_{stag}/T_0 + \beta_3 \delta F_T * M + \beta_4 \delta F_T^2 + \beta_5 T_{stag}/T_0^2 + \beta_6 M^2 \tag{5}$$

The coefficients of the separate terms are given in Table**??**. The data used for modeling is presented in Fig.13a and the data used for validation of the model in Fig.13b. Model fit was determined as appropriate with $RSME = 2.77N$.

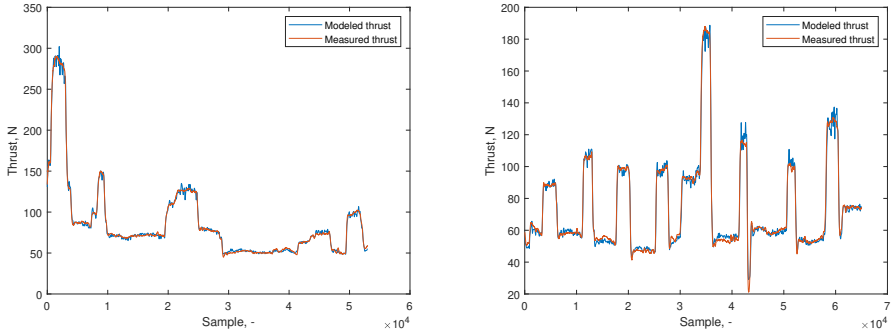
VII. Conclusion

A thrust measurement system was developed for a pylon-mounted miniature jet engine. The system was built and calibrated. During the static calibration with weights, accuracy of $0.64N$ was achieved (in terms of simulated thrust). Higher deviations were noted during the calibration check on the static test stand and during an on-site calibration check. However, due to the nature of the two calibration check methods, their accuracy was considered to be lower and the deviations from weight calibration results are to be taken with care. Nevertheless, good response of the thrust measurement system was confirmed under operational conditions with the running engine.

The system was also tested in-flight. The system measurements did not present any unexpected values during the steady flight. Some minor unexplained deviations were spotted during the unsteady segments. These, however, remained in the range of $2-5N$. Data gathered was seen appropriate to retrospectively model the thrust for the flights where no thrust measurement system was available.

To author's knowledge, this is the first time that thrust has been measured in-flight with high accuracy for a medium-sized unmanned aircraft.

Further investigation is needed to evaluate nature of raw signal oscillations and the deviations during unsteady parts of the flight. Additionally, another static test run should be done with the reinforced system to check the deviations



(a) Sample data used to create a thrust model of the engine. **(b) Validation of the created model.**

Fig. 13 Thrust modeling and validation from flight test data.

during high-thrust segments. Finally, the same mounting concept could be tested with an electric propeller engine to see if the high vibrations, observed by other authors, have the same influence on the thrust measurements.

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. The author J.B. would like to thank Mr. Pedro Alexandre Tonet Fleig, who designed, implemented and tested the measurement system, as well as Ms. Annina Metzner, who did further testing and investigation on the data. Additionally, J.B. would like to thank the rest of the T-FLEX flight test team for their contributions during flight tests with special thanks to Mr. Lars Nagel and Mr. Daniel Teubl who improved the logging system for the thrust measurements.

References

[1] Nguyen, N., Ting, E., and Lebofsky, S., "Aeroelastic analysis of a flexiblewingwind tunnel model with variable camber continuous trailing edge flap design," *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2015, pp. 1–28.

[2] MIDAP Study Group, *Guide to In-Flight Thrust Measurement of Turbojets and Fan Engines*, ADVISORY GROUP FOR AEROSPACE RESEARCH AND DEVELOPMENT, 1979. https://doi.org/AGARD-AG-237.

[3] Davidson, T. W., "Measurement of net thrust in flight," *Journal of Aircraft*, Vol. 1, No. 3, 1964, pp. 107–113. https://doi.org/10.2514/3.43575.

[4] Conners, T. R., and Sims, R. L., "Full flight envelope direct thrust measurement on a supersonic aircraft," *34th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*, Vol. 1998, No. July, 1998. https://doi.org/10.2514/6.1998-3872.

[5] Muhammad, H., Kuntjoro, W., and Sritjahjono, B. E., "In-Flight Thrust Determination by Load Measurement on the Engine Mounting System," *Proceedings of the 22nd International Congress of Aeronautical Sciences*, 2000, pp. 1–7.

[6] Martinez, A., "Design and manufacturing of a thrust measurement system for a micro jet engine," Master thesis, unpublished, Linköping University, 2018.

[7] Simavilla, E.-A. P., "Entwicklung und Implementierung eines on-board Schub- messsystems für UAV-Anwendungen Development and implementation of an on-board thrust measu- rement system for UAV applications," Master thesis, Technical University of Munich, 2020.

[8] Bronz, M., de Marina, H. G., and Hattenberger, G., "In-flight thrust measurement using on-board force sensor," *AIAA Atmospheric Flight Mechanics Conference, 2017*, 2017. https://doi.org/10.2514/6.2017-0698, URL https://hal-enac.archives-ouvertes.fr/hal-01432789.

[9] Sartori, D., and Yu, W., "Experimental Characterization of a Propulsion System for Multi-rotor UAVs," *Journal of Intelligent and Robotic Systems: Theory and Applications*, Vol. 96, No. 3-4, 2019, pp. 529–540. https://doi.org/10.1007/s10846-019-00995-2.

[10] Bergmann, D. P., Denzel, J., Pfeifle, O., Notter, S., Fichter, W., and Strohmayer, A., "In-flight lift and drag estimation of an unmanned propeller-driven aircraft," *Aerospace*, Vol. 8, No. 2, 2021, pp. 1–16. https://doi.org/10.3390/aerospace8020043.

[11] Bartasevicius, J., Koeberle, S. J., Teubl, D., Roessler, C., and Hornung, M., "FLIGHT TESTING OF 65KG T-FLEX SUBSCALE DEMONSTRATOR," *32nd Congress of the International Council of the Aeronautical Sciences*, Shanghai, China, 2021, pp. 1–16.

[12] Sendner, F. M., Stahl, P., Rößler, C., and Hornung, M., "Designing an UAV propulsion system for dedicated acceleration and deceleration requirements," *17th AIAA Aviation Technology, Integration, and Operations Conference, 2017*, , No. June, 2017, pp. 1–13. https://doi.org/10.2514/6.2017-4105.

[13] Bauer, P., Anastasopoulos, L., Sendner, F. M., Hornung, M., and Vanek, B., "Identification and Modeling of the Airbrake of an Experimental Unmanned Aircraft," *Journal of Intelligent and Robotic Systems: Theory and Applications*, Vol. 100, No. 1, 2020, pp. 259–287. https://doi.org/10.1007/s10846-020-01204-1.

[14] Koch, J., "Design and Development of a Jet Engine Thrust Measurement System for in-Flight Applications," Semester thesis, unpublished, Technical University of Munich, 2020.

[15] "Interfaceforce Series SSM/SSM2 Technical Data," , 2020. URL www.interfaceforce.de.

[16] Fleig, P. A. T., "Improvement and Further Design of a Thrust Measurement System for In-Flight Applica- tions on an Unmanned Aerial Vehicle," Master thesis, Technical University of Munich, 2020.

[17] Metzner, A., "Calibration and Testing of an In-Flight Thrust Measurement System for UAV Applications," Bachelor thesis, Technical University of Munich, 2021.

[18] Bougas, L., Rößler, C., and Hornung, M., "Design and experimental validation of a propulsion duct for a jet propelled low observable scaled UAV demonstrator," *17th AIAA Aviation Technology, Integration, and Operations Conference, 2017*, 2017, pp. 1–10. https://doi.org/10.2514/6.2017-4378.

7 Optical wing shape tracking

Methodology to measure wing deformations in-flight was researched in a Master Thesis by Mr. Pablo Varillas Iglesias at TUM (guided by Julius Bartasevicius). The relevant sections are adjusted and included in this section.

7.1 Development and implementation

The used hardware for implementing the wing shape measurement system is presented in section 7.1.1. In section 7.1.2 the software development and implementation is presented.

7.1.1 Hardware

The hardware used in the implementation of this wing shape measurement were the FLEXOP UAV T-FLEX and the two rear cameras. The two cameras are the central part of the hardware. The model is the Mobius HD Action Camera [7]. Its principal characteristics are the lightweight and small design. These features make the Mobius cameras a very good option for mounting them on a small UAV. The specifications of the Mobius camera are shown in table 1.

Table 1: Specifications of the Mobius HD Action Camera

Specification	**Value**
Size	61mm (L) x 34 mm (W) x 18 mm (H)
Weight	38 g
Max. resolution	1920x1080
Max. frame rate	60 fps
Battery	820 mAh
Autonomy	approx. 130 min
CMOS area	2304x1296
Resolution-frame config.	1920x1080p@30FPS, 1280x720p@60FPS

The flight test videos were recorded with the configuration 1920x1080p@30FPS. The camera saves the videos in the MOV video format with H.264/AVC1 video codec. Different modes such as time-lapse, photo, and video are available. Support for MicroSD Card up to 32 GB is supported [7].

The Mobius camera has only three control buttons to operate all functions. If more information is needed, refer to the instruction manual [6].

The CATIA 3D model of the T-FLEX was used to obtain the target's location in the ABFF. Figure 14 shows the CATIA model of the T-FLEX with the targets 3D coordinates.

The displayed rear coordinates are the position of the Mobius camera in the ABFF. As mentioned before, the Mobius cameras are mounted on the tail of T-FLEX, aiming at the wings. The integration with the T-FLEX fuselage is shown in figure 15. The integration is achieved by a black 3D printed part.

7.1.2 Software

In this section, the development and implementation of the software are illustrated. The wing shape measurement software was programmed entirely in the Python programming language. Additional Matlab scripts were written to plot the obtained results because of Matlab's user-friendly plotting interface. The software was developed on a PC with the operative system (OS) Windows 10 Professional.

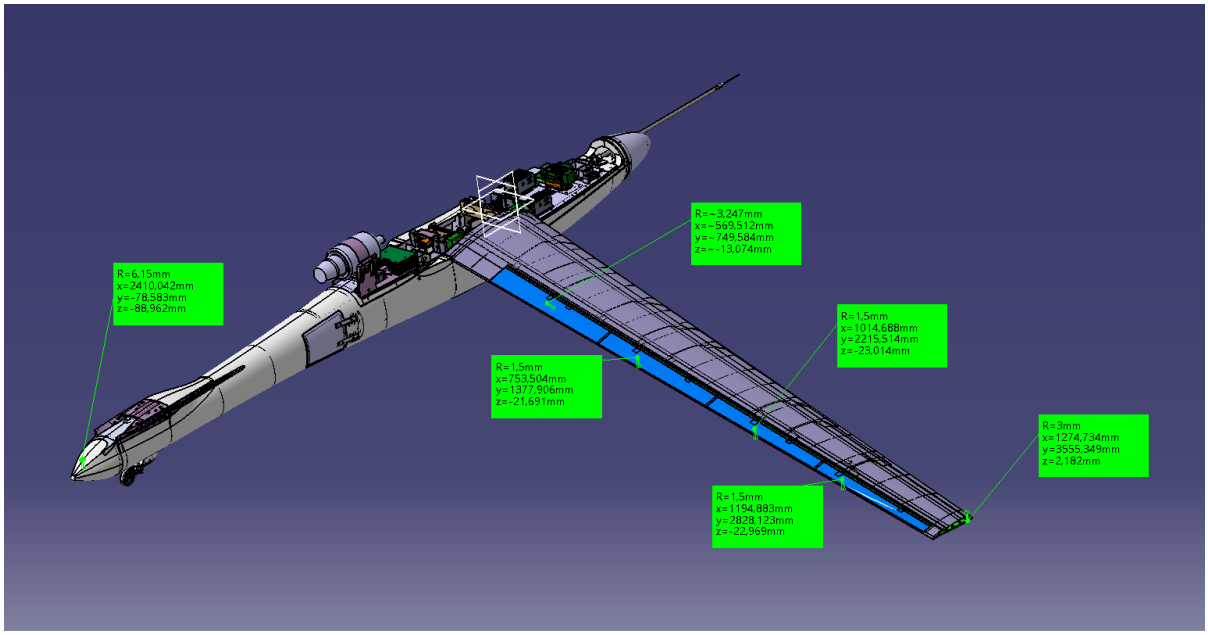


Figure 14: CATIA model of T-FLEX with the targets 3D coordinates

Figure 15: Integration of Mobius camera into the T-FLEX fuselage

Git was used for the version control from the beginning of the development. The Phyton software is hosted in the LRZ GitLab server (*https://gitlab.lrz.de/*).

The selected integrated development environment (IDE) was *Visual Studio Code*. It was chosen because of its friendly user interface and flexibility. This IDE has lots of extensions that can be installed to

facilitate the programming. The installed extension and respective versions are shown in the list below:

- Python v2021.10.1336267007
- Excel Viewer v3.0.44
- Python Docstring Generator v0.5.4

The Python extension was installed for the support of the programming language. The Excel Viewer extension is a visualization tool for csv files. When the software user gets arrays of many rows and columns, it is essential to have a tool to visualize them since this IDE does not have it by default. The Python Docstring Generator extension was installed to provide the code with docstrings and descriptions of classes and functions.

Python version 3.8 was used for writing the code because of the compatibility with the OpenCV library. OpenCV is an open-source Computer Vision software library. OpenCV was built to provide a common infrastructure for Computer Vision applications and accelerate machine perception in commercial products. The library has more than 2500 optimized algorithms, which includes a set of both classic and state-of-the-art Computer Vision algorithms, e.g., visual tracking algorithms, camera calibration, and 3D reconstruction [9]. As one can see this is an essential library for the purpose of this master's thesis.

The OpenCV library was written originally in the C++ programming language. However, it can be used in Python code via the OpenCV-Python application programming interface (API). OpenCV-Python is a Python API for OpenCV that combines the best qualities of the OpenCV C++ API and the Python language.

Python virtual environments were used for the development of the software. This way, it was possible to access different configurations of Python packages. The essential packages, which were used in the context of this implementation are shown in the following list:

- numpy v1.20.3
- opencv-contrib-python v4.5.3.56
- pandas v1.3.1
- imutils v0.5.4
- matplotlib v3.4.2

NumPy is a Python library that supports improved computation with arrays and vectors compared to the standard Python library. Having NumPy installed is a requirement for using OpenCV. The selected OpenCV package was the extended one (contributors-package). It has some additional modules that the standard OpenCV package does not have, e.g., the tracker implementations. The pandas package was used for saving and reading csv files. The imutils package has some excellent functionalities when working with OpenCV, like, e.g., an implementation of a FPS counter that is used for measuring the processing speed of the algorithm. Finally, the matplotlib package was used for plotting and visualizing the results.

The methodology followed during the development was to write small scripts with the essential functions and test them. Afterward, new scripts were created integrating all the functions. The followed coding philosophy was function-oriented. This means that only functional programming was used. This has some advantages as one function can be used multiple times by only having to write one extra line in

the code. In addition, a function can be loaded by other scripts so that the code looks cleaner than in other coding philosophies.

The essential scripts that were written are listed below.

- opencv_measurement.py
- camera_calibration.py
- pixel2metric.py
- plot_deflections.py

The target tracking algorithm was implemented in the script opencv_measurement.py. As a result, a csv file was saved so that the other scripts could read it and work with it. The csv file saves an array with twelve columns and many rows. In each row, the five target positions of the actual frame are saved. These are ten columns as the position are two-dimensional. The last two columns are the reference pixel coordinates where the original frame was cropped. In this way, the absolute deflections in the IPCS are obtained by adding the coordinates measured in the cropped frame to the reference value.

The camera calibration algorithm is implemented in camera_calibration.py. The calibration results are saved into binary files that the other scripts can read. The 3D reconstruction approach is implemented in the script pixel2metric.py. Here, the computation of the 3D coordinates from the 2D coordinates is done. In the script plot_deflections.py some nice plots were programmed for the visualization of the results.

Matlab was used for visualizing the results and for the creation of timetable variables from the csv files. The time-stamping algorithm needs the data to be in Matlab's timetable format.

7.2 Test and validation

The software for wing shape measurement was tested both during and at the end of development. Most of the functionalities were individually tested before integration was carried out. In this way, a correct operation of the software after the integration of the functionalities was ensured.

This section describes the tests and presents the results that were obtained during the tests. In addition to the testing of the individual modules, pre-tests on some core functionalities were made. Afterward, tests were also performed on the entire software. The software testing was carried out with videos from T-FLEX UAV test flights from the year 2019.

All the tests and pre-tests were made on a PC with the central processing unit (CPU) from Intel model Core i5-10600K [4]. This processor model has six cores and 12 threads. The clock of the processor runs at $4.10\ GHz$. The PC has $16\ Gb$ of random-access memory (RAM) and a graphics processing unit (GPU) from NVIDIA model GeForce RTX 3060 [1]. The OS of the computer is Windows 10 Professional [5]. The compilation number of the OS is 19043.1288. The system has a 64-bit architecture. Table 2 shows all the PC specifications described above.

The original video data of the Mobius cameras was in MOV format. The cameras are configured to save the videos in the SD-Card whenever the file size of the video reaches approximately $1GB$. This leads to the test flights being split into different video files with MOV format. Usually, a full test flight is split into three or four video files. The solution applied to obtain a video file of the full flight was to attach the multiple video files into one. This was accomplished by using the video editing software Camtasia 2020 [12]. After joining all video files together, the exported video file had an MP4 format. Both MP4- and MOV-format are encoded with the MPEG-4 codec, which is a standard codec in audiovisual files.

Table 2: Specifications of the PC where the tests were performed

Component	**Specification**
.1em.05em.05em CPU	Intel Core i5-10600K @4.10 GHz
RAM	16 GB
GPU	NVIDIA GeForce RTX 3060
OS	Windows 10 Professional compilation 19043.1288
System architecture	64-bit

7.2.1 Pre-tests

In this section, the performed pre-test and their results will be presented. Pre-tests on the stability and speed performance of the OpenCV-trackers were made. Thus, the wings were tested to check which were the best features to track. Camera calibration tests were also performed previous to the final testing of the software. Based on the results, the best performers were implemented in the final software.

As mentioned in section 7.1.2 the OpenCV library has a lot of implementations of Computer Vision functions. Tests to see the individual overall performance of these implementations were performed.

For the pre-tests of the OpenCV-trackers, only a fragment of a test flight was used. The video file is called *REC_0002.MOV*, and it was recorded in the test flight on the 1st of August 2019. The video has an original resolution of 1920x1080p but is cropped to a resolution of 1595x341p. In these pre-tests, two aspects were analyzed. The first is the tracker stability, i.e., how many times the tracker fails. Tracking failure is meant when the BB disappears or drifts away from the target that it is supposed to track. The second aspect is the speed performance of the tracker, i.e., how many FPS the tracker can process. A FPS counter was implemented in the software to obtain the mean value of the processed FPS of the video.

This pre-test was done three times at different targets of the wing: wingtip, approximately half of the semispan, and wing's root. The goal of the pre-test was to find the OpenCV-trackers that performed better in this setup. Tables 3 - 5 show the performance of the OpenCV-trackers in the two aspects mentioned above.

Table 3: Pre-test of the OpenCV-trackers: wingtip

OpenCV-tracker	**FPS processing speed**	**Tracking failures**
CSRT-tracker	$\sim$ 27 FPS	2
KCF-tracker	$\sim$ 32 FPS	8
MOSSE-tracker	$\sim$ 45 FPS	4
BOOSTING-tracker	$\sim$ 32 FPS	12
MIL-tracker	$\sim$ 16 FPS	$>$ 30
TLD-tracker	$\sim$ 11 FPS	$>$ 30
MEDIANFLOW-tracker	$\sim$ 32 FPS	$>$ 30

In tables 3 - 5, it is shown that the CSRT-tracker is the most stable implementation in OpenCV, followed by the MOOSE-tracker. The processing speed of the CSRT-tracker is a little bit lower than the average, but the reliability is the best. This characteristic makes the CSRT-tracker a good option for tracking the wing's outer targets. On the other hand, the best implementation in processing speed is the MOSSE-tracker. In addition, this implementation is very stable, matching the CSRT-tracker at the wing's root target. Thus, the MOOSE-tracker recovers on most occasions when a tracking failure happens. The BOOSTING-, MIL-, TLD- and MEDIANFLOW have so many tracking failures that they make them almost unusable. Thus, the processing speeds of the mentioned trackers are not outstanding.

Table 4: Pre-test of the OpenCV-trackers: half of semispan

OpenCV-tracker	**FPS processing speed**	**Tracking failures**
CSRT-tracker	$\sim$ 31 FPS	2
KCF-tracker	$\sim$ 34 FPS	6
MOSSE-tracker	$\sim$ 44 FPS	3
BOOSTING-tracker	$\sim$ 32 FPS	8
MIL-tracker	$\sim$ 18 FPS	$>$ 30
TLD-tracker	$\sim$ 14 FPS	$>$ 30
MEDIANFLOW-tracker	$\sim$ 33 FPS	$>$ 30

Table 5: Pre-test of the OpenCV-trackers: wing's root

OpenCV-tracker	**FPS processing speed**	**Tracking failures**
CSRT-tracker	$\sim$ 30 FPS	2
KCF-tracker	$\sim$ 32 FPS	6
MOSSE-tracker	$\sim$ 45 FPS	2
BOOSTING-tracker	$\sim$ 33 FPS	7
MIL-tracker	$\sim$ 19 FPS	$>$ 30
TLD-tracker	$\sim$ 10 FPS	$>$ 30
MEDIANFLOW-tracker	$\sim$ 31 FPS	$>$ 30

This pre-test shows that the two best implementations for this scenario were the CSRT-tracker, where high stability is required, and the MOSSE-tracker for the inner wing targets, where stability is not a concern and pure speed is needed.

The goal of the next pre-test was to determine which targets should get tracked. For this purpose, the OpenCV function *cv2.goodFeaturesToTrack()* was used to determine strong corners on a frame. The function finds the most prominent corners in the image or in the specified image regions as described in [11]. Those prominent corners should match the best target locations on the wing. Three different background scenes were chosen to test which targets were the best for all background conditions. The same video as in the previous section was used for this pre-test. The maximum corner variable was set to five to detect the five most prominent corners. Figures 16 - 18 show three different frames of a test flight, where the Shi-Tomasi Corner Detector is being applied.

Figure 16: Shi-Tomasi Corner Detector under clouded background

In the figures above, one can observe that according to the Shi-Tomasi Corner Detector, the strongest corners are the wheel at the wingtip and the four servos under the wing. As mentioned above, with three different backgrounds, the corner detector finds the same five spots in all situations.

This pre-test shows that the best targets to track in the target tracking algorithm are the wheel at the wingtip and the four servos under the wing.

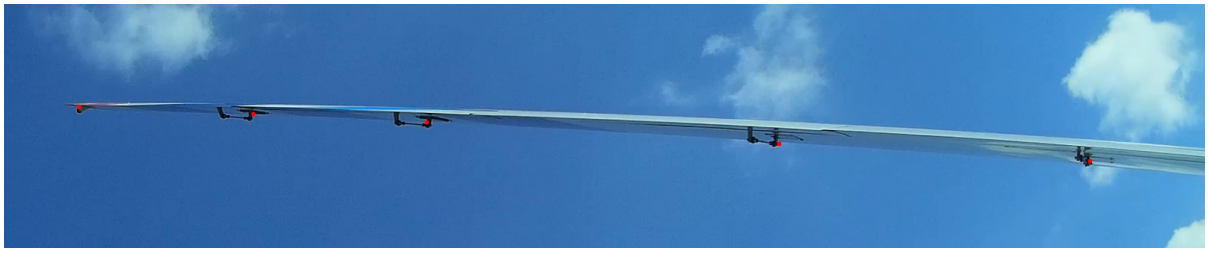


Figure 17: Shi-Tomasi Corner Detector under blue background (sky)

Figure 18: Shi-Tomasi Corner Detector under green background (field)

In the next pre-test, camera calibration of the two Mobius cameras was performed. The Mobius cameras were calibrated separately and marked with 'L' and 'R' to differentiate between the two cameras. The implemented calibration approach is based on the scientific paper by zhang2000. This approach is implemented in the OpenCV library in the function cv2.calibrateCamera().

A checkerboard pattern was printed and attached to a planar surface. 16 images were taken with each Mobius camera as the recommended minimum are approximately 10 images of the pattern [8]. The cameras were at a static position during the calibration. The checkerboard pattern was translated and rotated in each image, ensuring diversity in the images. The square side length was measured and is 27.6mm long. The calibration results are the intrinsic camera matrix, the coefficients of optical lens distortion, and the translation and rotation vectors. Note that the rotation vector can be converted into the rotation matrix by the Rodrigues formula [2]. This method is also implemented in OpenCV in the cv2.Rodrigues() function. Figure 19 shows the 16 images used for the calibration of the left camera.

In the figures above, we can see images from the left Mobius camera after the calibration was applied. One can observe that the feature points were perfectly found. Thus, optical lens distortion is only visible when the checkerboard pattern is near the camera. This is not the case when the cameras aim at the FLEXOP wings as there is a considerable distance between the camera and target points. The assumption of ignoring lens distortion that is made in burner2003 is also applicable in this scenario.

The results were saved as binary files in the npy-format (numpy). Rotation and translation vectors were not saved as the extrinsic parameters must be obtained with the camera mounted on the T-FLEX. The results of both Mobius camera calibration are shown in table 6 and 7.

7.2.2 Tests with videos of full flights

In this chapter, the results with the full flight videos will be discussed. As mentioned above, full flight videos were obtained by attaching the different partial videos with the video editing software Camtasia 2020. In total, we obtained five complete videos of flight tests: three left-wing videos and two right-wing videos. The videos were edited so that the take-off phase starts only a few seconds after the video begins. The videos end a few seconds after the landing is completed and the FLEXOP has stopped. Table 8 shows the five videos and their following characteristics: video name, date, aimed wing, duration

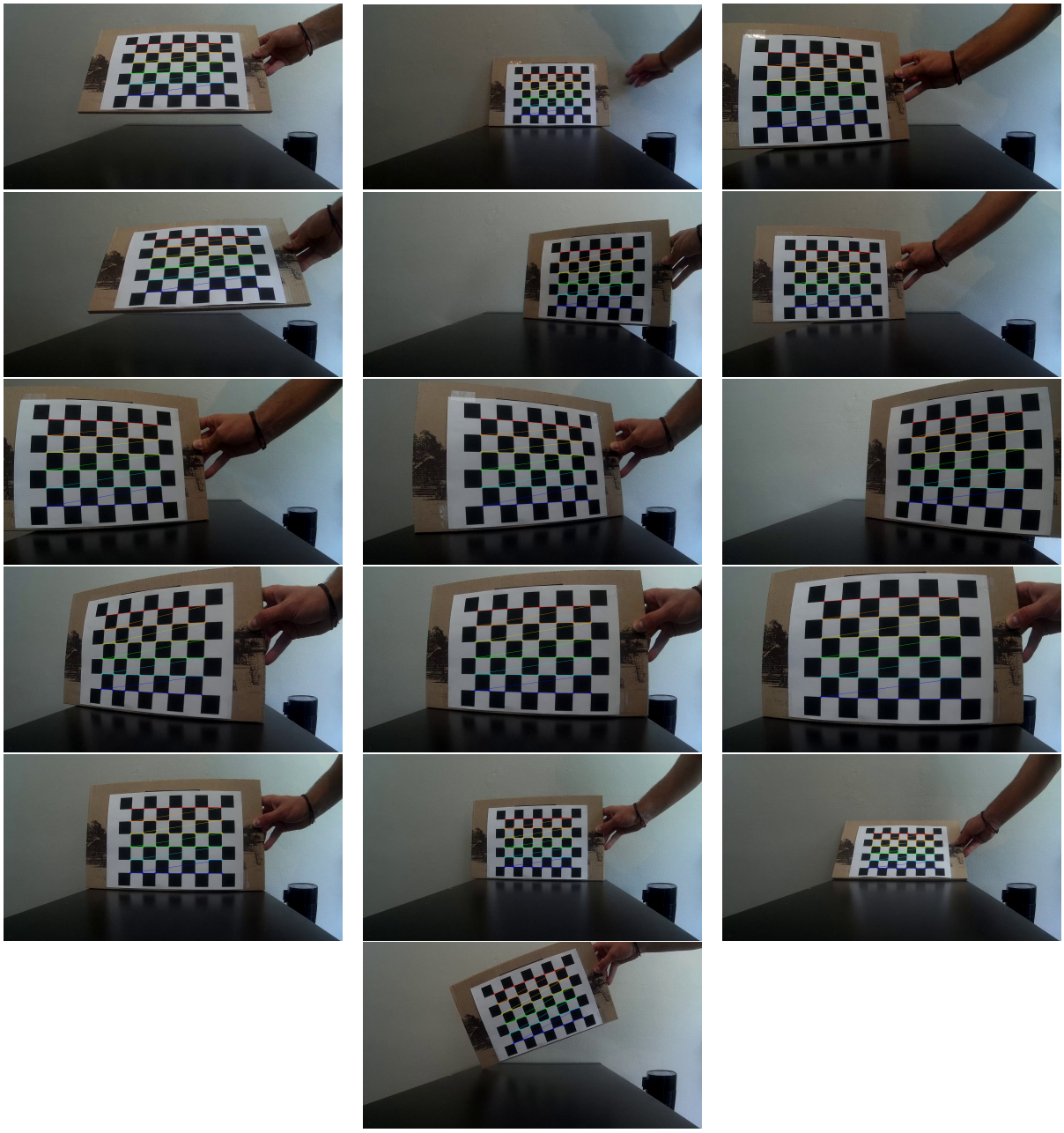


Figure 19: Images used for calibration of the left Mobius camera

of the video, and weather conditions. The videos

The tracking failures in the full videos were tested. These tests were performed using the CSRT- and MOSSE tracker as these two showed the best performance in the pre-tests. The MOSSE-tracker was used for the two inner targets, and the CSRT-tracker was used to track the three outer targets. In this case, tracking failure means that the BB was not inside the AOI defined.This occurs primarily because of the BB disappearing or drifting away from the target. First, the CSRT-tracker stability will be analyzed in the five full videos to see how many times the tracker fails. In figures 6.12 - 6.14, the tracking failures

Table 6: Calibration results: left Mobius camera

Parameter	**Value**
focal length f_x	$1.46292577\,\exp(+03)$
focal length f_y	$1.46683821\,\exp(+03)$
Skew factor γ	0
Principal coordinate c_x	$1.02045320\,\exp(+03)$
Principal coordinate c_x	$5.88698589\,\exp(+02)$
Distortion coefficient k_1	-0.37839949
Distortion coefficient k_2	0.11026849
Distortion coefficient p_1	0.00342506
Distortion coefficient p_2	-0.001392
Distortion coefficient k_3	0.10850981

Table 7: Calibration results: right Mobius camera

Parameter	**Value**
focal length f_x	$1.50592831\,\exp(+03)$
focal length f_y	$1.50702372\,\exp(+03)$
Skew factor γ	0
Principal coordinate c_x	$1.03978361\,\exp(+03)$
Principal coordinate c_x	$5.30131041\,\exp(+02)$
Distortion coefficient k_1	-0.40766303
Distortion coefficient k_2	0.30293375
Distortion coefficient p_1	-0.0026379
Distortion coefficient p_2	-0.00163383
Distortion coefficient k_3	-0.20115594

Table 8: Calibration results: right Mobius camera

Video name	**Date**	**Wing**	**Duration**	**Weather**
.1em.05em.05em *190801_FT1_001_1_01_left.mp4*	01.08.2019	Left	15:08 min	Partially clouded
191106_FT5_001_1_01_left.mp4	06.11.2019	Left	18:10 min	Very clouded
191106_FT5_001_1_01_right.mp4	06.11.2019	Right	18:14 min	Very clouded
191119_FT6_001_1_02_left.mp4	19.11.2019	Left	21:23 min	Clouded
191119_FT6_001_1_02_right.mp4	19.11.2019	Right	21:23 min	Clouded

are shown as red vertical lines. More plots for the rest of the locations are shown in Appendix B. The plots show the vertical wing deflections measured every two frames on all three test flights.

The figures above show an acceptable tracker stability at the target 2. Taking in account the duration of the videos, it is a good performance because little intervention from the user is needed. In the second flight test plot, we can observe more tracking failures than in the rest of the videos. This could be because of this video's light conditions, which are very dark because of the very clouded sky. In addition, the plots of the third flight test show no tracking failure in the right-wing and only one failure at the left-wing.

This is very similar at the location 3 because it is also the CSRT-tracker and the target is very similar. For the location 1 at the wingtip, the stability is not as good as location 2. This could be due to the wheel not being such a strong corner, added to the fact that this location has the fastest and biggest deflections. For both inner targets 4 and 5, many failures are shown in this plot. However, they do not fit the reality accurately. The MOSSE-tracker has many situations where it fails for one or two frames in

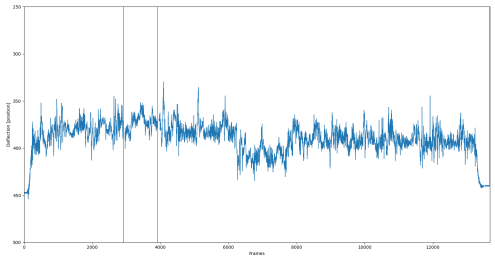
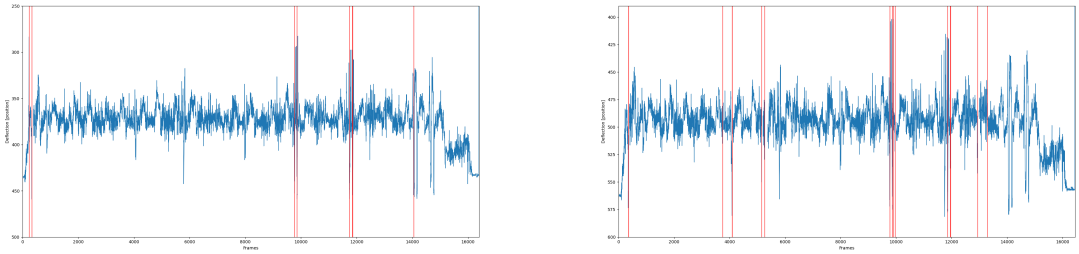
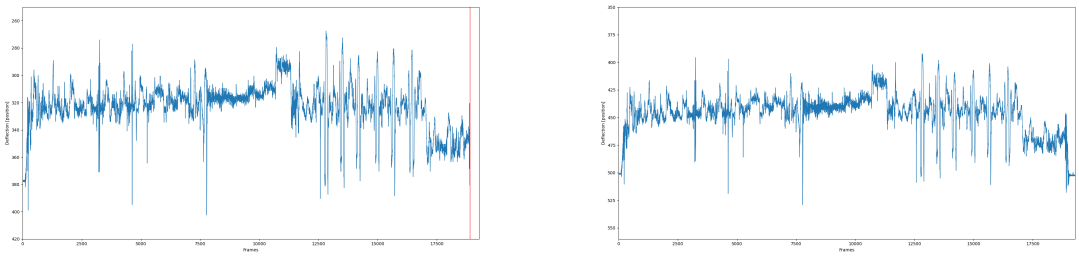


Figure 20: Tracking failures in flight test *190801_FT1_001_1_01* at target 2



(a) Left-wing (b) Right-wing

Figure 21: Tracking failures in flight test *191106_FT5_001_1_01* at target 2



(a) Left-wing (b) Right-wing

Figure 22: Tracking failures in flight test *191119_FT6_001_1_02* at target 2

a row but instantly recovers and continues tracking the correct target. This happens mostly when the video changes the predominant color, e.g., in fast maneuvers of the FLEXOP rapidly.

Another aspect is that the graph's pattern is very similar, comparing left-wing and right-wing. This is also positive because it gives credibility to our target tracking algorithm. The trend of the wing deflections is, in both cases, very similar.

7.3 Summary and outlook

In this section, the design and implementation of an in-flight wing deflection measurement system was presented. The state-of-the-art of science and technology in this field was presented. The three main fields of research are target tracking, camera calibration, and 3D reconstruction. After an overview of the topics, the theoretical foundations were explained. The critical mathematical concepts concerning this thesis were illustrated. Afterward, the developed algorithm for the wing deflection measurement was explained in detail. The development and implementation of the proposed system were presented. The main code was written in Python using the functions of the OpenCV library. Finally, pre-tests and tests and their results were discussed.

The tests showed that the best location for the target tracking was the wheel at the wingtip and the four servos under the T-FLEX wings. Testing also showed that the best OpenCV-trackers for the setup were the MOSSE-tracker and CSRT-tracker. The results showed a high-speed performance of the MOSSE-tracker and also good stability for the T-FLEX setup. The CSRT-tracker performed the best in stability terms. Also, the mentioned tracker performed nicely in the processing speed. The tests also showed good stability in the full flight videos. Locations 2 and 3 showed the best stability. In some of the analyzed videos, these two locations did not have any tracking failure.

A modification that could improve the proposed system would be to change the rear Mobius cameras for cameras with higher resolution and higher frame rate. This would lead to better colors, better contrast, and more fluidity in the recorded videos. These aspects could benefit the target tracking algorithm making it less prone to tracking failures. In addition, the software user could also benefit from these modifications because the target tracking algorithm would be even more automated and would not need so much reinitialization from the user.

Another improvement to the measurement system would be to add a 360-camera to record a 360-video of the test flights. In this work, research was carried out to mount a camera on the top of the fuselage. Figure 23 shows the most promising spot for mounting the 360 camera.

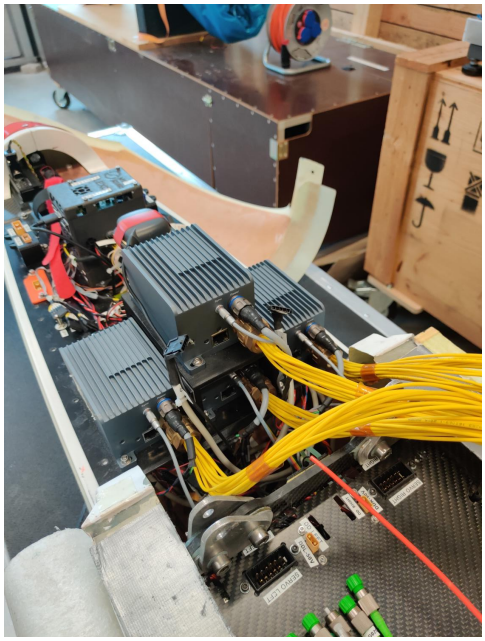


Figure 23: Location of a good spot for mounting the 360-camera

This location was most appropriate because the black case under the FBG module can be easily modified and changed as it is 3D-printed. The idea was to extend the black case so that the 360-camera could sit on top. In addition, pre-tests were made to check if the location recorded both wings. The position met the requirements, so it was decided it would be the best location for the 360-camera.

The model of the camera is the Insta360 ONE X. A 3D model of the camera was made in CATIA V5 based on the camera blueprints published in the camera's webpage [3]. In addition to the 3D model, two other parts were designed in CATIA; the case of the Insta360 camera and the case's support. The following list shows the names of the designed 3D parts:

- *insta360_case_FLEXOP.CATPart*
- *insta360_ONE_X_camera.CATPart*
- *insta360_supoport_case_FLEXOP.CATPart*
- *insta360_support-case-camera_FLEXOP.CATProduct*
- *insta360_support-case_FLEXOP.CATProduct*

Adding a 360-camera to the system could improve the quality of the 3D reconstruction approach as stereo-vision methods often have better accuracy. On the other hand, the algorithm would become more complex as two images would need to be analyzed to do the 3D reconstruction. In addition, the 360-camera would need to be calibrated, increasing the complexity of the system. Calibrating 360-cameras can be more complicated due to the high optical lens distortion caused by ultra-wide-angle lenses. However, work has been done in the field of omnidirectional camera calibration like e.g., [10]. An implementation of the 360-cameras was not carried out due to not having enough time.

The calibration from camera to physical coordinate system was not achieved. It will be further investigated if the method is worth pursuing.

8 Actuator diagnostics

8.1 RC servos

The SHM (Servo Health Measurement) unit will not be sufficient for all of our purposes, because of its noise, resolution and dependency of servo on temperature.

The first objective of temperature sensor is to detect and prevent malfunctioning through heat production. However the temperature sensor is placed on the outer shell of the servos, so it has a delay to the actual value of inside temperature, therefore can not accurately predict error through inside temperature.

Provided information by the measurement unit (All of available ones):

- Current position
- Current shell temperature

Our ”first gen” servos in wing -0 and -1 are MKS X6 HBL599. These are measuring position for inner control by a potentiometer. The SHM unit is measuring also this voltage with a 12 bit Analog-Digital Converter (ADC). Voltage was fitted for full scale: 5V to 360 degree, but we are using servos in a smaller range, not larger than about 120 degree. Only the third remains of the 12 bit domain. That means our scale is divided to 1365 unit. Secondly, this measurement shows us it is dependent of temperature, about 0.05 [°/℃] scale.

If we would know resistance of both halves of the sensor, it could eliminate the error, but first generation SHM only has 2 ADC input, and the other one is reserved for temperature sensor. Another solution could be using the temperature sensor, however this is placed out of the shell, therefore this is not exactly the same temperature which misleads the measurement, so not as good for correction. (Guys at TUM planning on setting up an ambient temperature sensor on the fuselage. Read more about in section 5)

Because of physical contact of electrical parts and possibly of ESD (Electro Static Discharge) phenomenons, electric noise is sitting on the signals. It is often visible when servo starts from a stationer state, there are high non-realistic spikes.

Controlling servos on individual PWM channels and wires consumes a lot of resources and space, and PWM is also not a fully digital communication can be source of noises on the control side, and these noises will be propagated through the servo.

The SHM has to retrofitted to the actuator, which is a time consuming and cumbersome process, with a success rate that is sub-optimal. This combined, with the inaccuracies, prompted us to investigate other solutions. We aimed for an off the shelf alternative, that could deliver the same, or better performance, whilst simplifying installation and improving reliability.

We started to look for actuators that use Controller Area Network (CAN) for communication. This way the servo would be able to provide feedback, and possibly other diagnostics information about itself, on the same bus it is using to receive the position reference. While many manufacturers have promised to bring out such devices, so far only HITEC is the only company to actually do so. We ordered a HITEC MD850TW-CAN actuator to test its capabilities. This is the strongest one in the company’s MD series, and it has the same outer diameter as the MKS HBL599 actuators that are currently installed in the aircraft.

The actuator promises to provide the following diagnostics data (Only the ones that might be relevant for us, are listed):

- Current position
- Current velocity
- Current torque
- Current voltage
- Current MCU temperature
- On time

In addition, the servo has configurable parameters, many of which can be modified during operation. These include:

- Speed
- Position limits
- Position command limit
- Voltage and temperature limits
- Deadband
- Acceleration and deceleration time.

For our use case, the position feedback is the most important, voltage and MCU temperatures are nice to have, and torque could be interesting. However from testing it seems that torque data is not really usable.

Most of the configurable features will have to be turned off. The different limits, if exceeded usually generate an emergency stop for the given actuator. This would not be desirable in the aircraft. Position command limits however are interesting. If activated, these will make sure that the actuator does not exceed the configured positions, even if a command is issued to do so. This is a convenient feature, to make sure that no damage is done to the control surfaces, due to a misconfigured test signal, or during any unforeseen error.

From testing we determined that a single servo is able to get position commands at 200 Hz, and send back two diagnostics values. We plan on reading the position in every cycle (at 200 Hz) and get voltage and temperature data in an alternating pattern (So effectively at 100Hz each). This is the maximum amount of data, that the servo can handle reliably.

We only have access to one actuator at the time of writing. But from partially simulating the others, and from simple calculations, it seems that a single CAN bus running at 1000 kbauds, can handle 12 of these actuators. All of them should be able to get the new position data, get the request for the diagnostics data, and send back the requested data.

Currently we plan on using a maximum of 9 actuators on 1 CAN bus, so the bus itself wont be a limitation.

The manufacturer does not specify the accuracy of the position feedback. We devised a test to measure it's accuracy and compare it to the SHM's. Here are the results:

As you can see the CAN actuator has a better accuracy than the current solution, while being significantly easier to install, and operate.

Measurement	CAN servo position error	CAN servo command error	PWM servo position error	PWM servo command error
Mean of abs. err. [deg]	0.3847	0.3982	0.5168	0.4786
Deviation [deg]	0.4935	0.5016	0.6312	0.5920
Min [deg]	-1.0089	-0.9620	-1.4503	-1.4866
Max [deg]	0.7040	0.6600	1.1686	1.0322

Table 9: Position measurements for MD950TW-CAN and HBL995+SHM

8.2 Direct Drive

One of the goals of the flexible WING-1 design is to have a high-bandwidth and high torque actuator on the outermost aileron control surface. This solution could demonstrate active flutter suppression to increase safe flight envelope. During horserace pattern flights, testlegs could be flown with increasing speeds.

8.2.1 System overview

In order to operate the flutter controller, several subsystems are necessary. Suppose that the baseline autopilot is operational and flying normal speed horserace pattern. To safely increase the flight speed, we have to estimate the "*distance*" to the flutter condition. The measurements of the flutterIMU sensors can be processed after collecting the data of one straight horserace leg (see Section 3). The OBC-II platform (see Section 4) is capable of running a script which can provide the information of current flutter status (on Fig 1 this is shown as "*Modal Analysis Data*"). Depending on this information, the flight speed may be increased, and at a certain point, the flutter control will command the DriectDrive to suppress the evolving flutter. At this point, it is crucial to have the fastest possible reaction time and the most precise reference tracking of the autopilot signals. After a flight test, for evaluating the results, diagnostic data is useful to exactly know the limits of the construction.

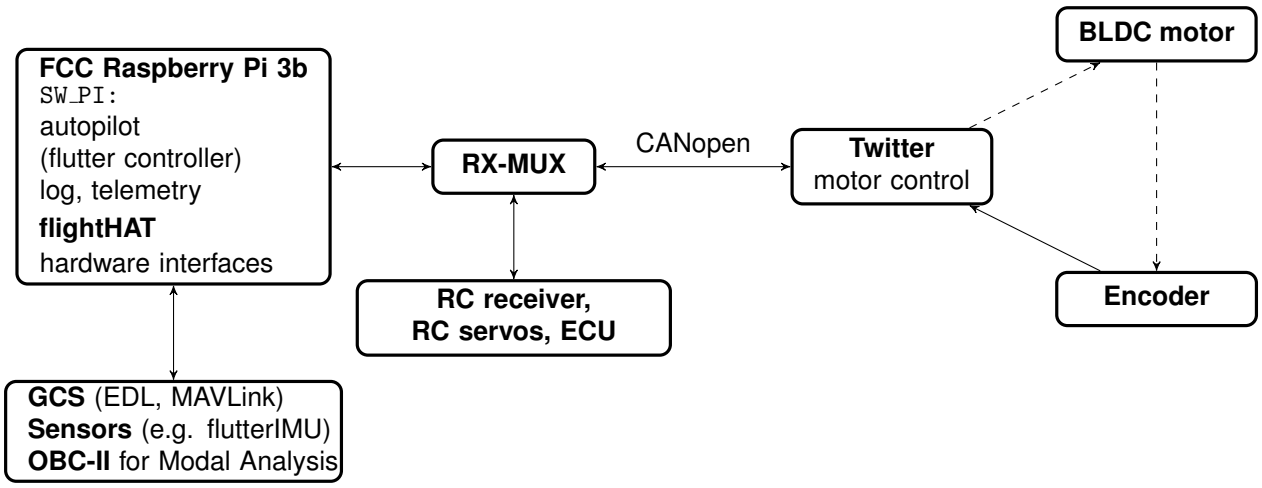


Figure 24: System overview of the DirectDrive

8.2.2 Integration and testing

Previously, the necessary CANopen messages for the operation of the DirectDrive were defined. Using this set of the messages, the inner state machine of the Elmo Gold Twitter could be handled properly. We verified the operation by sending out these message set with a Serial-CAN converter. A simple MATLAB GUI solution provided the interface for these tests. Finally, we managed to switch on the motor, set speed and acceleration limits, setpoints for moving the DirectDrive and to hold its desired position. Furthermore, diagnostic data queries are possible, e.g. supply voltage, current, or the position and velocity of the BLDC motor.

Still, the aforementioned functionality had to be integrated to use it with the FCC on the aircraft in operations. This involves several tasks to achieve full system integration. On the CANopen side, the RX-MUX controller have to implement the state machine and the corresponding messages for state transitions. The reference command for the actuator have to be given out in also manual mode and during autopilot operation, e.g. for the flutter controller. On the RX-MUX, look-up tables (LUT) have to be applied, to convert aileron deflection to rotation of the BLDC. However, it is expected to be almost linear due to small deflections, LUT is still an important part: enforcing the physical limits to the output command is crucial for safe operation.

During autopilot tests, some test signals were sent out on Raspberry Pi to measure the time lag before it sent out from RX-MUX. Then, the Twitter moves, and the 17 byte digital rotary encoder provides the feedback. The position is read out from the Twitter by the RX-MUX and sent back to the autopilot on the Raspberry Pi to close the loop. (This measurement route can be seen at Figure 1). Basically, that latency is is the reaction time what the flutter controller have to work with.

8.2.3 Results and future work

Unfortunately, after the implementation of the messages on the currently used RX-MUX controller, which utilises a Microchip PIC controller, the time lag measurement showed severe jitter in the data rates, and even the safety critical main cycle's execution times. This means that currently, the DD can only be used as a relatively slow actuator, and for ground tests only.

Since a new generation of RX-MUX hardware and software is designed from scratch using STMicrocontrollers STM32F4 platform, a way robust operation is expected soon. Using this new architecture, the flutter controller will have a trustworthy high-bandwidth actuator on the outermost control surface.

9 New wing concept (-3)

Since a new wing design could not be realized by the design loop under development on time, the concept of the -3-wing plans to use an existing -0 wing and refurbish it with a new flap layout using additional actuators based on an in-depth induced drag optimization campaign, with keeping either the -0 or -2 wing structure intact, but modifying the aerodynamic control surface layout. In the following sections, the rational of the process and the design is laid out.

9.1 Wing -0/-2 Refurbishing Feasibility Study

In an initial study, the possibility to refurbish an existing wing -0 or wing -2 for with eight instead of the existing four flaps was done. The following questions were investigated:

1. Attachment of the additional servos actuators required.
2. Supply of the additional servo actuators with power and control signal.
3. Attachment of the new flaps to the wing.
4. Production of the new flap design.

To the questions stated above, the following potential solutions were found:

1. (a) Attachment of servo actuators outside the wing.
   (b) Attachment of servos inside the wing.
2. (a) Using Y-cables from the existing wiring for power supply.
   (b) Reusing the additional CAN-bus cable existent in the wing.
   (c) Routing new cables between the rear-spar and the flaps.
3. (a) Reusing existent attachment points.
   (b) Setting new attachment points in the rear spar.
4. (a) Producing new flaps by cutting existing flaps.
   (b) Producing new flaps from CFRP-material and hand-layup process in a mold.

Since new flaps require new servo actuators, possible mounting strategies for the additional servos were investigated. Both mounting options were deemed feasible, even the mounting on the existing wing skin due to its thickness. A mounting inside the wing is deemed preferred, however, since it does not influence the aerodynamics as much. The aspect of cutting, and thereby damaging, the wing skin is not estimated to be a problem due to the design method applied.
The supply of the additional servos with power and control signal is deemed the most challenging since cross-talk between servos already lead to problems during the refitting of the landing gear and the difficulty (and from a practical view impossibility) to install new servo-cables inside the wing. Two different versions were considered: Using servo actuators using PWM-signals and servos using CAN-bus signals for receiving the rotary position. In the first case, the supply is deemed possible by Y-forking the existing power supply cables of the existing servos as well as reusing an existing spare CAN-bus cable with four wires to supply the servos with the PWM-signal needed. In the second case, it was deemed feasible to supply the CAN-bus servos by Y-forking the existent power supply cables as well as

using the spare CAN-bus cable to supply the servos with the control signal. Another possibility could be the supply of the servo with power using the CAN-bus cable as well.
Every flap is to be hinged with two hinge-points, since only 8 flaps are currently existing, it is decided to attach more hinge-points on the rear-spar of the existing wing. Either by using glue or screws or a combination of both.
For creating the new required flaps, the possibility of reusing the existing flaps by cutting them apart and the production of a new set of flaps using CNC-routed molds as well as a hand-lamination process were assessed. The second approach is deemed more suitable due to the comparable small size of the flaps and the large amount of experience in the production process at the institute.

9.2 Aerodynamic Analysis for the -3-Wing Design

Even though the full optimization process is not available to date, aerodynamic investigations were done in order to assess the improvements of the aerodynamic (induced) drag coefficient $C_{D,i}$ when wing shape control techniques are applied. The goal of the investigation was the comparison of effects of aforementioned technique on the -0- and -2-wing. The following figures 25 and 26 show the change in lift distribution over a range of airspeeds V for the -0- and -2-wing.

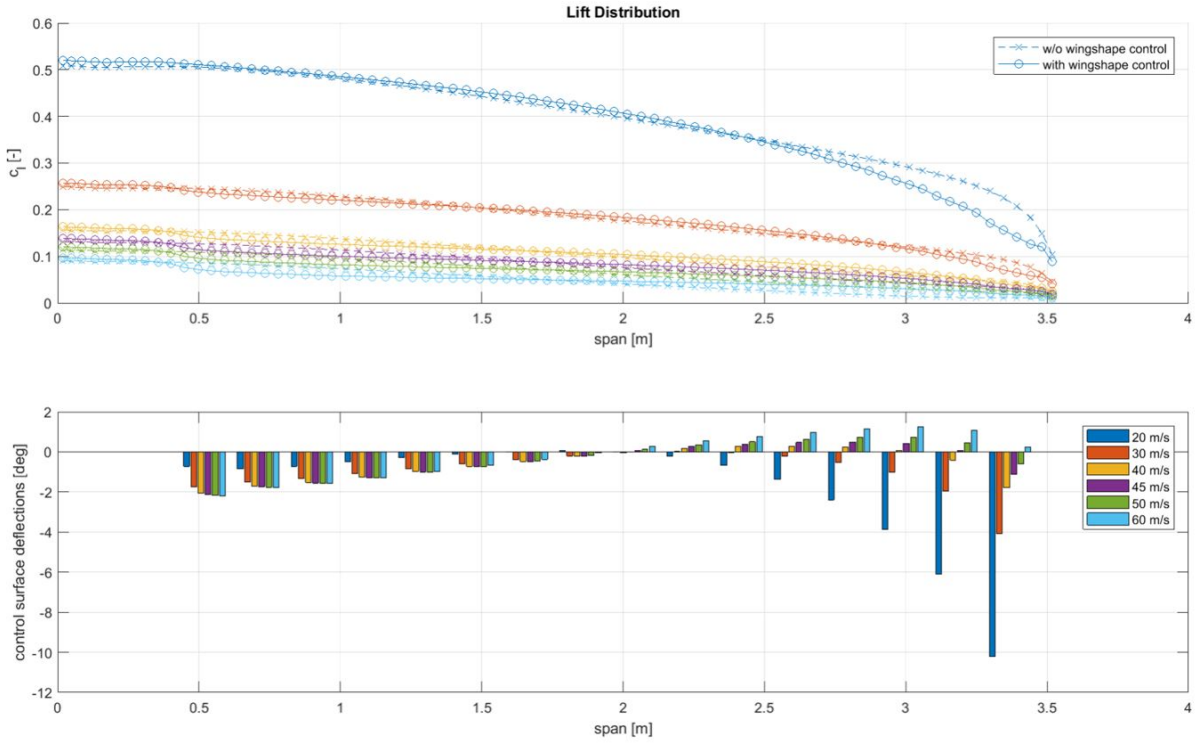


Figure 25: Wing -0: Lift distribution with and without wing shape control at different airspeeds and the respective flap deflections.

The improvements of the induced drag coefficient $C_{D,i}$ are shown in the following table 10

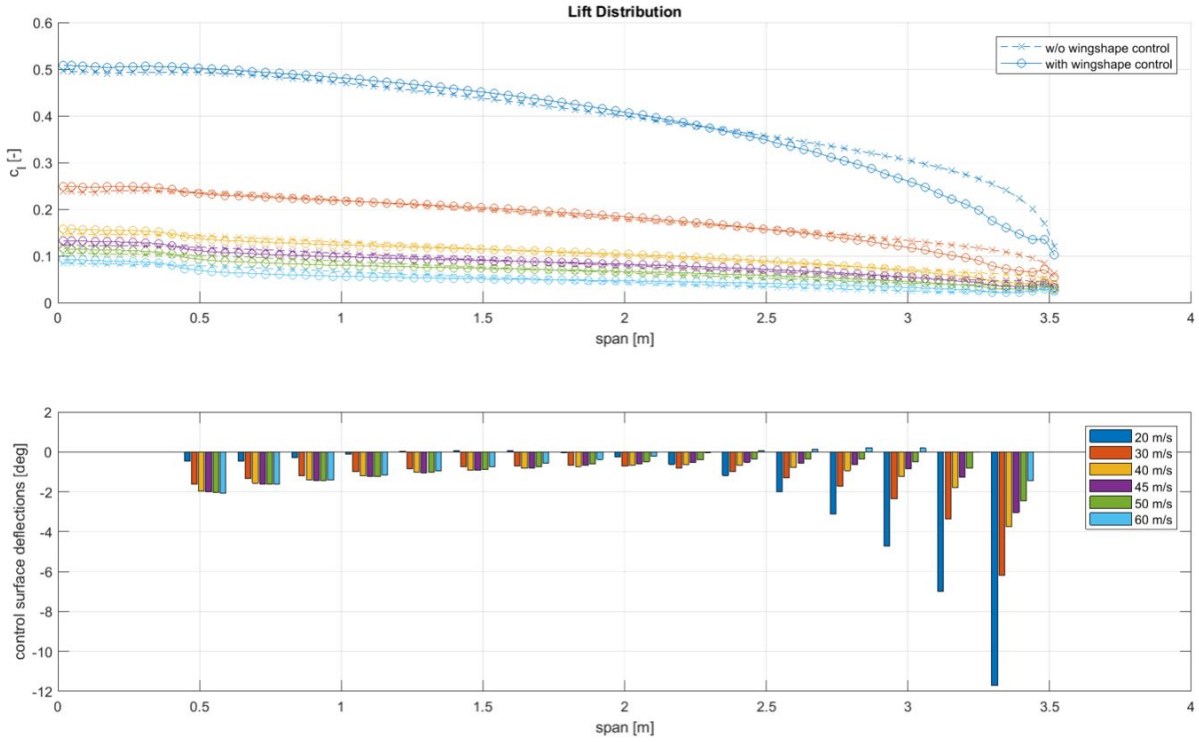


Figure 26: Wing -2: Lift distribution with and without wing shape control at different airspeeds and the respective flap deflections.

Wing	$20\frac{m}{s}$	$30\frac{m}{s}$	$40\frac{m}{s}$	$45\frac{m}{s}$	$50\frac{m}{s}$	$60\frac{m}{s}$
-0	3.59%	2.47%	4.32%	6.7%	6.87%	17.32%
-2	4.82%	6.62%	4.56%	4.88%	5.48%	7.58%

Table 10: Improvements of the induced drag coefficient $C_{D,i}$ at different airspeeds of the wing -0 and wing -2 due to wing shape control technique.

Since according to table 10 higher improvements drag are to be expected for the -0-wing. Therefore, this wing es selected for further investigations.
As gets apparent from figures 25 and 26, the investigation so far only considered 16 flaps, therefore another investigation was done comparing the effect of different flap number (16 vs. 9). The number of 9 was derived from the understanding, that a finer granulation of flap distribution on the outer parts of the wing are beneficial, therefore choosing a 1-2-3-3 separation of the existing flaps. Another difference to the first analysis is further, that for this investigation the jig-shape of the wing is being considered. Again, the different shapes of lift distribution are shown in the following figures 27 and 28, and the expected reduction in drag coefficient $C_{D,i}$ in the table 11.

No. flaps	$20\frac{m}{s}$	$30\frac{m}{s}$	$40\frac{m}{s}$	$45\frac{m}{s}$	$50\frac{m}{s}$	$60\frac{m}{s}$
16	3.76%	2.97%	3.98%	5.36%	7.21%	11.69%
9	3.74%	2.88%	3.76%	4.92%	6.79%	11.05%

Table 11: Improvements of drag coefficient $C_{D,i}$ at different airspeeds of the wing -0 with 16 and 9 flaps due to wing shape control technique.

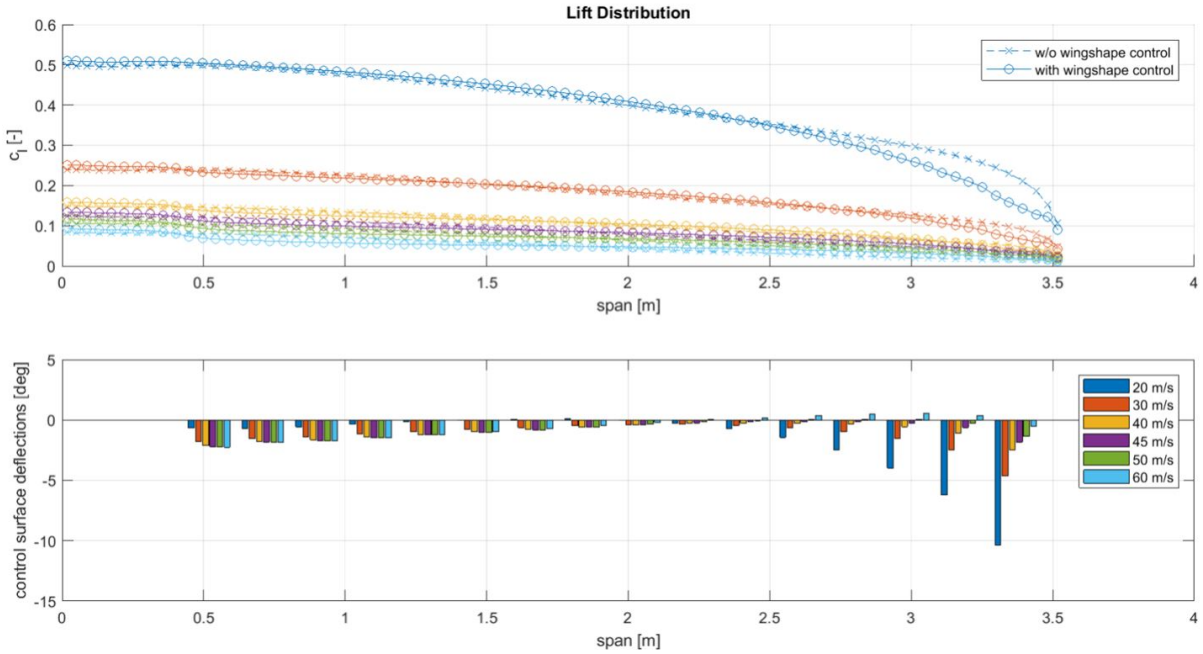


Figure 27: Wing -0 with 16: Lift distribution with and without wing shape control at different airspeeds and the respective flap deflections.

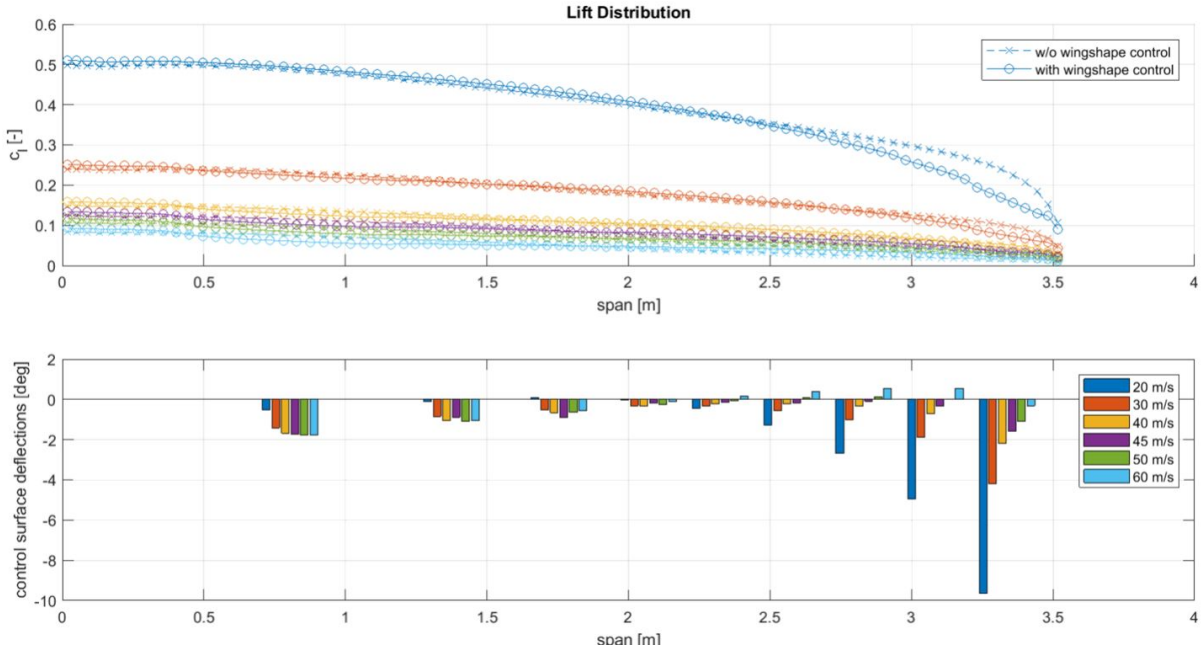


Figure 28: Wing -0 with 9 flaps: Lift distribution with and without wing shape control at different airspeeds and the respective flap deflections.

It gets apparent, that the differences between the concept featuring 9 flaps does not perform substantially worse than the concept featuring 16 flaps. Therefore, arrangements are made, to refurbish an

existing -0-wing with nine flaps.

9.3 Current state of the -3-Wing Design

To date, a master's thesis at TUM has been submitted that has focused on the redesign of the wing. The hinge moments of the different flaps have been investigated, the servos placed and a detailed design of the flaps created. The resulting -3-wing is depicted in figure 29. The next steps will comprise the

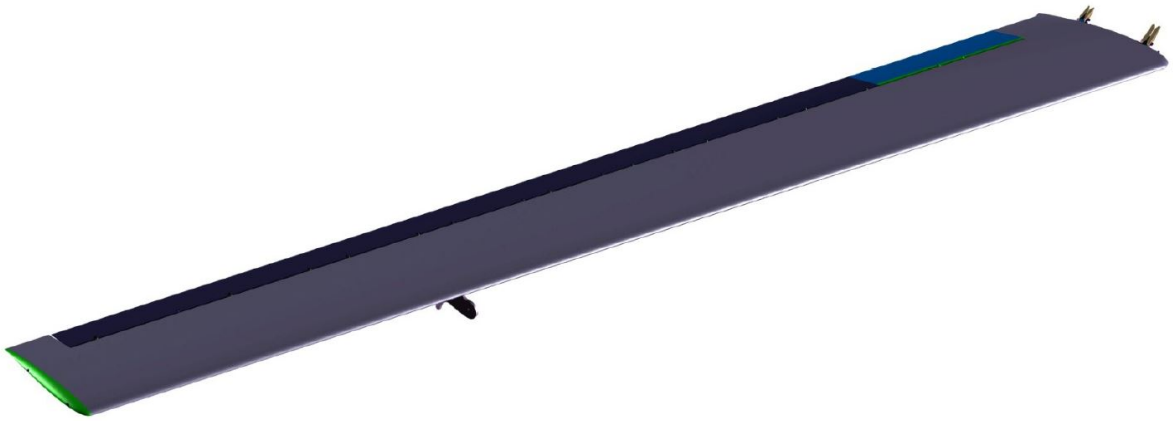


Figure 29: Overview of the -3-wing design implemented in the CAD-program *CATIA V5*.

detailed design of the molds for flap production, servo selection, wiring planning and implementation.

10 Conclusion and outlook

Based on the presented concepts, the manufacturing of the new wing is starting soon. Still there are some details to be discussed, but the main objectives were outlined in this document.

We can conclude that several subsystems of the aircraft were improved: either in trustworthiness, or with new functionalities. Regarding the FCC, beyond software updates for the new sensor interfaces, a brand-new RX-MUX board was designed with extended computational power and with easier development tool-chain (Section 2).

New sensor configurations are discussed in Section 3 for flutter IMUs, and for the ADS & xSens in Section 5. New sensors on-board were also introduced: TMS in Section 6 and optical wingshape monitoring in Section 7. Their results were also evaluated.

New digital actuators are presented in Section 8 with comparison to the currently used ones in other wings. The feasibility of integration of the sensors and actuators is discussed in Section 9 along with the aerodynamic design of the new -3 wing.

The presented solutions in their current status are meant to be subsystems, working more or less independently - since due to the global chip shortage several components are difficult to purchase. As mentioned in Section 4, this has the advantages that not every hardware components are necessary for the development, thus there is no need to be shared with every consortium partners in their physical form.

11 Bibliography

[1] NVIDIA Corporation. GEFORCE RTX 3060 FAMILY, 2021. Available from: https://www.nvidia.com/en-us/geforce/graphics-cards/30-series/rtx-3060-3060ti/ [Accessed: 15/08/2021].

[2] O. Faugeras. Three-dimensional computer vision: A geometric viewpoint. In *Artificial Intelligence Series*. MIT Press, 1993.

[3] Insta360. Insta 360 ONE X, 2021. Available from: https://www.insta360.com/es/product/insta360-onex/ [Accessed: 30/08/2021].

[4] Intel. Intel® Core™ i5-10600K Processor, 2020. Available from: https://www.intel.com/content/www/us/en/products/sku/199311/intel-core-i510600k-processor-12m-cache-up-to-4-80-ghz/specifications.html [Accessed: 15/08/2021].

[5] Microsoft. Windows 10 Pro, 2021. Available from: https://www.microsoft.com/de-de/d/windows-10-pro/ [Accessed: 15/08/2021].

[6] Mobius. Instruction Manual for the Mobius ActionCam), 2015. Available from: http://www.mobius-actioncam.com/wp-content/uploads/2015/01/Mobius-Manual-23jan15a.pdf [Accessed: 15/06/2021].

[7] Mobius. Mobius 1080p HD Action Camera Standard Lens pack (Lens A2), 2021. Available from: https://www.mobius-cam.com/en/mobius-1-action-camera-c-29_6/mobius-1080p-hd-action-camera-standaard-lens-set-lens-a2-p-48?zenid=967hkuga6fhav17qks70u92ls6 [Accessed: 15/06/2021].

[8] OpenCV. Camera Calibration, 2021. Available from: https://docs.opencv.org/master/dc/dbb/tutorial_py_calibration.html [Accessed: 18/08/2021].

[9] OpenCV team. About, 2020. Available from: https://opencv.org/about/ [Accessed: 03/05/2021].

[10] D. Scaramuzza, A. Martinelli, and R. Siegwart. A flexible technique for accurate omnidirectional camera calibration and structure from motion. In *Fourth IEEE International Conference on Computer Vision Systems (ICVS'06)*, New York, NY, USA, January 2006. IEEE.

[11] Jianbo Shi and Carlo Tomasi. Good features to track. In *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, Seattle, WA, USA, June 1994. IEEE.

[12] TechSmith. Camtasia 2020, 2020. Available from: https://www.techsmith.com/camtasia-2020-press-release.html [Accessed: 15/06/2021].

[13] Matthias Wüstenhagen, Keith Soal, Szabolcs Tóth, Dániel Balogh, Fanglin Yu, Julius Bartasevicius, Sebastian Köberle, Daniel Teubl, and Bálint Vanek. D1.1 wing and demonstrator actuation requirements. Technical report, 2020.