Project Report

A Robust Method for Automatic Flight Detection

Author(s):
Severin Sadjina

Report No:
2021:10 -

Client(s):
BagID AS

SINTEF Ålesund AS
Postal address:
Borgundvegen 340
6009 Ålesund, Norway
Switchboard:

Enterprise /VAT No:
NO 918 787 747 MVA

Project Report

A Robust Method for Automatic Flight Detection

KEYWORDS:
Flight detection, data analysis, machine learning, accelerometer, air pressure sensor

VERSION
1.2

DATE
2021-06-03

AUTHOR(S)
Severin Sadjina

CLIENT(S)
BagID AS

CLIENT'S REFERENCE
Jan Vidar Nalbant

PROJECT NO.
312000130

NUMBER OF PAGES/APPENDICES:
33

SUMMARY

A Robust Method for Automatic Flight Detection

Here, we present a robust method for detecting flight automatically for use in digital luggage tags. The method is based on simple statistical aggregates using air pressure and 3D accelerometer measurements and complies with IATA and FAA requirements and recommendations. We achieve a correct detection of flight phases for 98.4% of the recorded data samples (with an ROC AUC score of 98.8%).

Data was recorded onboard several commercial flights in Norway, both in cabin and in the luggage compartment. Due to technical setbacks during the project period and the COVID-19 pandemic severely reducing the number of flights, the data available for analysis and method development was very limited. To mitigate the risk for severe overfitting, we used several countermeasures such as diligent cross-validation and the choice to keep model complexity low.

PREPARED BY
Severin Sadjina

SIGNATURE

CHECKED BY
Armin Pobitzer

SIGNATURE

APPROVED BY
Henning Borgen

SIGNATURE
Henning Borgen
Henning Borgen (Jun 3, 2021 15:46 GMT+2)

REPORT NO	ISBN	CLASSIFICATION	CLASSIFICATION THIS PAGE
10	-	Open	Open

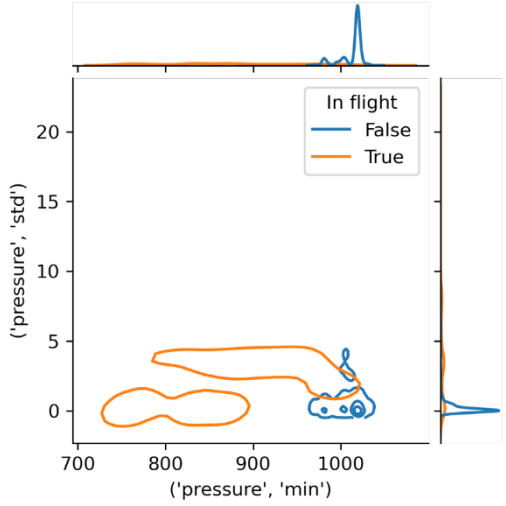
Document history

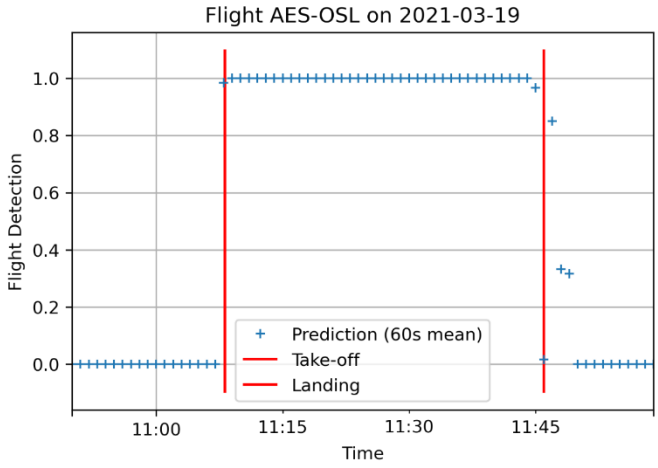
VERSION	DATE	VERSION DESCRIPTION
1.2	2021-06-03	Includes minor changes and improvements
1.1	2021-05-27	Includes final feedback from BagID AS
1.0	2021-05-23	Initial report

Table of contents

Summary

Here, we describe a method for automatic flight detection for use in digital luggage tags developed during a research project with BagID AS as an industry partner. To **comply with regulatory IATA and FAA requirements and recommendations**, the method is based on air pressure and 3D accelerometer sensor readings to provide two independent means to turn mobile communications off during flight.
We collected and analyzed data from several commercial flights in Norway (both in cabin and in the luggage compartment) and used various machine learning methods and techniques to identify the most reliable and most robust combination of sensor signals to detect flight. The final method predicts "In flight" or "Not in flight" based on simple statistical measures calculated over a 60 second time window prior to prediction. It can be called at pre-defined intervals (for example, every 10min) or upon request. The method **detects the flight phase correctly for 98.4% of all samples** (and "Not in flight" correctly for 95.0%).
The COVID-19 pandemic made it very challenging to collect data from commercial flights during the project period. Because of this, there was a substantial risk to develop a method that works well for the data at hand but performs poorly for yet unseen data (overfitting). To mitigate this risk as best as possible, we chose several counter-measures: A strong preference for keeping method complexity low, the diligent use of cross-validation, and reliance on domain knowledge and simple physics.
BagID is currently using online updated flight time data (take-off and landing times) in conjunction with sensor readings to determine whether the device is in flight or not, and to control mobile communications accordingly. While we have yet to validate this approach with the methods described here, it is very likely that such a combination further increases the reliability of the automatic flight detection method described here.






```python
def detect_flight(pressure_min, pressure_std, acceleration_norm_l1_max):

    in_flight = ((pressure_min <= 978.5) & (acceleration_norm_l1_max <= 1.9915)) |
    ((pressure_min > 978.5) & (pressure_std > 1.0819))

    return in_flight
```

1 Background and Motivation

Several stages along a passenger journey by air, such as ticket ordering and issuing or luggage check-in, already happen digitally. But luggage handling and check-in still relies on paper tags printed at the airport. On one hand, this constitutes a sizeable amount of paper and ink and a large expense for airlines. On the other hand, tracing by scanning paper tags at pre-determined locations is not quite optimal either and contributes to lost or damaged luggage which, in turn, causes flight delays. Smart and digital solutions hold great promise to make air travel more efficient and less taxing on the environment.

BagID AS is an Ålesund-based company offering digital luggage tags that make luggage handling and control easier and less costly. The tags have built in mobile communications and sensors which allow to trace the luggage, provide its users with real-time updated flight information, or detect shocks to identify potentially damaging handling.

To be able to use such devices onboard aircraft—and to be able to offer them on the market—flight authority requirements need to be fulfilled. According to the Federal Aviation Administration (FAA) and the International Air Transport Association (IATA), mobile communications need to be turned off during the entire flight phase, so that digital luggage tags must be able to **detect flight automatically and reliably**.

Between May 2019 and May 2021, SINTEF Ålesund carried out a research project with BagID as a partner to develop a reliable and robust method for the automatic detection of flight for use on aircraft. To that end, data from various sensors was collected and analyzed, and algorithmic methods for flight detection were developed and validated. The report at hand describes the project methods and results and comments on future work and relevant risk factors.

1.1 Outline

The present document is structured as follows. First, we will outline the relevant regulatory and technical requirements, list the key research questions, and related previous work in Sec. 2. In Sec. 3, we will outline the data collection process, the choice of sensors, and explain what data was available for algorithm development. We discuss the results of the data analysis and the chosen signal processing steps in Sec. 4. Sec. 5 will then describe the methodology for algorithm development and Sec. 6 will present the final results. Finally, we discuss how the method can be further improved and what risks and shortcomings need to be considered in Sec. 7.

2 Design of a Robust Method for Flight Detection

2.1 Requirements

The two regulatory frameworks relevant to the project and the development of an automatic flight detection algorithm are the IATA *Electronic Bag Tag Implementation Guide 1.1* [1] and the FAA advisory circular *AC 91.21-1D* on the use of portable electronic devices (PEDs) aboard aircraft [2]. IATA states that the "PED must be designed with a minimum of **two independent means to turn off** completely, turn off cellular or mobile functions, or a combination of both when airborne". The FAA "prohibits cellular telephone operation while airborne" but "allows the use of cellular telephones in aircraft while on the ground". It further defines "airborne" by using "the **takeoff and landing phases of flight** as the last possible conditions to detect turning off and back on again" (see Appendix B.1).

In addition to these regulatory requirements, the method should further:

- Focus on Norway as the geographic region of interest.
- Use as little energy as possible to help extend battery life.
- Be easily implementable in the production device firmware.
- Use only simpler functions for computation (for easy implementation and to save battery).
- Be reliable across the widest range of relevant conditions possible.
- Err on the side of simplicity rather than complexity (to increase reliability and reduce any risk of unforeseen or unwanted behavior).

- Be reasonably simple to further improve on in the future (and even support self-improvement processes).

2.2 Research Questions

Based on these requirements, we identified the following research questions for the project:

- Which signals are most useful for detecting flight in general?
- How do the sensors need to be configured and how do their measurements need to be processed?
- Which combinations and processing of sensor signals gives the highest reliability for flight detection?
- What information, other than sensor signals, is available that can aid automatic flight detection?
- What should happen if one of the signals is unavailable to still be able to fulfill the requirements?
- How is the method best implemented in production devices?
- How reliable can the method be made?

2.3 Related Work

In 2018, BagID and SINTEF Ålesund carried out a pre-study to determine which sensor readings would be most promising to detect flight automatically. The pre-study used the off-the-shelf data logger MSR145 (145B8) manufactured by MSR Electronics GmbH [3] to collect air pressure (10Hz), acceleration (50Hz), air temperature, air humidity, and light intensity (all 1Hz) during 4 commercial flights. It concluded that air pressure and accelerations were the most useful and reliable quantities to determine flight, with light intensity also appearing promising. It further suggested the use of a microphone to potentially pick up relevant sounds and noises (especially engine noises).

In [4], Tawk *et al*. describe how a 3D accelerometer is used together with simple signal processing to automatically detect flight. The patent [5] also uses 3D accelerometer data and simple signal processing for flight detection. A patent filed by Samsung in 2014 [6] uses an air pressure sensor in combination with a library of cabin pressures for various flight types. It also describes how the GSM module reveals the flight phase when it reports to be out-of-range.

3 Data Collection and Data Overview

The data used to develop the flight detection method was collected from BagID's production devices on commercial flights in Norway between 14.10.2020 and 19.03.2021. Sensor signals were first stored locally on the device and then transferred to the BagID servers via mobile communications. They were finally made available to SINTEF for analysis and development. Unfortunately, local data storage and buffering gave us several challenges and limited data acquisition during flight phases to 4500 seconds (at 1Hz storage rate).

In addition to the sensor data collected by the devices themselves, meta data about the various flight phases was available for most of the flights, see Table 1. This includes times for luggage drop off, aircraft push back from gate, take off, seatbelt on and off signals during flight, any turbulence events, landing, parking, and luggage pick up.

Table 1: Overview over all data sets available to the project.

File name	Route	Aircraft Type	Device	Take-off	Landing
2020-10-14-JV-OSL-AES-2_sensor_202010210729.csv	OSL-AES	Boeing 737-800	Production	2020-10-14 18:25:12	2020-10-14 19:06:58
2020-10-22-LEV-AES-OSL_sensor_202010220744.csv	AES-OSL	Boeing 737-800	Production	2020-10-22 04:44:12	2020-10-22 05:28:00
2020-10-23-LEV-OSL-AES_sensor_202010261339.csv	OSL-AES	Boeing 737-800	Production	2020-10-23 12:33:30	2020-10-23 13:13:00

2020-10-23-LEV-OSL-AES_sensor_202010261345.csv	OSL-AES	Boeing 737-800	Production	2020-10-23 12:33:30	2020-10-23 13:13:00
2020-11-09-JB-OSL-TOS.csv	OSL-TOS	Boeing 737-800	Production	2020-11-09 10:57:00	2020-11-09 12:48:00
2020-11-20-EV-AES-BGO_22255.csv	AES-BGO	De Havilland Canada Dash 8-400	Production	2020-11-20 05:28:10	2020-11-20 06:01:40
2020-11-20-EV-AES-BGO_24822.csv	AES-BGO	De Havilland Canada Dash 8-400	Production	2020-11-20 05:28:10	2020-11-20 06:01:40
2020-11-20-EV-AES-BGO_24855.csv	AES-BGO	De Havilland Canada Dash 8-400	Production	2020-11-20 05:28:10	2020-11-20 06:01:40
2020-11-20-EV-AES-BGO_24863.csv	AES-BGO	De Havilland Canada Dash 8-400	Production	2020-11-20 05:28:10	2020-11-20 06:01:40
2020-11-20-EV-AES-BGO_26124.csv	AES-BGO	De Havilland Canada Dash 8-400	Production	2020-11-20 05:28:10	2020-11-20 06:01:40
2020-11-20-EV-AES-BGO_30001.csv	AES-BGO	De Havilland Canada Dash 8-400	Production	2020-11-20 05:28:10	2020-11-20 06:01:40
2020-11-20-EV-BGO-AES_22255.csv	BGO-AES	De Havilland Canada Dash 8-400	Production	2020-11-20 12:02:45	2020-11-20 12:33:30
2020-11-20-EV-BGO-AES_24822.csv	BGO-AES	De Havilland Canada Dash 8-400	Production	2020-11-20 12:02:45	2020-11-20 12:33:30
2020-11-20-EV-BGO-AES_24855.csv	BGO-AES	De Havilland Canada Dash 8-400	Production	2020-11-20 12:02:45	2020-11-20 12:33:30
2020-11-20-EV-BGO-AES_24863.csv	BGO-AES	De Havilland Canada Dash 8-400	Production	2020-11-20 12:02:45	2020-11-20 12:33:30
2020-11-20-EV-BGO-AES_26124.csv	BGO-AES	De Havilland Canada Dash 8-400	Production	2020-11-20 12:02:45	2020-11-20 12:33:30
2020-11-20-EV-BGO-AES_30001.csv	BGO-AES	De Havilland Canada Dash 8-400	Production	2020-11-20 12:02:45	2020-11-20 12:33:30
2020-12-07-JV-OSL-AES.csv	OSL-AES	Boeing 737-800	Production	2020-12-07 09:55:40	2020-12-07 10:36:45
2021-03-19-JV-AES-OSL_20697.csv	AES-OSL	Boeing 737-800	Production	2021-03-19 11:08:10	2021-03-19 11:46:00
2021-03-19-JV-AES-OSL_20960.csv	AES-OSL	Boeing 737-800	Production	2021-03-19 11:08:10	2021-03-19 11:46:00
2021-03-19-JV-AES-OSL_25282.csv	AES-OSL	Boeing 737-800	Production	2021-03-19 11:08:10	2021-03-19 11:46:00
Logger 1 - 03.10.18.csv	OSL-AES	Bombardier CRJ900	MSR145 data logger	2018-10-03 13:30:00	2018-10-03 14:19:00
MSR325489_180907_153000.csv	AES-OSL	Boeing 737-800	MSR145 data logger	2018-09-07 16:22:15	2018-09-07 17:02:50
MSR325489_181005_173000.csv	AES-OSL	Boeing 737-800	MSR145 data logger	2018-10-05 18:50:50	2018-10-05 19:27:00
MSR325489_181101_140500.csv[1]	-	-	MSR145 data logger	2018-10-31 15:07:00	2018-10-31 15:51:00

In total, 21 datasets from 8 separate flights were made available to the project. Several datasets contained missing data and less reliable readings. Complete data (at least air pressure and acceleration available during all phases of flight) was only available for 4 flights, all of which happened with the same aircraft type (Boeing 737-800) and on the same route (AES—OSL). 2 more flights on the route AES—BGO with De Havilland Canada Dash 8-400 were largely complete and only lacked some data during take-off. Data was also collected for different placements of the digital luggage tag (in the cabin and in the luggage compartment, either fastened on the outside of the luggage or inside the luggage). In addition, data from 4

[1] This data for this flight was erroneously recorded on the wrong day and thus contains no flight event at all. We included it for validation purposes.

flights was available from the pre-study [7]. While it was captured with different equipment (see Sec. 2.3) it still proved useful for development and validation. Table 2 gives an overview of the airports for which data was recorded during the project.

Table 2: The four Norwegian airports represented in the available flight data.

Airport	Call sign	Altitude / [m]	Latitude	Longitude
Ålesund (Vigra)	AES	21	62°33'45"N	6°07'11"E
Oslo (Gardermoen)	OSL	208	60°12'10"N	011°05'02"E
Bergen (Flesland)	BGO	52	60°17'37"N	005°13'05"E
Tromsø (Langnes)	TOS	10	69°40'53"N	018°55'04"E

Overall, a relatively very **small amount of data was available** for analysis and method development. We tried to address this through careful and robust development and validation (see Sec. 5). Further comments on the risks stemming from little data are found in Sec. 7.1.1.

3.1 Choice of Sensors

The pre-study [7] revealed that **air pressure and accelerations** were most useful for the automatic detection of flight and that the device must be able to sense them. BagID's production devices, in addition, include sensors for light intensity, air temperature, and air humidity. For some of the flights, 3D gyroscope data was also available which can be used to identify horizontal and vertical acceleration components efficiently and reliably.

Table 3: Sensors used in the devices used for data collection.

Sensor model	Type	Sampling frequency used	Note
Bosch BMI088[2]	3D accelerometer	1Hz	Gyroscope data was not used during the project to reduce method complexity and power consumption
Bosch BME280[3]	Combined humidity and pressure sensor	1Hz	
LITE-ON LTR-303ALS-01[4]	Light sensor	1Hz	

3.1.1 Sampling Frequencies

To be able to register engine noise, a sampling frequency of at least 50Hz is recommended for the accelerometer [7]. Because such frequencies were initially deemed much too high by BagID with respect to on-device data buffering, we investigated the use of 1Hz and finally found it to be sufficiently useful for flight detection. All other sensors were also **logged at 1Hz** which is well within the range recommended by the pre-study.

[2] BOSCH BMI088 Accelerometer data sheet: https://ae-bst.resource.bosch.com/media/_tech/media/datasheets/BST-BMI088-DS001.pdf. Accessed 21.05.2021.
[3] BOSCH BME280 Combined humidity and pressure sensor data sheet: https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bme280-ds002.pdf. Accessed 21.05.2021.
[4] LITE-ON LTR-303ALS-01 data sheet: https://optoelectronics.liteon.com/upload/download/DS86-2013-0004/LTR-303ALS-01_DS_V1.pdf. Accessed 21.05.2021.

Table 4: Sensor spec recommendations for flight detection. Due to data buffering constraints, accelerations were recorded at 1Hz.

Sensor	Sampling Rate		Resolution	Range
	Minimum	Recommended		
Accelerometer	60 Hz	100 Hz	16 Bit	(-2, +2) g
Pressure		10 Hz	16 Bit	(0, 2000) mbar
Gyroscope	(Same as accelerometer)		16 Bit	(-100, 100) °/s
Temperature		1 Hz	16 Bit	(-10, 40) °C
Humidity		1 Hz	≥ 8 Bit	(0, 100) %

4 Data Analysis and Processing

4.1 Sensor Data

Except for air pressure, a general trend that can be observed across all flights, is that flight phases show lower amounts of variations. That is, extreme values and the deviations from typical levels in general are smaller than there are outside of the flight phases.

4.1.1 Air Pressure

Air pressure measurements contain a low amount of noise and seem to be **quite reliable** for detecting flight automatically. While on the ground, air pressure typically shows lower amounts of variation while large changes are observable during ascent and descent, see Figure 2. Naturally, lower air pressure readings are consistently observed during flight, see Figure 1.

Interestingly, sudden "humps" appear around take-off and landing phases that can be observed across all data sets, see Figure 3. We were not fully able to discern the cause for this phenomenon. It is likely related to the acceleration and deceleration which would cause the air pressure inside the aircraft to change as air is forced towards the back or the front of the aircraft, respectively.

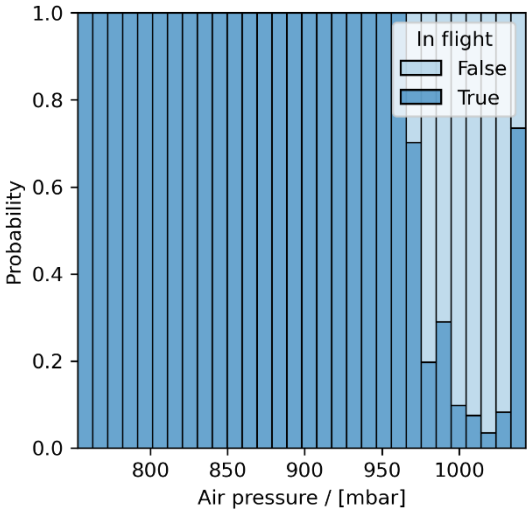


Figure 1: Distribution of all air pressure measurements.

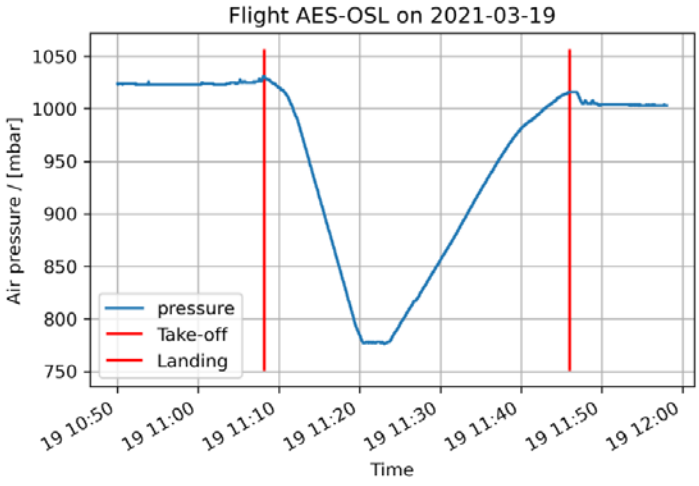


Figure 2: Air pressure recorded on a flight from AES to OSL.

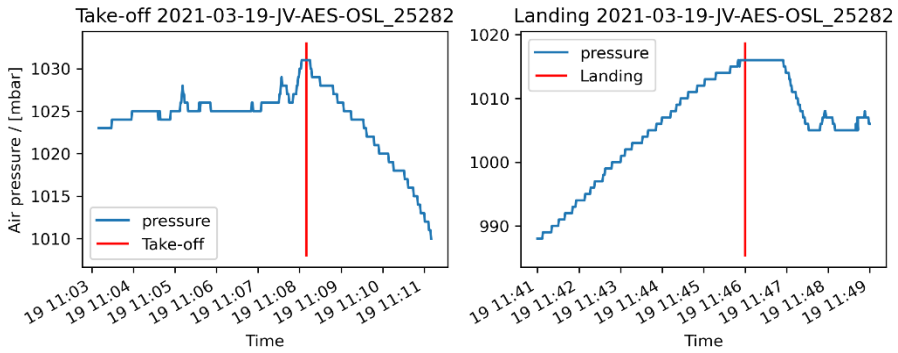


Figure 3: Air pressure recorded around take-off (left) and landing (right) on a flight from AES to OSL.

4.1.2 Accelerations

Generally speaking, it is more **difficult to extract patterns** from 3D accelerometer data, which makes accelerations—though promising—challenging to use for flight detection. This is, largely, because it is a 3-dimensional quantity, the sensor's orientation in space is not always known, and there are many other signals and patterns in the data that are not useful for flight detection.

The captured information spans the entire frequency spectrum and gives a comparatively very high "noise backdrop" against which it is difficult to extract the relevant signals for flight detection. As an example, sudden shocks and knocks can, especially outside of flight, lead to very large readings and change the sensor orientation. Figure 4 shows the y-component measured during a flight between Bergen (BGO) and Ålesund (AES) and illustrates these issues (unknown and changing orientations, high noise levels, and shock event with large magnitudes). Figure 5 shows the distribution of accelerometer readings for all flights. Note how variations are noticeably smaller during flight phases.

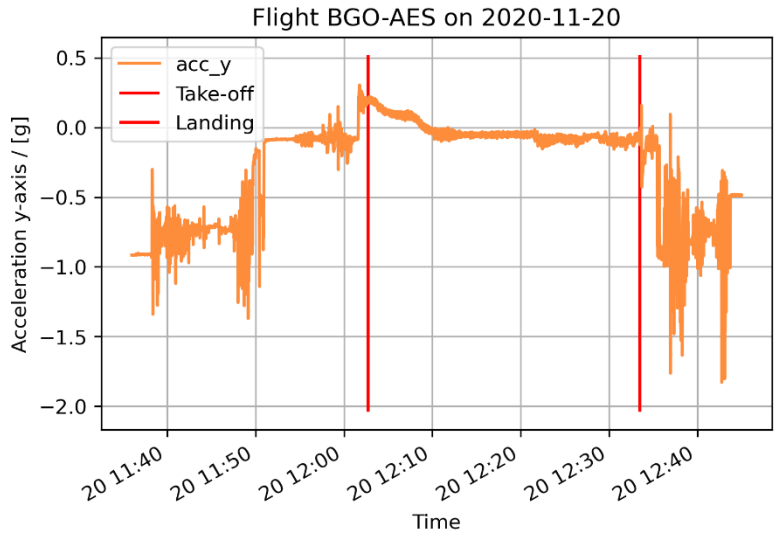


Figure 4: Y-axis acceleration recorded on a flight from BGO to AES.

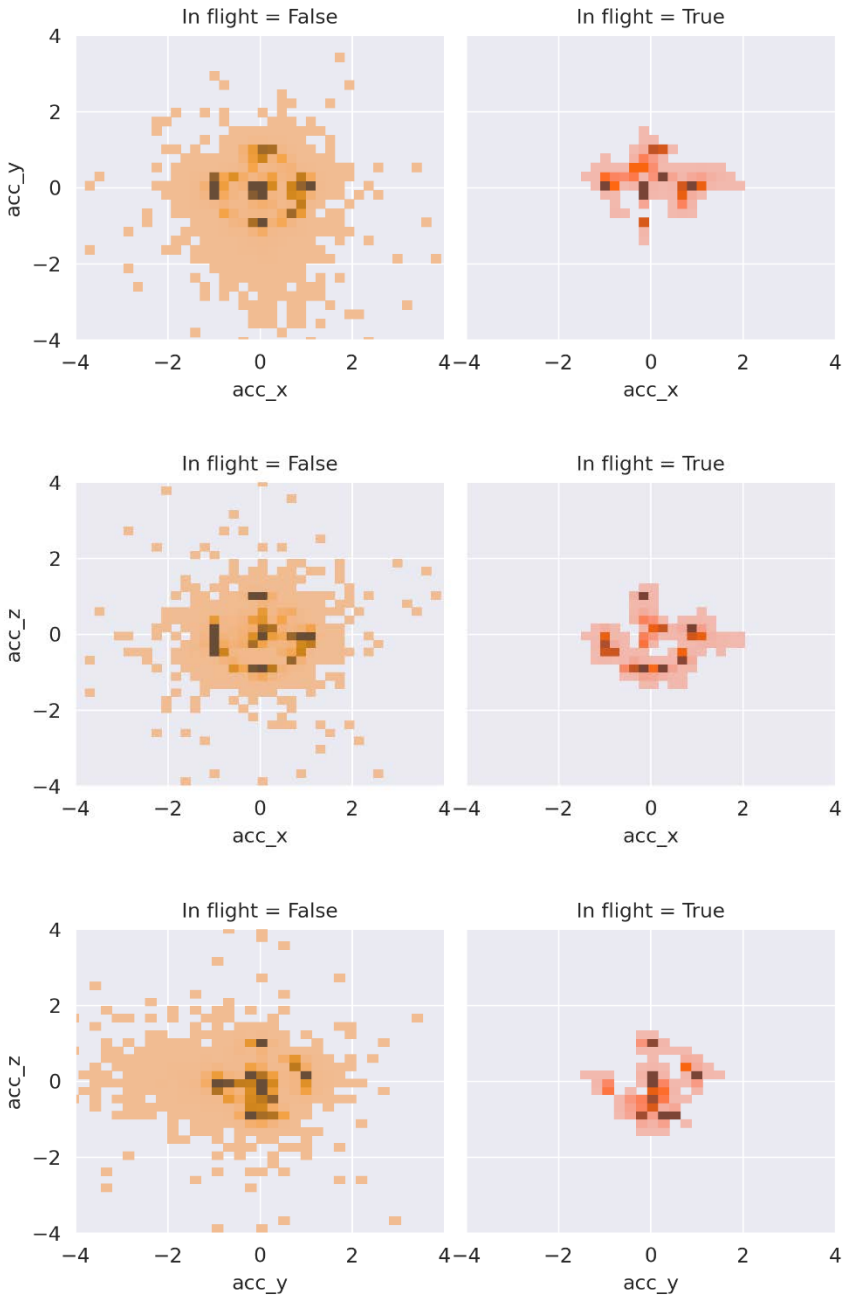


Figure 5: Density plots for the 3D-accelerometer components during (right) and outside (left) of flight phases. Shown are the x, y, and z components ("acc_x", "acc_y", "acc_z") in units of 1g.

4.1.3 Other Sensors

We also captured and analyzed light intensity, air temperature, and air humidity during the project. In conclusion, **none of these was very useful** and we were not able to establish to what extent they can be used for flight detection with the small amount of data we had to work with. Light intensity, however, may still be a promising measurement (in addition to air pressure and accelerations). As mentioned, all data show smaller amounts of variations during flight phases.

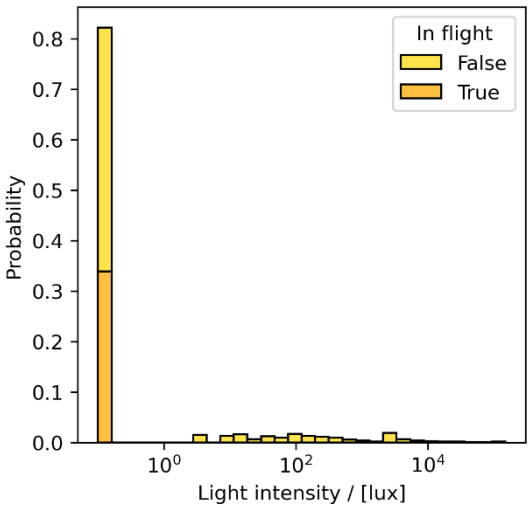


Figure 6: Distribution of all light intensity measurements. (A constant 0.1lux is added to the measurements to be able to plot on logarithmic axis.)

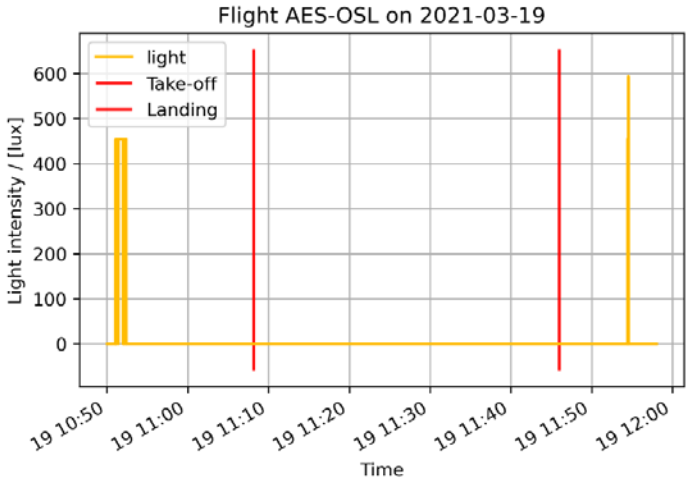


Figure 7: Light intensity recorded on a flight from AES to OSL.

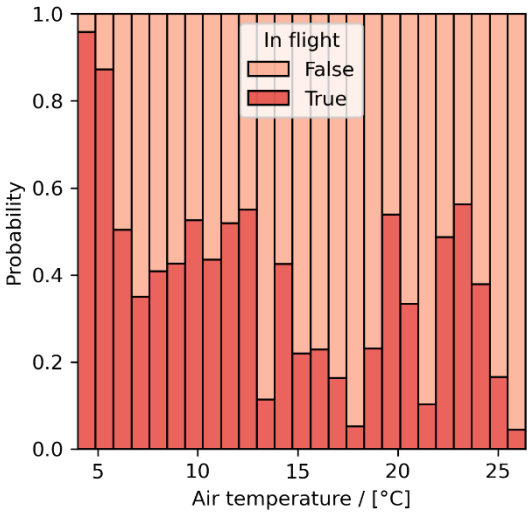


Figure 8: Distribution of all air temperature measurements.

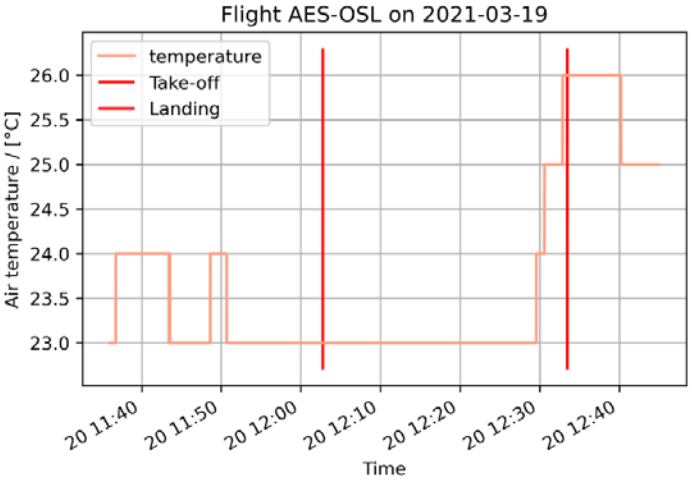


Figure 9: Air temperature recorded on a flight from AES to OSL.

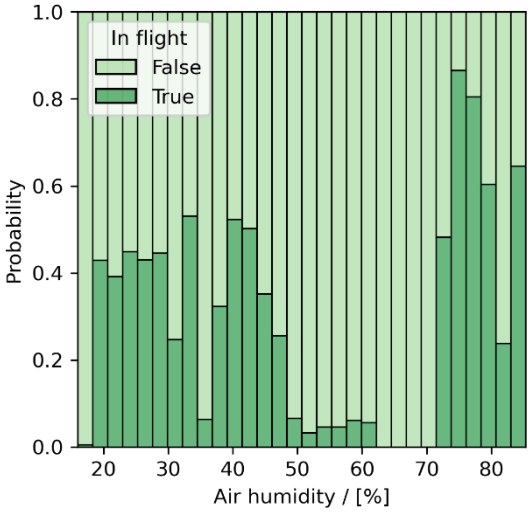


Figure 10: Distribution of all air humidity measurements.

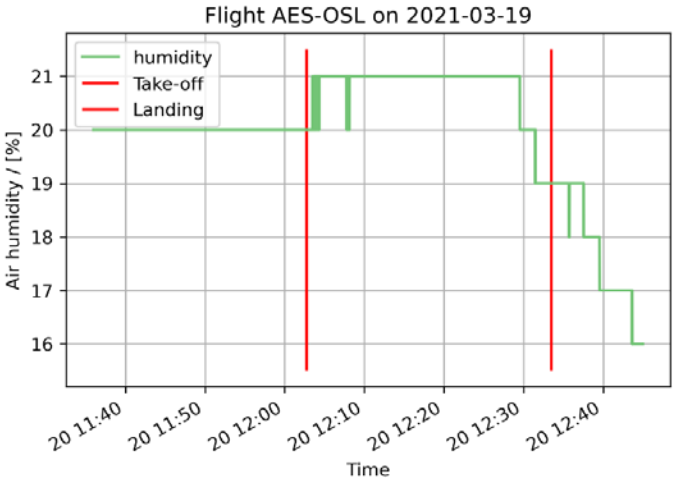


Figure 11: Air humidity recorded on a flight from AES to OSL.

4.2 Signal Processing

The **resampling and the aggregation** of the sensor signals is necessary to be able to develop a simple but robust and reliable method according to the requirements (see Section 2.1). To this end, we studied time windows of 30, 60, and 120 seconds and various aggregation methods to quantify signal magnitudes as well as variations. For most parts, we used **simpler statistical measures**: Minimum, maximum, median, and mean to represent magnitudes; and quantiles, interquartile range, standard deviation, and mean average deviation to represent variations. In addition, we also tested methods used in human movement detection from 3D acceleration data [8]: Entropy, signal energy, kinetic energy, RMS, and zero crossing rate. Figure 12 and Figure 13 exemplify the signal processing for the standard deviation of the air pressure using 60s aggregation windows. Note the much larger variations during flight phase owing to aircraft ascending and descending.

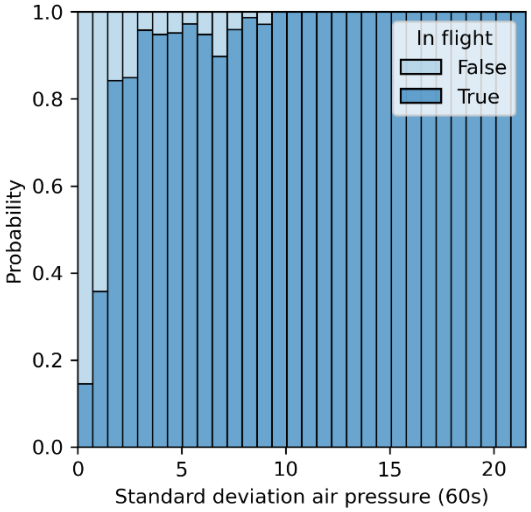


Figure 12: Distribution of the standard deviation for all air pressure measurements using 60s time windows.

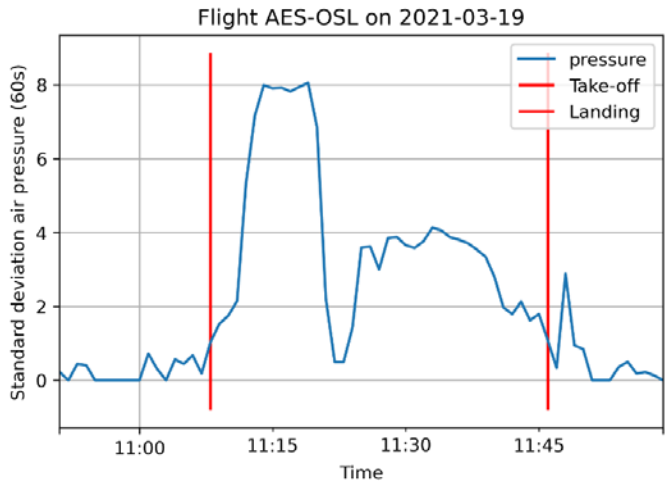


Figure 13: Standard deviation of air pressure over 60s time windows recorded on a flight from AES to OSL.

4.2.1 Accelerations

For accelerations, which are given as 3-dimensional vectors, different processing was available and needed. One very useful (and simple) measure are vector norms, we used both **L1 and L2 norms**. Figure 14 and Figure 15 show the distribution of the mean and the standard deviation of the vector norms for all flight measurements over 60 second time windows. Note how larger values are almost exclusively found outside of flight phases.

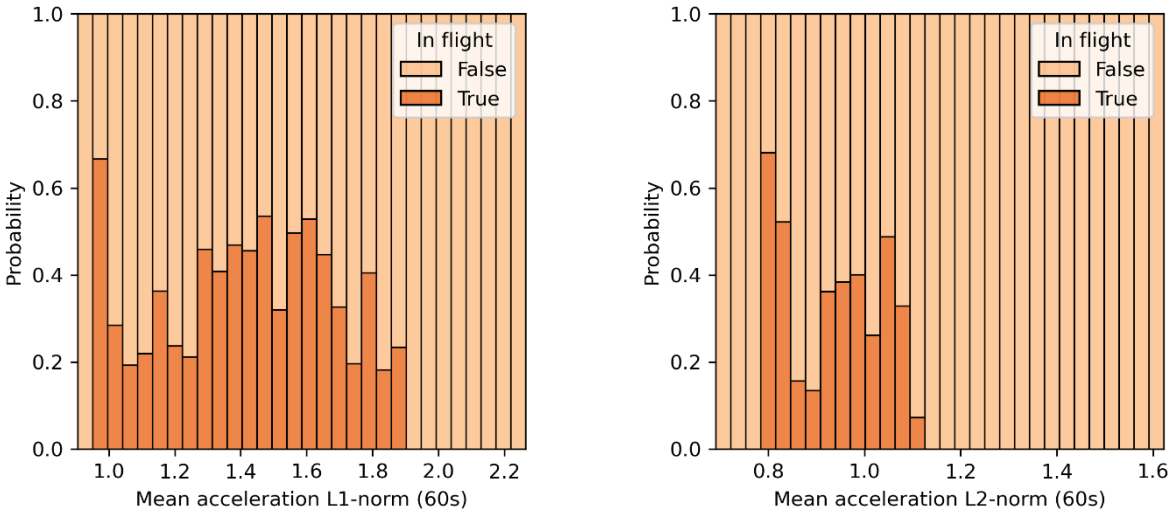


Figure 14: Distribution of the mean of the acceleration L1-norm (left) and L2-norm (right) using 60s time windows.

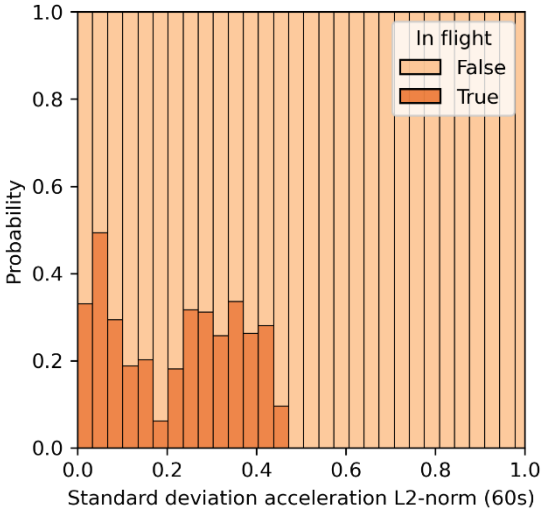


Figure 15: Distribution of the standard deviation of the acceleration L2-norm using 60s time windows.

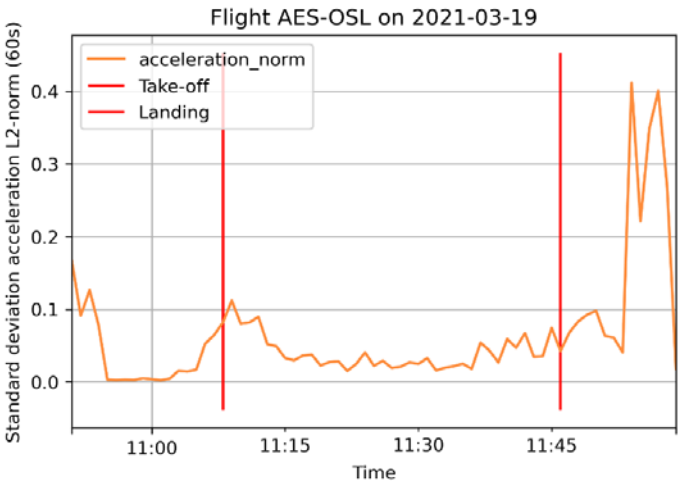


Figure 16: Standard deviation of the acceleration L2-norm recorded on a flight from AES to OSL.

The distribution of total acceleration (L2-norm) shown in Figure 17 also reveals heavy tails: While almost all readings are clustered around 1g, some measurements (shocks and knocks) extend far beyond that typical range.

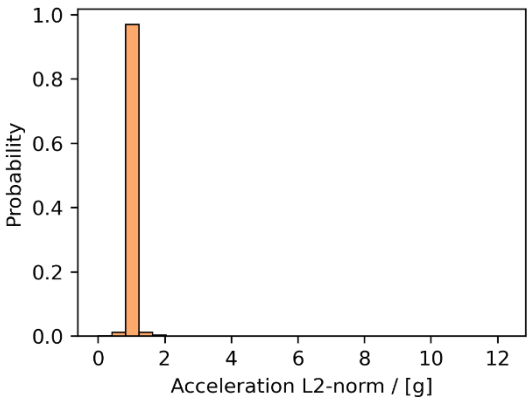


Figure 17: Distribution of total acceleration (L2-norm) showing heavy tails.

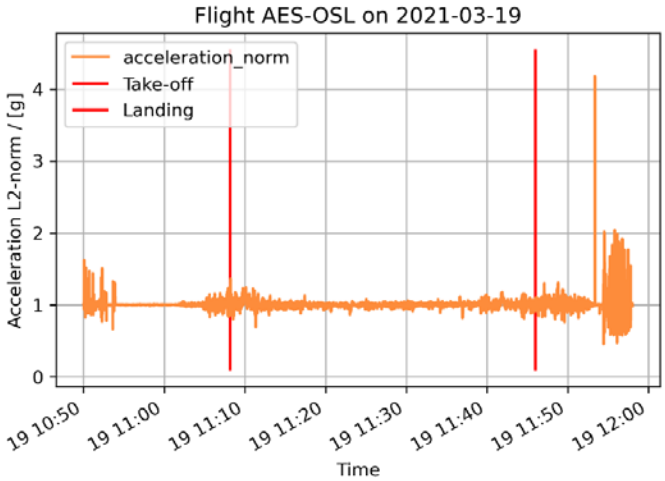


Figure 18: Acceleration L2-norm recorded on a flight from AES to OSL.

Knowing the accelerometer's orientation in space allows to decompose the signals into **horizontal and vertical acceleration components**. The orientation can directly be measured using a gyroscope. As mentioned, we refrained from using the gyroscope during the project due to issues with availability and power consumption and to keep final method complexity low.
Alternatively, we investigated the following method to decompose accelerations into horizontal and vertical components:

1. First, the "activity" is detected using simple thresholds (mean and variance) for the sensor readings to fulfill the assumption that the (norm of the) acceleration is approximately 1g and, thus, the device at rest.
2. Next, if no activity is detected, roll and pitch angles are calculated from the 3-dimensional accelerometer data. If activity is detected, this calculation is unreliable, and previous roll and pitch angles are used.
3. Using the roll and pitch angles, the 3D acceleration vector is rotated and then decomposed in horizontal and vertical components.

Figure 19 shows the distribution of horizontal and vertical acceleration components deconstructed in this way. Again, it is evident that flight phases are characterized by less variation in the signals with the vertical component largely registered around 1g (as expected). However, also note how the deconstruction method described here is not always reliable and gives vertical components between -1g and 1g.

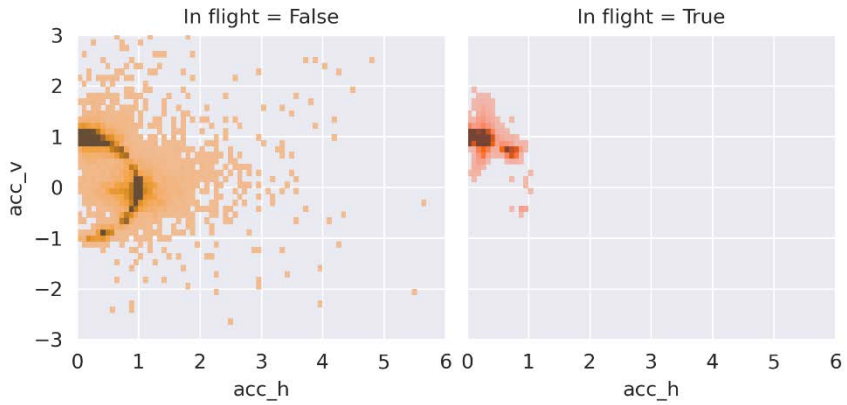


Figure 19: Density plots for the horizontal ("acc_h") and vertical ("acc_v") accelerometer components during (right) and outside (left) of flight phases (in units of 1g).

All in all, we found it difficult to reliably extract useful patters from decomposition into horizontal and vertical acceleration components. In some cases, see Figure 20, the method worked well enough to show the prolonged period of horizontal acceleration associated with take-off and ascent. But other flight phases were much harder to discern, and the method outright failed for several flights.

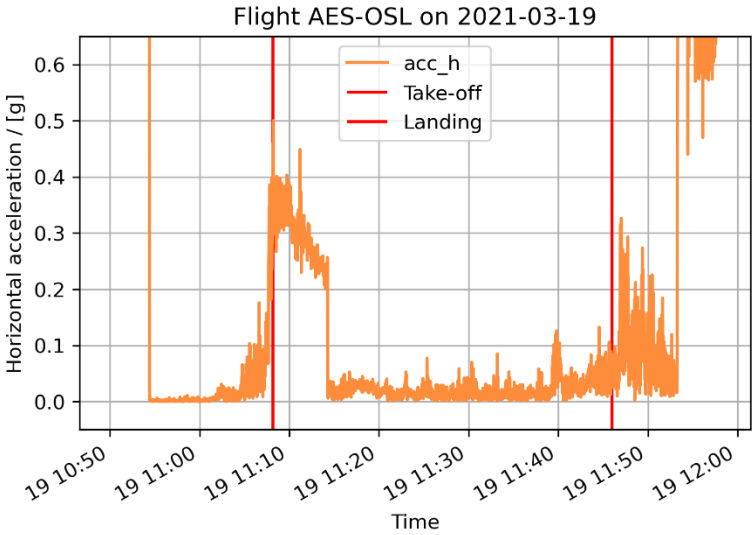


Figure 20: Horizontal acceleration deconstructed from measurements from a flight from AES to OSL. Larger values are clipped to better illustrate the flight phase.

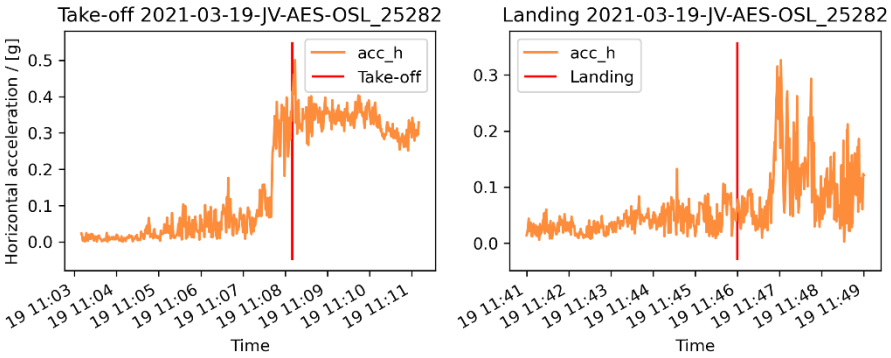


Figure 21: Deconstructed horizontal acceleration around take-off (left) and landing (right) on a flight from AES to OSL.

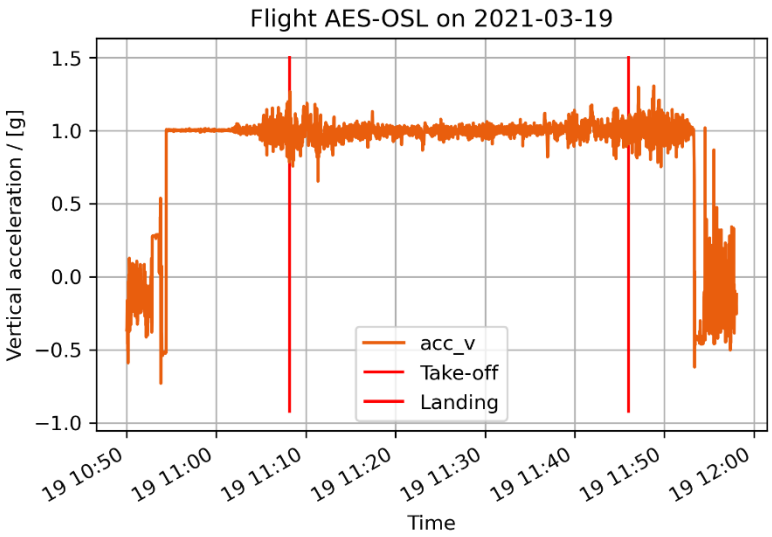


Figure 22: Vertical acceleration deconstructed from measurements from a flight from AES to OSL.

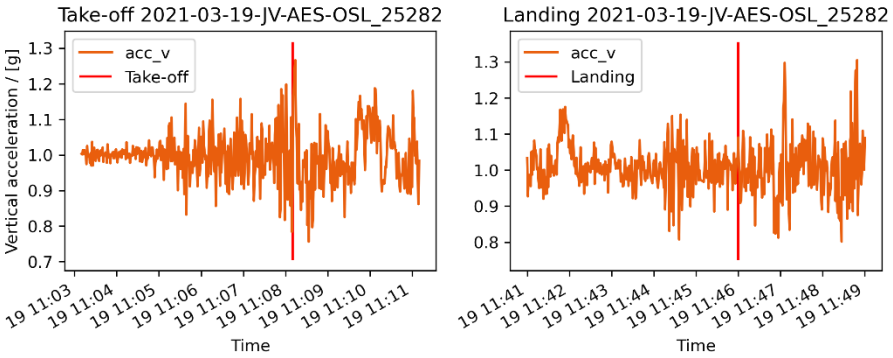


Figure 23: Deconstructed vertical acceleration around take-off (left) and landing (right) on a flight from AES to OSL.

Lastly, it is worth noting that acceleration data has very high levels of noise and is comparatively hard to analyze and process. Because of this, we briefly investigated the use of a Butterworth bandpass filter. We eventually dropped this approach to keep method complexity low and because it would require more data, time, and work.

5 Method Development

A first general consideration for the flight detection method was whether it should be stateless or not, that is, whether the strategy should be to detect take-off (as the last possible time to turn off communications) and landing (as the earliest possible time to turn them back on) directly. In the end, we chose a stateless method because it gives two key benefits:

- It is less complex and, thus, more robust in that it probably works better for conditions that we could not yet analyze due to a lack of data.
- It allows to activate the sensors and call the method at given intervals (e.g., every 10 minutes) or upon request, and to save power in between.

The final strategy was thus to try to detect whether the device is in flight only based on sensor readings during a limited time window (for example 60 seconds) immediately prior. This way, the device collects sensor readings for a shorter amount of time, calls the method to detect flight based only on these, and then turn mobile communications on (if no flight was detected) or tries again at a later time (if flight was detected).

Nonetheless, some work was done during the project on trying to detect take-off and landing directly, see also Section 7.2.

5.1 Data Preparation and Preprocessing

The first stage of processing the data was resampling so that measurements from a previous time slice can help determine whether we are in flight right at that moment. We studied time slices of 30, 60, and 120 seconds, with the shorter slices offering a faster response and the longer ones offering more reliability and less noise. Overall, **60 second time slices** gave the best tradeoff between performance, response, and ease of use (implementation).

To greatly increase the amount of data to train and validate the method on, we performed resampling with all possible time shifts (the start time of each time slice was shifted by 0, 1, 2, …, 59 seconds) see Figure 24. In the end, this process gave a total of 143.117 individual samples (time slices) to work with (including the data from the pre-study).

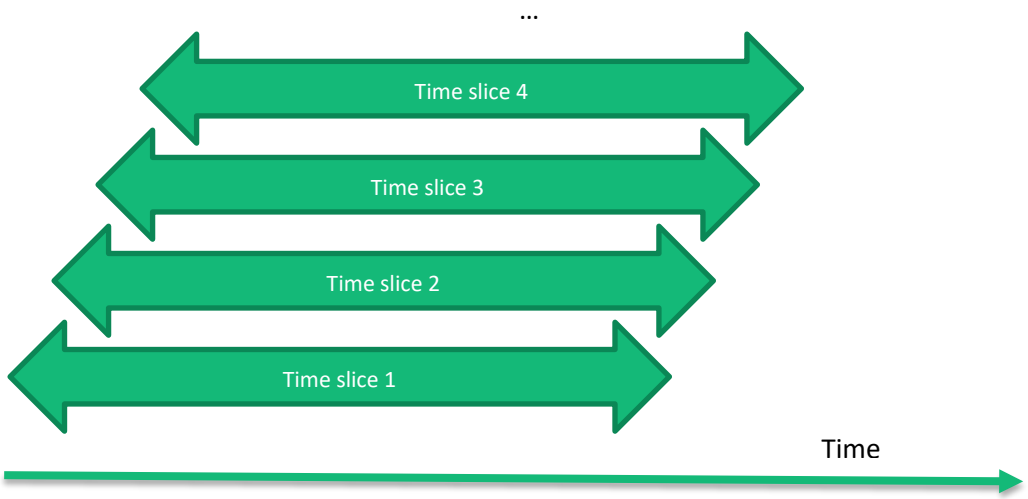


Figure 24: Resampling of data with different time shifts. Each slice is 60 seconds long and uses different start times.

After resampling, we computed the various statistical functions (such as mean, quantiles, or entropy) for each signal (air pressure, acceleration norm, etc.) based on each of the time slices (samples).

5.2 Methodology for Finding the Best Combinations

To help identify the **most reliable and strongest relationships** between combinations of sensor readings and flight, we used different machine learning methods. The general strategy was to train a machine learning model on all, or part of, the processed sensor data and have it predict whether the signals correspond to "In flight" or "Not in flight". Decision trees, random forests, logistic regression, and Gaussian naive Bayes classifiers were tested and used to help identify the most promising combinations.

A **decision tree classifier** is a simple machine learning method which, based on data, tries to identify the best variables (features) and the best thresholds to make a prediction (here, "In flight" or "Not in flight"). Not only can it be used to identify the best combination of sensor readings, a trained decision tree can also easily and directly be implemented as code. Finally, decision tree classifiers are able to predict probabilities for each of the classes (here, "In flight" or "Not in flight").

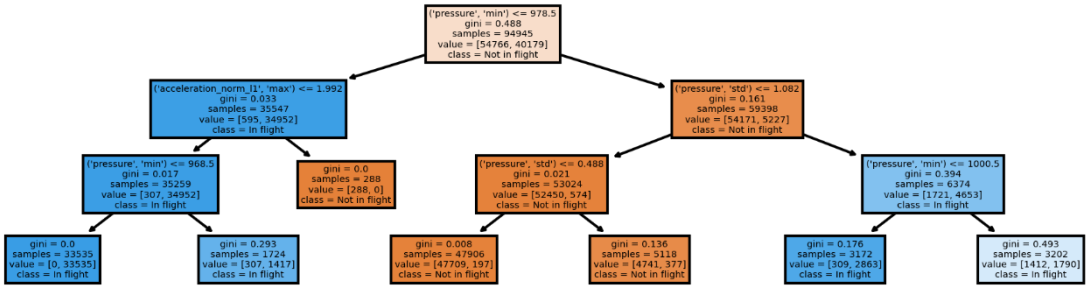


Figure 25: Example for a decision tree classifier. Conditions are checked from top to bottom: If true, move to the left, if false, move to the right. For example, if the air pressure is not above 978.5mbar and the acceleration L1-norm is above 1.992g, predict "Not in flight".

A **random forest classifier** is a method based on several individual decision trees each of which is only trained on a subset of the data or the features. This typically gives better performance and is more robust compared to single decision trees but is also not as easy to implement. We trained a random forest classifier on the data to help identify useful feature combinations using impurity-based feature importances, see Figure 26, and by studying the best performing individual trees.

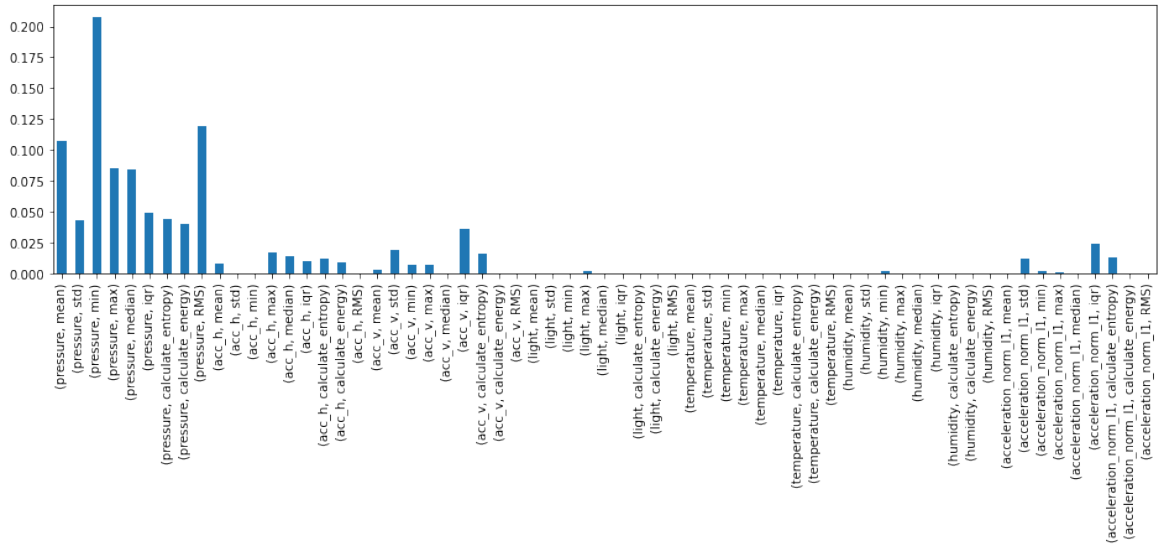


Figure 26: Impurity-based feature importances retrieved from a random forest classifier trained on all available data from production devices.

Lastly, a **cross-validated feature selection** method was used to cross-check the results from the decision tree and random forest classifiers and for optimization. It evaluates all possible feature combinations (or randomly drawn subsets) using cross-validated grid search for hyperparameter tuning for each of the combinations.

5.3 Training and Validation

The machine learning methods were trained on all data from the BagID production devices, while the data from the pre-study was used as a hold-out test set. Classification performance was assessed using **10-fold stratified k-folds cross-validation** and Area Under the Receiver Operating Characteristic Curve (**ROC AUC**) scoring. The use of cross-validation and rather simplistic methods is critical to reduce the (very likely) **risk for overfitting** resulting from the small amount of data available, see Section 7.1.1.

We chose the **trade-off between classifier specificity and sensitivity** with the help of confusion matrices, see Figure 27, and precision-recall curves, see Figure 28. To avoid overfitting, their computations were based on cross-validated estimates (10-fold stratified k-folds).

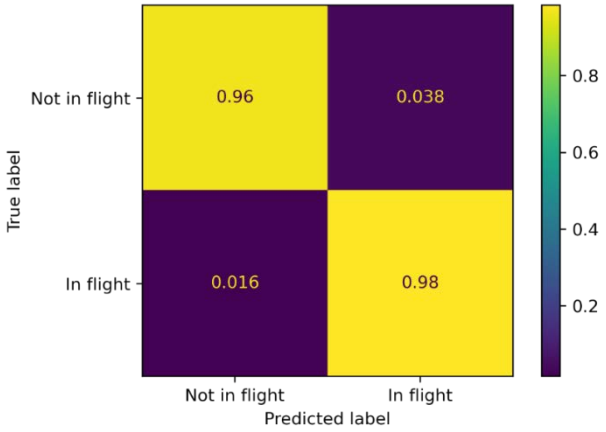


Figure 27: (Cross-validated) confusion matrix for the decision tree classifier (threshold = 0.5).

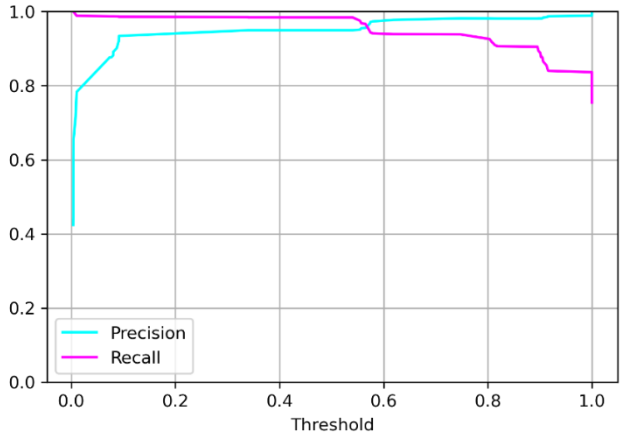


Figure 28: (Cross-validated) precision-recall curve for the decision tree classifier. Here, a threshold of 0.2 was chosen as optimal.

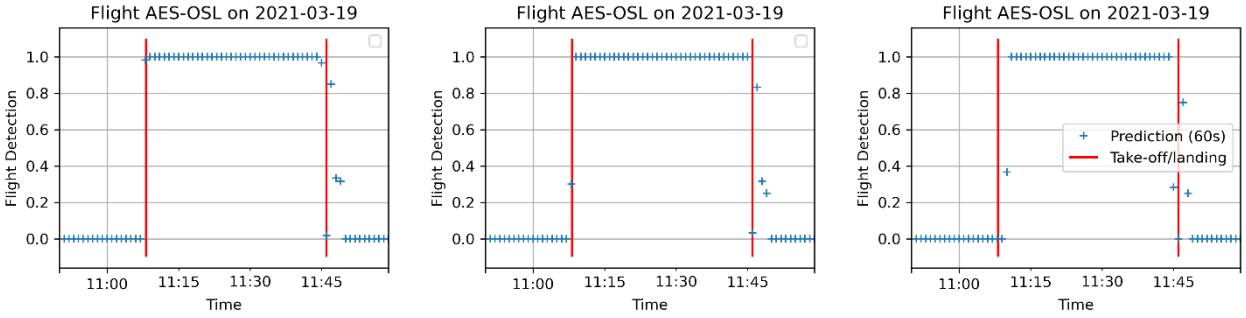


Figure 29: Flight detection for a flight from AES to OSL using a decision tree classifier (left), a random forest classifier (middle), and logistic regression (right). Predictions are averaged over 60s time windows for better visualization.

It is worth pointing out that the **ground truth** for the labels "In flight" and "Not in flight" was established using the take-off and landing times given in Table 1 and that these will not be 100% accurate, generally speaking. Because of this, there is some inherent uncertainty in the final scores and performance metrics given in this report, and it is hard to distinguish between this effect and true method performance.

6 Results

As mentioned before, we found time windows of 60 seconds in combination with 1Hz signal sampling frequencies best for flight detection in production devices. Based on the methodology described in the previous Section 5, **air pressure levels and variations together with acceleration norms** were the best performing combination of features. Minimum and standard deviation worked best for air pressure levels and variations, respectively, and maximum worked best for the acceleration norm. There were, however, only minor differences between using these and similar corresponding measures, such as median or mean (instead of minimum or maximum) and entropy or interquartile ranges (instead of standard deviation). Likewise, we observed only minor differences in performance between using the L1 or L2 norm for the acceleration.

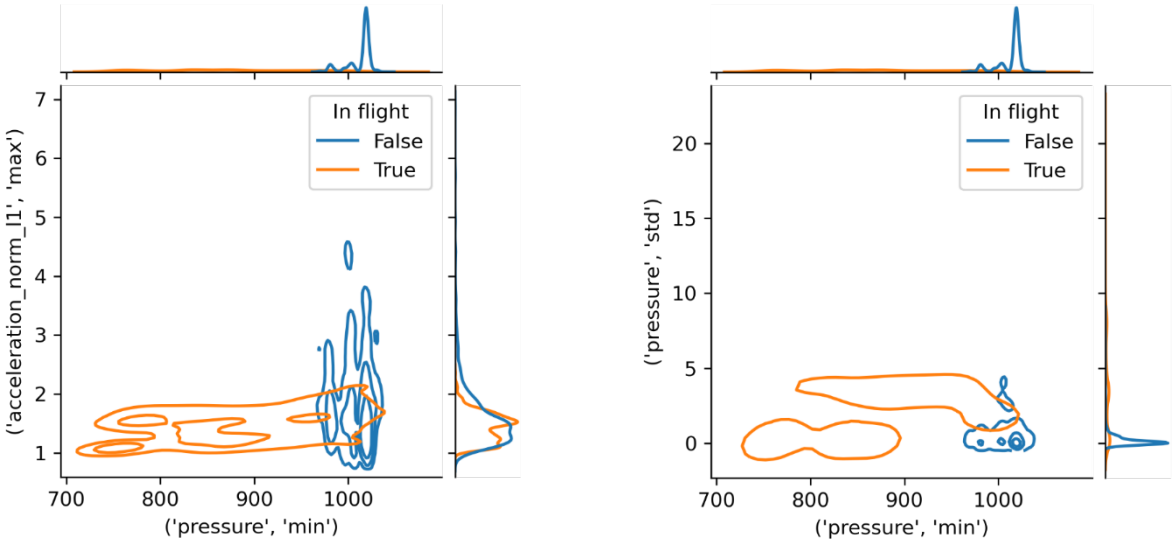


Figure 30: Contour plots for the feature combinations that were identified as most useful for the detection of flight based on 60s time slices: Minimum air pressure and maximum acceleration L1-norm (left) and minimum air pressure and standard deviation of the air pressure (right). (Air pressure in units of 1mbar and acceleration in units of 1g)

All final methods were directly based on a **decision tree classifier** because it provides the best feature combinations along with simple-to-implement value thresholds. In other words, it allows us to express the flight detection method in terms of simple rules directly, offering a very high level of interpretability.

6.1 Final Method

Code 1 and Figure 31 illustrate the final method which worked best on the available data in terms of the validation described in Section 5.3. It is based on a decision tree classifier using minimum air pressure, standard deviation of the air pressure, and the maximum of the acceleration norm (L1), all with respect to a 60 second time window prior to detection.

Code 1: Final method implemented in Python. It is based on the minimum air pressure ("pressure_min"), the standard deviation of the air pressure ("pressure_std"), and the maximum of the acceleration L1-norm ("acceleration_norm_l1_max").

```
def detect_flight(pressure_min, pressure_std, acceleration_norm_l1_max):

    in_flight = ((pressure_min <= 978.5) & (acceleration_norm_l1_max <= 1.9915)) |
    ((pressure_min > 978.5) & (pressure_std > 1.0819))

    return in_flight
```

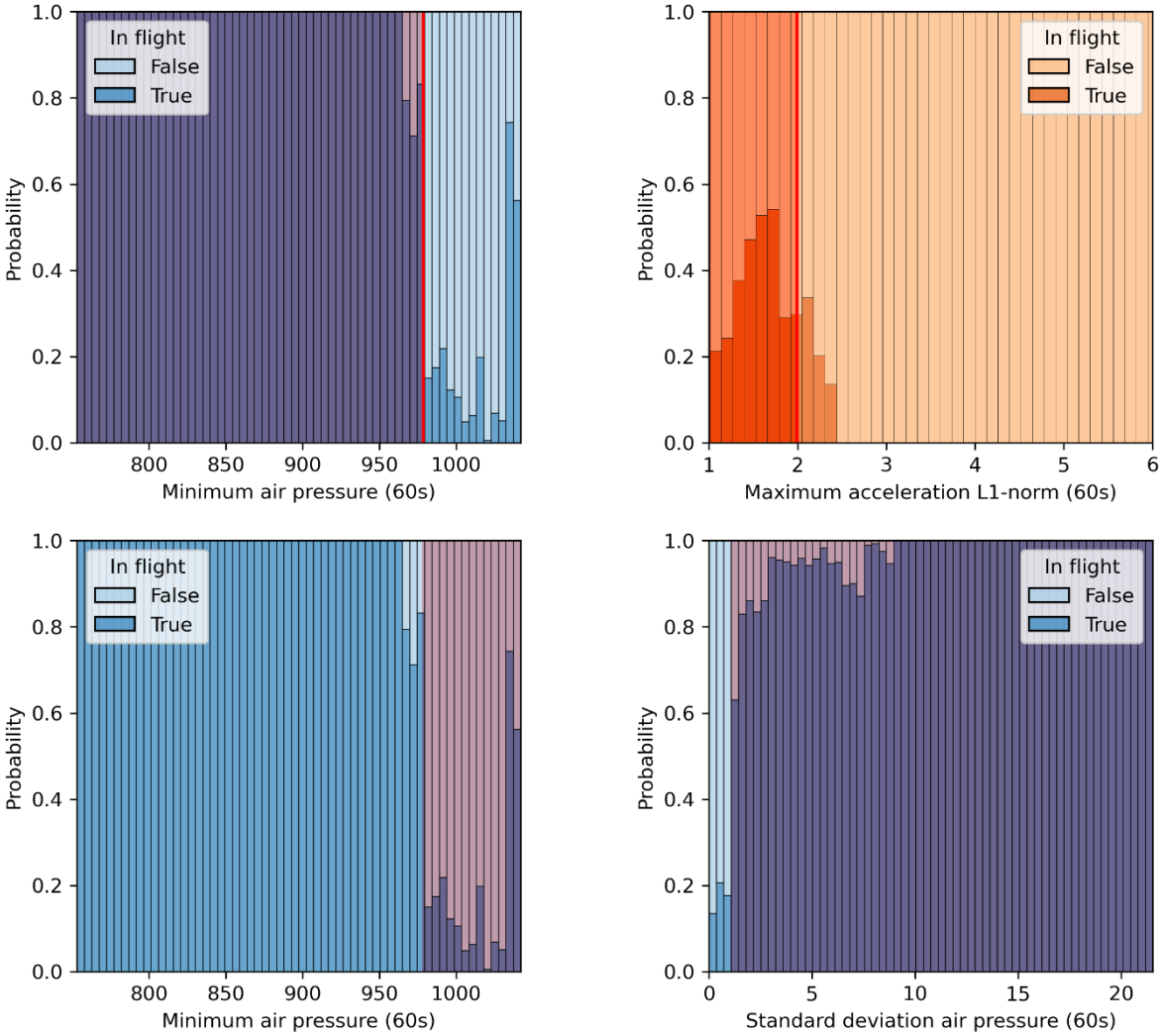


Figure 31: The best features and thresholds found for flight detection based on a decision tree classifier. "In flight" is predicted if at least one of these two statements is true with respect to the 60s time window prior: The minimum air pressure ≤ 978.5mbar (top, left) and the maximum acceleration L1-norm ≤ 1.9915g (top right); or the minimum air pressure > 978.5mbar (bottom, left) and the standard deviation of the air pressure > 1.0819mbar. If neither of these two is true, "Not in flight" is predicted. (Thresholds are shown as red overlays).

This method achieved a **ROC AUC score of 98.8%**. It correctly predicts "In flight" for 98.4% of all samples (recall), and the correct prediction of "Not in flight" for 95.0% of all samples (precision), see Figure 32.

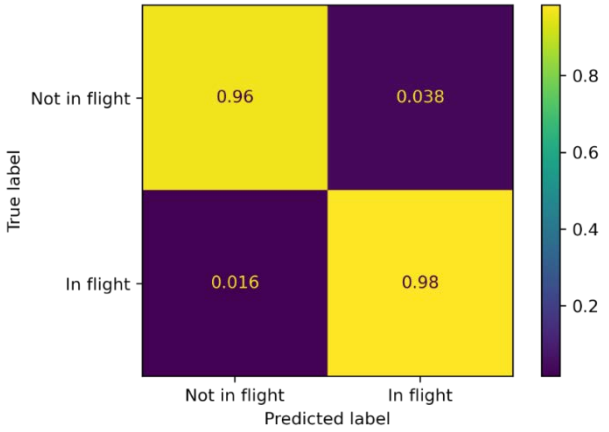


Figure 32: (Cross-validated) confusion matrix for the final flight detection method.

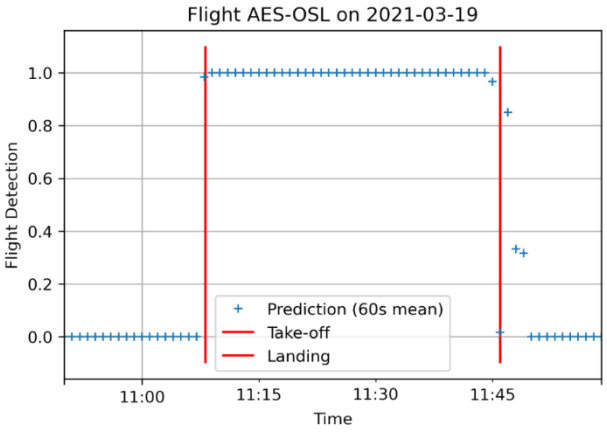


Figure 33: Flight detection for a flight from AES to OSL. Predictions are averaged over 60s time windows for better visualization.

6.2 Method Based on Accelerations Only

For investigation and as a backup, we also developed a method based on 3D accelerometer sensor readings only. The best performance was reached with a combination of interquartile range calculated from the L2 norm and standard deviation calculated from the L1 norm of the acceleration over the prior 60 seconds.

Code 2: Alternative method based on accelerations only implemented in Python. It is based on the interquartile range of the acceleration L2-norm ("acceleration_norm_l2_iqr") and the standard deviation of the acceleration L1-norm ("acceleration_norm_l1_std").

```
def detect_flight_acc_only(acceleration_norm_l2_iqr, acceleration_norm_l1_std):

    in_flight = (acceleration_norm_l2_iqr > 0.0059) & (acceleration_norm_l1_std <=
    0.0776)

    return in_flight
```

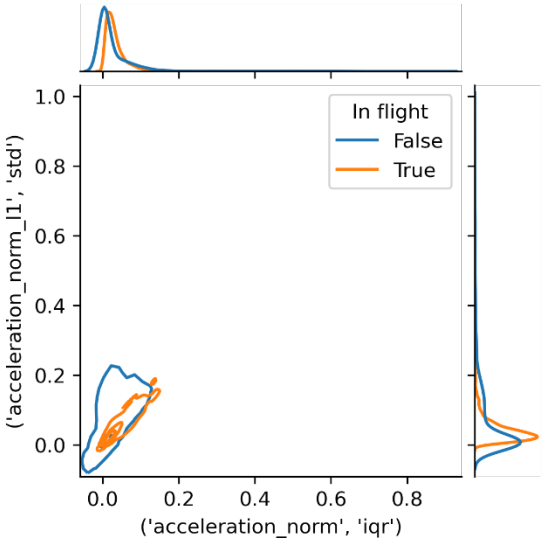


Figure 34: Contour plot for the feature combination that was identified as most useful for the detection of flight based on 60s time slices when only using accelerations: Interquartile range of the L2-norm and standard deviation of the L1-norm. (In units of 1g)

As expected, it performed significantly worse than methods also using air pressure readings. It achieved a **ROC AUC score of 88.0%**. It correctly predicts "In flight" for 85.9% of all samples (recall), and the correct prediction of "Not in flight" for 80.1% of all samples (precision), see Figure 35.

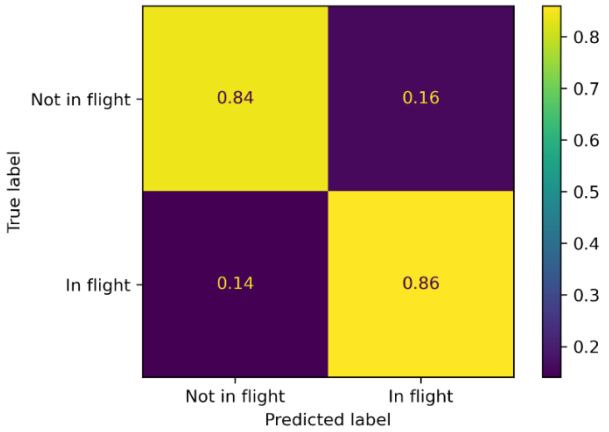


Figure 35: (Cross-validated) confusion matrix for the flight detection method based only on acceleration signals.

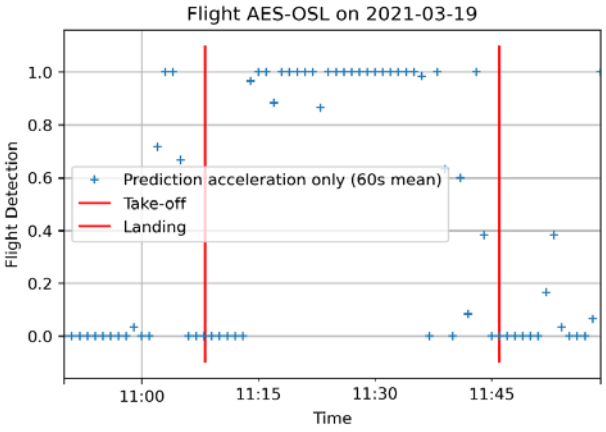


Figure 36: Flight detection based only on acceleration for a flight from AES to OSL. Predictions are averaged over 60s time windows for better visualization.

7 Considerations and Future Work

7.1 Risk Factors

7.1.1 Overfitting

The **largest risk factor** is overfitting. As mentioned in Section 3, the data available for analysis and method development is heavily biased in favor of only 4 airports and only 2 different aircraft types. Moreover, all data was collected between October and March, and the local buffering issues described in Section 3 restricted flight lengths to 4500s. Because of this, the risk that the analysis and method work significantly

better on the available data as compared to yet unseen data must be assumed high. Ideally, both should be extended and updated to include more airports, more aircraft types, different seasons, different routes, and different flight durations. It is highly recommended that data on at the very least 100 flights are collected and used to cover Norway alone, with a balanced variation with respect to aforementioned factors.

To mitigate overfitting risk, we chose several **counter-measures**: A strong preference for simple methods to keep complexity low, the diligent use of cross-validation, and reliance on domain knowledge and simple physics.

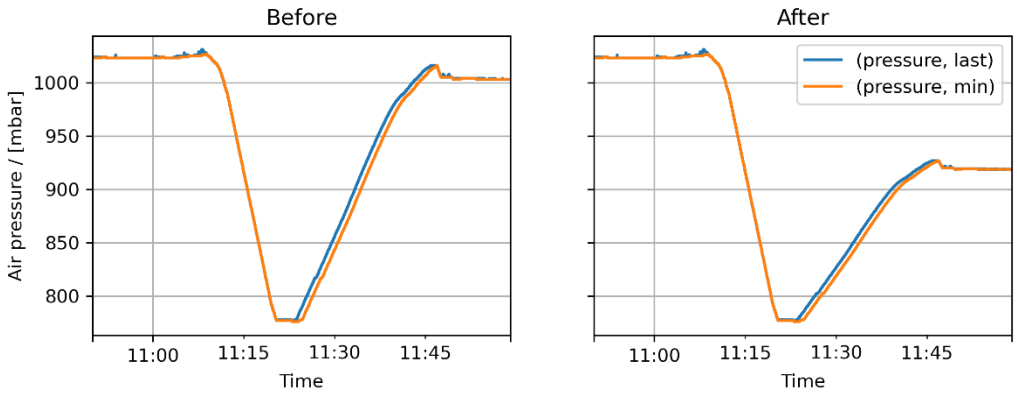
7.1.2 Use on Routes with Higher Ground Altitudes

A likely issue related to overfitting (see Section 7.1.1), is the use for airports that are significantly higher than the 4 airports for which data was recorded (see Table 2). With higher altitude, air pressure decreases. Geilo is the airport with the highest altitude in Norway at 798m above sea level and air pressure will often be well below the threshold used in the flight detection method described in Section 6.1.

Because the method also uses air pressure variation and accelerometer data, this does not necessarily mean that it will fail for higher altitude airports. But it means that its performance is only **validated for airports with up to 208m in altitude** (and only for the weather conditions represented in the available data between October and March).

One possible way around this issue is the use of relative air pressures dependent on the airport altitudes for the specific flight route. As per 2021-05-12, BagID is testing such an approach, but it should be validated properly with the methodology described in Section 5.3. Alternatively, all air pressures could be normalized to sea level using airport altitudes.

We also performed a preliminary check for which we scaled the air pressure during descent such that it rises to a lower value of 920mbar after landing (the approximate air pressure level for Geilo airport). The final flight detection method described in Section 6.1 correctly predicts "In flight" for 100.0% of this artificial flight data (recall), and "Not in flight" for 84.5% of it (precision). So, while by no means conclusive or evidential, this indicates that the method's ability to detect flight and turn off mobile communications in accordance with the regulatory guidelines **may be robust to lower air pressures at higher airport altitudes** (though it also indicates that it makes it harder to detect when to turn them back on again). Figure 37 illustrates the results of this test.



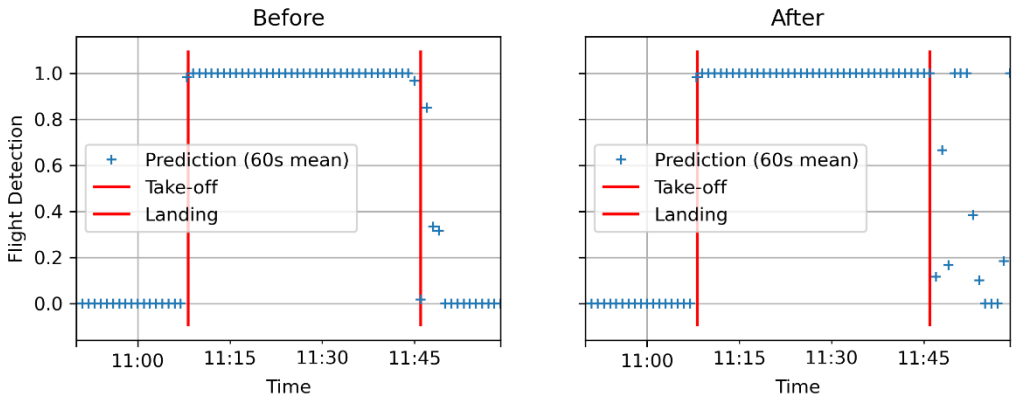


Figure 37: Performance of the final flight detection method for an artificially modified descent to an airport with higher altitude and lower air pressure levels. Original air pressure (top, left) and "In flight" prediction for the original data (bottom, left) and modified air pressure (top, right) and "In flight" prediction for the modified data (bottom, right).

Finally, weather conditions can also influence air pressures and influence the performance of flight detection based on pressure. We concluded with that this effect, although certainly present, is comparatively small compared to the effect of altitude changes.

7.1.3 Air Pressure Regulation on Aircraft

Another likely risk factor is how various aircraft types regulate cabin and luggage compartment pressures. The patent that Samsung filed in 2014 [6] for an automatic flight detection method was based on a library of cabin pressures for various flight types. Again, more data should be collected to better understand to what extend this can cause issues and how to respond to it, if it does.

7.1.4 Missing Data and Sensor Errors

It is very common to have methods fail when sensors or other parts of the hardware and firmware pipeline are malfunctioning. The project was missing some sensor data for some of the flights. Designing reliable and potent fault detection methods is a formidable task in itself, but the implementation of simple sanity checks is still highly recommended.

7.2 Improvements

The easiest way to improve on the existing method is to **collect more data** and repeat the steps of the analysis and the methodology described in Sections 4 and 5. This will make it possible to find decision tree thresholds that work even better, or it may also give better sensor signal combinations and processing steps.

In addition, several other attempts can still be made to improve on the method, for example:

- A closer look at the 3D accelerometer signal decomposition into horizontal and vertical acceleration components described in Section 4.2.1.
- Investigate the use of flight phase predictions made earlier during a journey or the use of lagged features.
- The decision tree classifier's ability to predict probabilities can be used (instead of predicting classes directly). By varying the associated probability threshold, the detection can be tuned to increase either the ability to correctly identify "In flight" or the ability to correctly identify "Not in flight" (but not both), see Figure 38. This should only be done after (much) more data is acquired, ideally.
- The use of higher sampling frequencies for the accelerations (at least 50Hz, see Table 4) to possibly identify aircraft engine signature vibrations (as done in [4]). Figure 39 shows an example for an acceleration spectrogram that shows various signal signatures during flight.

- Using different sample weights when training the decision tree classifier to counteract the bias towards flight scenarios that may be overrepresented in the data (and may cause overfitting).
- Formulating the problem in terms of a multi-category classification to study and detect various flight phases (acceleration, ascent, descent, deceleration) in more detail. Some work on this was done during the project to detect take-off and landing events directly.
- The use of a microphone could help pick up engine sounds, which in turn should be valuable for automatic flight detection.

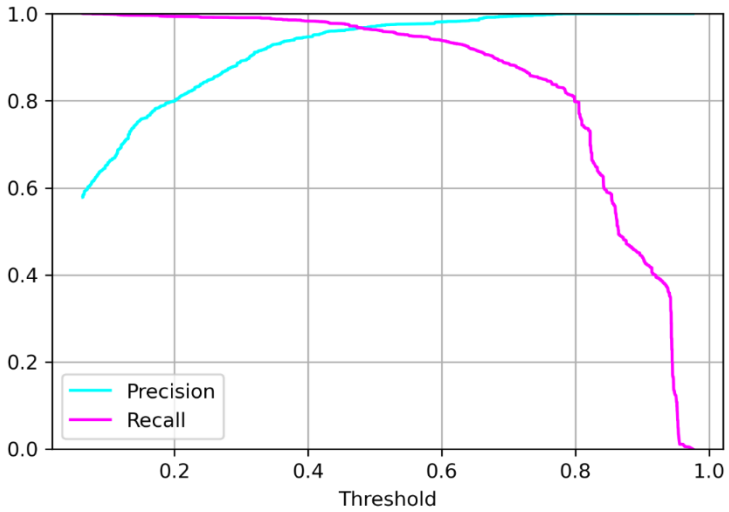


Figure 38: Example for a precision-recall curve that can be used to choose a threshold for which the desired balance between precision ("Not in flight" is predicted correctly) and recall ("In flight" is predicted correctly) is obtained.

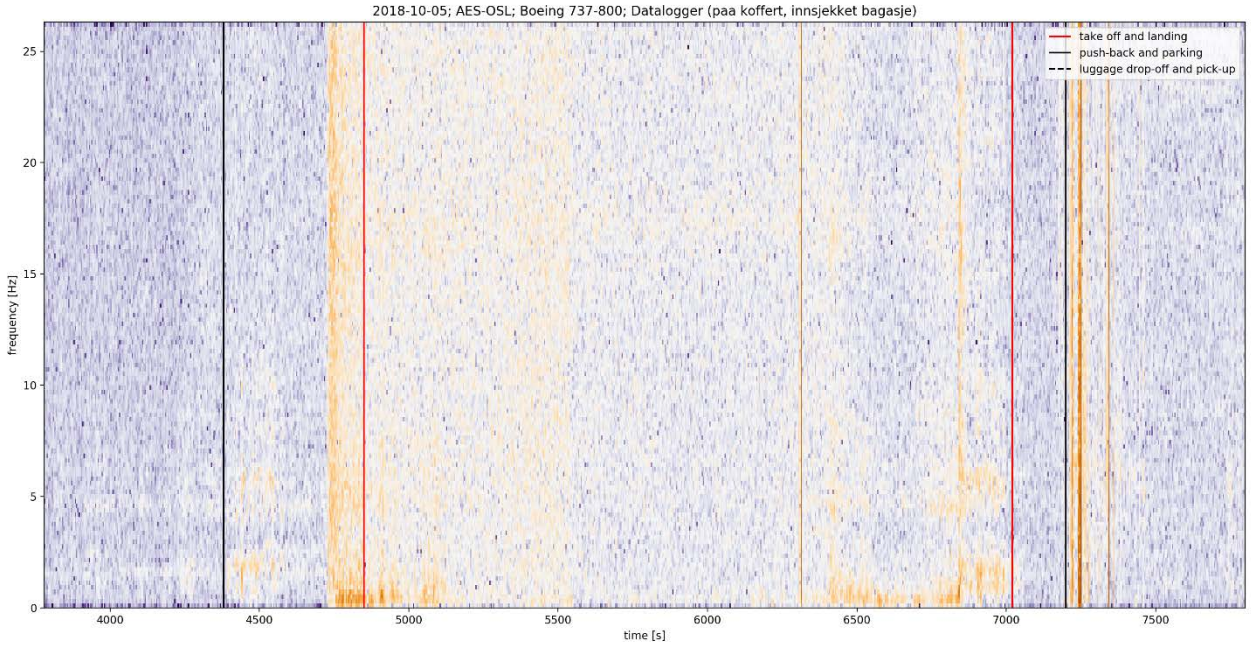


Figure 39: Example for an acceleration spectrogram (L2-norm) obtain in 2018 on a flight from AES to OSL [7].

8 References

[1] Electronic Bag Tag sub-working group, "Electronic Bag Tag (EBT) Implementation Guide," International Air Transport Association, 2018.

[2] Federal Aviation Administration, "Use of Portable Electronic Devices Aboard Aircraft 91.21-1D," U.S. Department of Transportation, 2017.

[3] MSR Electronics GmbH, "MSR145: Mini Data Logger for Temperature, Humidity, Air Pressure, Fluid Pressure, Light, Acceleration," MSR Electronics GmbH, [Online]. Available: https://www.msr.ch/en/product/datalogger-temperature-humidity-pressure-acceleration-msr145/. [Accessed 23 5 2021].

[4] Y. Tawk, A. Jovanovic, P. Tomé, J. Leclère, C. Botteron, P.-A. Farine, R. Riem-Vis and B. Spaeth, "A New Movement Recognition Technique for Flight Mode Detection," *International Journal of Vehicular Technology,* p. 18, 2013.

[5] M. J. Wengler, "In flight detection". US Patent US8380458B2, 2013.

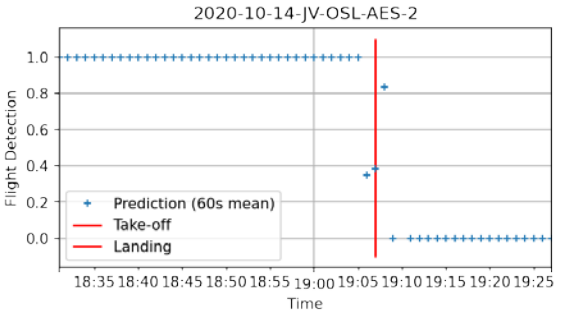
[6] W.-U. Kwon, "Method and apparatus for switching operation mode of mobile phone". US Patent US20140308940A1, 2014.

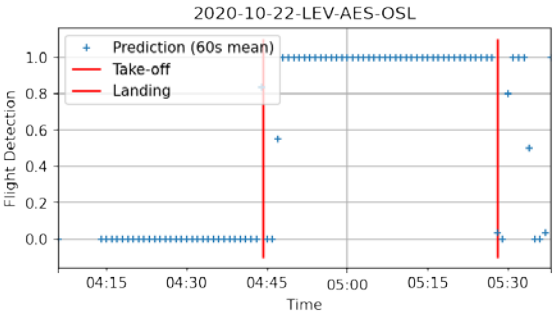
[7] SINTEF Ålesund, "Vurdering av sensorteknologi til bruk i smarte bagasjetag," Ålesund, Norway, 2018.

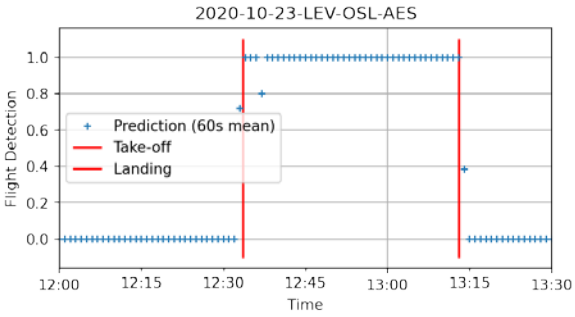
[8] X. Zhang, Y. Wong, M. S. Kankanhalli and W. Geng, "Hierarchical multi-view aggregation network for sensor-based human activity recognition," *PLoS ONE,* vol. 14, no. 9, pp. 1-20, 2019.

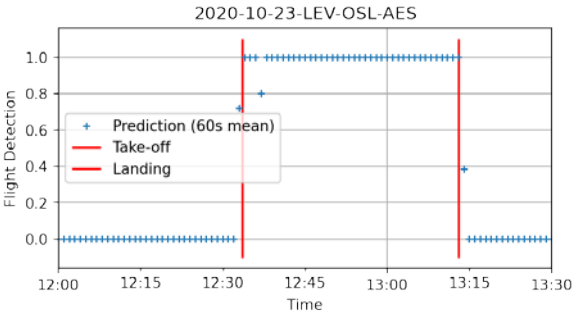
A Predictions for All Datasets

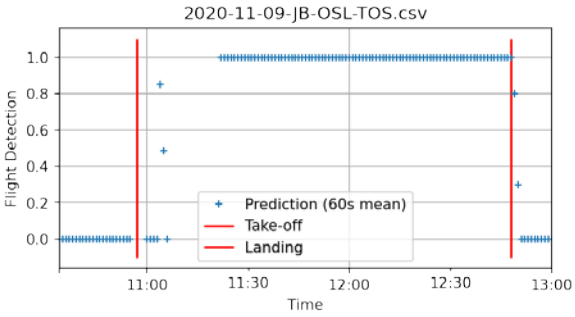
The following plots show the flight detection performance for all recorded flights and data sets. Predictions are averaged over 60s time windows for better visualization. Note that several flights lack data points (in which case no prediction is plotted) and recordings during take-off or landing phases.

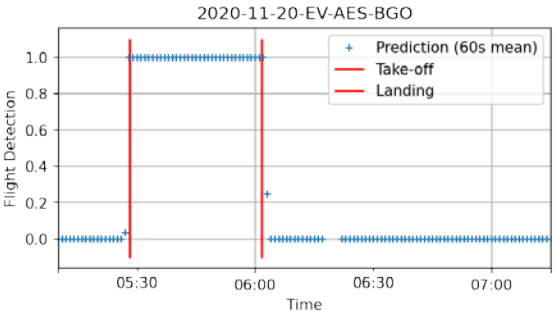


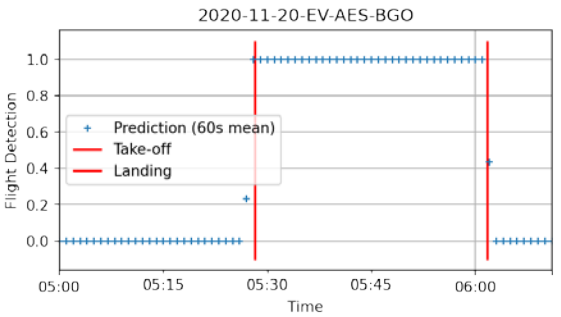


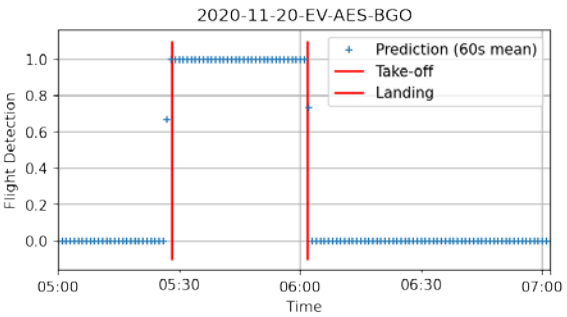


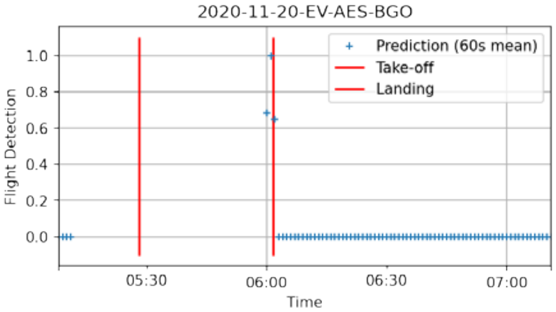


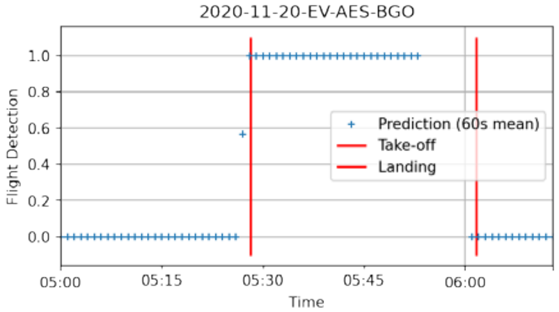


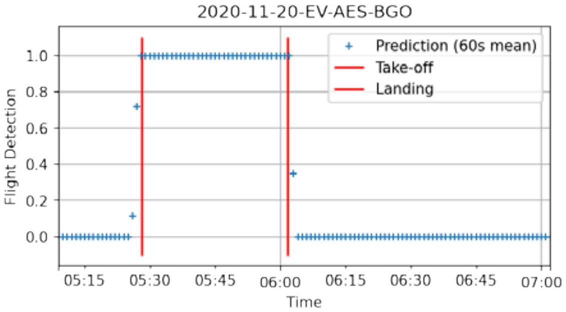


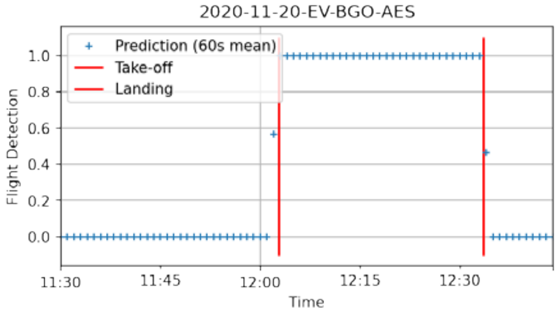


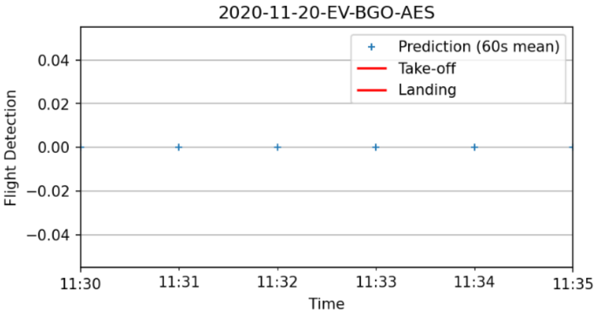


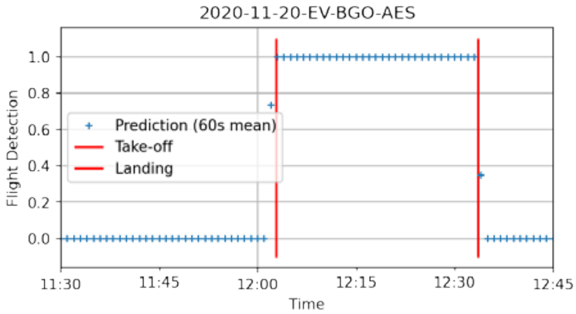

2020-11-20-EV-AES-BGO
Flight Detection
Time
Prediction (60s mean)
Take-off
Landing

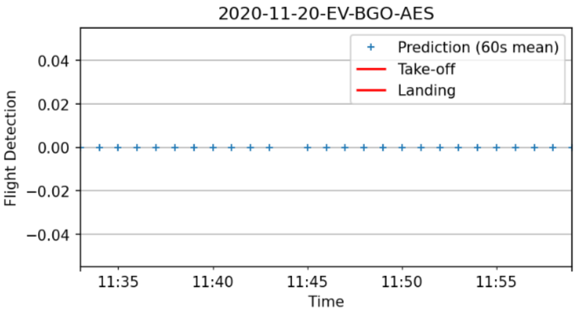

2020-11-20-EV-AES-BGO
Flight Detection
Time
Prediction (60s mean)
Take-off
Landing

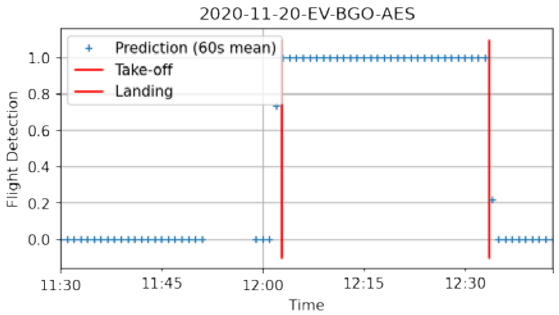

2020-11-20-EV-AES-BGO
Flight Detection
Time
Prediction (60s mean)
Take-off
Landing

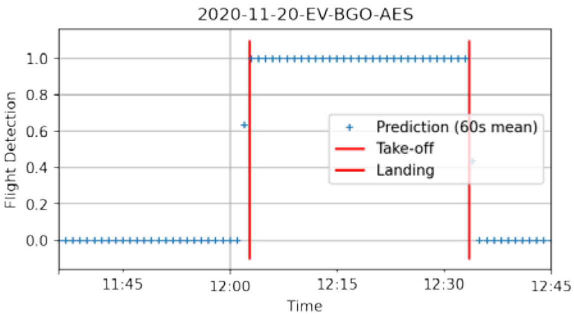

2020-11-20-EV-BGO-AES
Flight Detection
Time
Prediction (60s mean)
Take-off
Landing

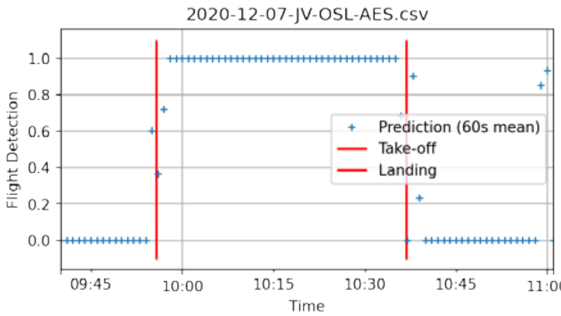

2020-11-20-EV-BGO-AES
Flight Detection
Time
Prediction (60s mean)
Take-off
Landing

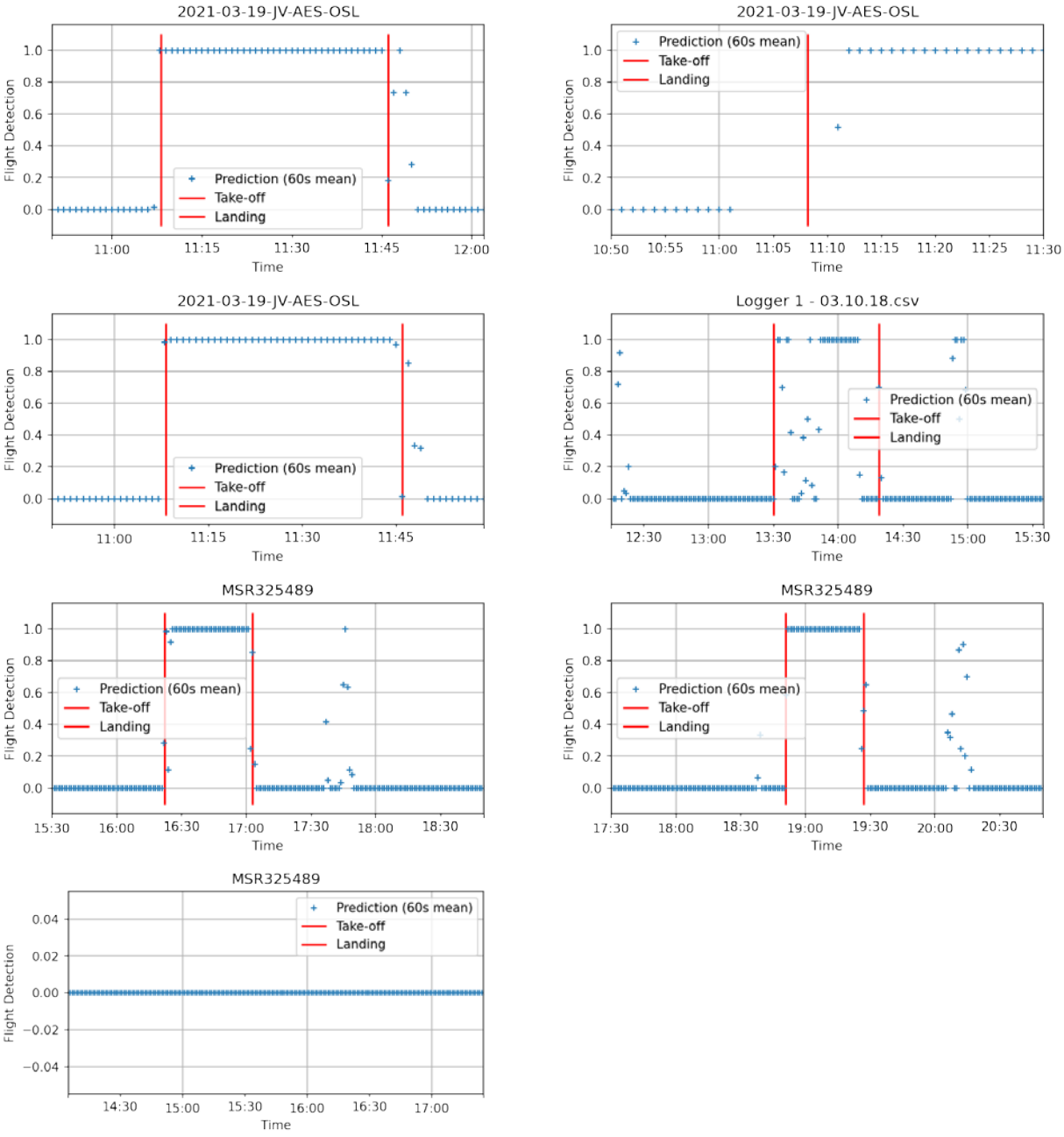

2020-11-20-EV-BGO-AES
Flight Detection
Time
Prediction (60s mean)
Take-off
Landing


2020-11-20-EV-BGO-AES
Flight Detection
Time
Prediction (60s mean)
Take-off
Landing


2020-11-20-EV-BGO-AES
Flight Detection
Time
Prediction (60s mean)
Take-off
Landing


2020-11-20-EV-BGO-AES
Flight Detection
Time
Prediction (60s mean)
Take-off
Landing


2020-12-07-JV-OSL-AES.csv
Flight Detection
Time
Prediction (60s mean)
Take-off
Landing



B Correspondence with the Federal Aviation Administration

B.1 Definition of Flight Phase

E-mail sent from Brian Verna (FAA) to Severin Sadjina (SINTEF Ålesund) on 08.11.2018 about the definition of "airborne" with respect to mobile communications in PEDs.

*I would use the **takeoff and landing phases of flight as the last possible conditions to detect turning off and back on again**. Detecting conditions prior to takeoff and after landing is also acceptable.*

Brian Verna
Aerospace Engineer

Federal Aviation Administration
Flight Standards Service
Aircraft Maintenance Division (FS-300)
Avionics Branch (FS-360)

B.2 Method Implementation

E-mail sent from Brian Verna (FAA) to Eivind Vinje (BagID) on 16.12.2020 documenting how the method described in Section 6.1 fulfills the Federal Aviation Administration (FAA) recommendations.

Q1: Are we understanding the sensor/method redundance correct that only one sensor/method is needed to enable "flight mode", while we need two sensors/methods to disable "flight mode"?
A1: This design element of how you are accomplishing the use of the sensors is not too particularly important. The recommendation for two independent means to turn off intentional transmissions is ***achieved by your description****. The idea originates from the fact the FAA does not have regulatory authority over the design of portable electronic devices, so we cannot require a design feature for a certain failure condition. Most PEDs use COTS hardware and are not design and produced at the same pedigree as aviation software/hardware. Therefore, the FAA recommends the two independent means, if one fails, then the other can reliably perform the function. This gives us the safety assurance needed for aircraft operators to adopt this policy, or choose another policy. This decision is completely up to the airline.*

Q2: If we detect that one of the sensors is defect (pressure or accelerometer), can we disable that part of the algorithm and use scheduled departure/arrival time as one of the triggers to "flight mode"? We are pulling scheduled departure/arrival time every 10 minute from the device, so these will be updated in case of delays.
A2: Again, I think ***your concept meets the criteria*** *for dual independent means, and your offering a third independent means for when one of the two sources fails is definitely sufficient. I would even say if your device detected one source failed, then relying on the other source 100% would meet the criteria. This gives credence to our recommendation to not relying on one means to shut off intentional transmissions before flight.*

Q3: Is it OK to use three sensors to detect landing instead of two, and if two of them are detecting landing we disable "flight mode"?
A3: Same as above. I really like the concept and it is creative. However you choose to implement the turning on and off of the intentional transmitter is up to you. I think this ***builds extra reliability and adds an extra layer of safety that many airlines would be pleased with****.*

I hope this helps you out going forward. Please don't hesitate to reach out if you have any more questions or concerns.

Brian Verna
Senior Technical Advisor

Federal Aviation Administration
Flight Standards Service
Office of Safety Standards