Lucerne University of Applied Science & Architecture

ARIS - Data Fusion for a Sounding Rocket

Bachelor Thesis

Author:	Michael Kurmann
Supervisor:	Prof. Marcel Joss
Expert:	Werner Scheidegger
Industrial Partner:	ARIS (Akademische Raumfahrt Initiative Schweiz) Oliver Kirchhoff
Submission date:	22.12.2017
Classification:	Access

Declaration

Hereby, I declare that I have composed the presented paper independently on my own and without any other resources than the ones indicated. All thoughts taken directly or indirectly from sources are properly denoted as such. This paper has neither been previously submitted to another authority nor has it been published yet.

Horw, March 26, 2018

Abstract

This is the Abstract

Contents

1 Introduction **2**
1.1 Purpose . . . 2
1.2 Research . . . 2
1.3 Problems . . . 2
1.3.1 Different Sensors . . . 2
1.3.2 System Load . . . 2
1.3.3 Precision . . . 3
1.3.4 Settling Time . . . 3
1.3.5 Reliability . . . 3
1.3.6 Modularity . . . 3
1.4 Requirements . . . 4
1.5 Desired Solution . . . 4

2 Approach **5**
2.1 Tipps . . . 5
2.1.1 Barometer . . . 5
2.2 Possibilities . . . 5
2.3 Choosing . . . 5
2.4 Explenation . . . 5

3 Implementation **6**
3.1 Sensor Model . . . 6
3.2 System Model . . . 6
3.3 Simulation . . . 6

4 Tests **7**

5 Conclusion **8**
5.1 Thanks . . . 8

Appendices **10**

Chapter 1

Introduction

Here comes the Introduction what is Aris what are the aims and why is a Data/Sensor fusion needed?

The Academic Space Initiative Switzerland (ARIS) is student group which tries too compete in the yearly Intercollegiate Rocket Engineering Competition (IREC). To aim for the right apogee (3000 m) a Control algorithm is implemented. This algorithm relays on the information of different sensors to determine the rockets actual state. Cause there are different sensors to measure the same value a algorithm which fuses those data would come in handy. With this fusion algorithm it should also be possible to be more accurate as with each sensor on its own. For this the problem as well as the desired solution will be defined in this chapter. After that the dynamics of the rocket as well as the parameter of the different Sensors are defined at beginning of chapter 2, the most suited algorithm will be chosen. Chapter 3 will then describe how this fusion algorithm implemented in a simulation in detail. To verify that the implementation is working as intended, the fused data will be held against the ground truth which are provided from this years test flights in chapter 4. In the last chapter 5 a summary of the achieved knowledge will be stated. The purpose of this thesis is to find and implement the algorithm which is most suitable for this task.

1.1 Purpose

The hardware as well as the most of the software parts that will be used for this competition is already defined. Also it is a suitable assumption that the sensors and the dynamics of the rocket will be stay more or less the same for the competitions coming. Therefore this thesis will mainly focus on finding a algorithm for this given surroundings, but it is also will try to find as modular solution as possible, so that achieved knowledge can be used in further competition.

1.2 Research

Write about the Papers/book you used: Kalman-filter Optimal state estimation The Master Thesis Sensor/-data fusion is an engineering field since the first rocket flights. Therefore there is already a lot of previous work which can be used in this thesis.

1.3 Problems

Out of the research and the previous competition, different problems appeared that need to be addressed in this thesis to ensure a good solution as possible.

1.3.1 Different Sensors

First of all there are different sensors which all do measure different values and have different parameters (precision, sampling time). So the algorithm has to be play out the strength of the different sensors to uneplay their individual weaknesses. Additionally, because this algorithm is system critical, it has to be reliable enough that it still is working properly if sensors failing.

1.3.2 System Load

The cycling time will be around 1 ms on a embedded system. This time was chosen on the behalf that it would be difficult to get the exact needed cycling time on ensure the needed controlability of the rockets

apogee. Therefore the system load that the algorithm can cause, has to be strongly limited, so that it can be run on this given system. The system this year is an 32 bit Arm Processor which runs on 168 MHz, assumed that the algorithm has at maximum the half of a software cycle, the maximum given clock cycles are around 84 000. This number should not change in great manner over the next competitions.

1.3.3 Precision

The Precision is after the system load the most critical attribute, if the algorithm does not get into the required accuracy the whole thing is more or less for nothing. The Control stated that the maximal error between the estimated and ground truth height should not exceed two meter. This accuracy is needed to proper control the aim of the apogee.

1.3.4 Settling Time

The settling time defines the time span when the first reliable measurements arrive after burnout until the estimation is into the required precision. This time span has to be small enough to ensure that the controlling has enough time to aim for the desired apogee. In the current system the burnout occurs occurs around 3-3.5 seconds after ignition, whereas the whole flight upwards only takes around 23 seconds. Therefore the settling time needs to be around just one second so that the control has as much time as possible for the controlling.

1.3.5 Reliability

Due to given surroundings that come if a sensor package is placed into a rocket, the assumption has to be made that it will be possible that sensors fail in execution. Therefore the algorithm should provide the reliability of still working in a proper manner with some sensors failed. So that the execution of the controlling software in terms of functionality, but locally it has not to be as accurate as it would be with all sensors working.

1.3.6 Modularity

Although it can be assumed that they will stay more or less they same over the next competitions, it is not ensured that exactly this sensors will be used. Therefore the presented algorithm should provide the possibility to exchange the sensors, as long as they resemble the old sensor in a feasible way. This will ensure a long term use of the provided algorithm.

Problems found so far:

- How to calculate Height out of Pressure/Temp/Humidity Fabian version: $44330 * (1 - (\frac{pressure}{101325})^{\frac{1}{5.255}})$
- How to parameterize the different sensors (Measuring, Test Flight, Data Sheet) ?
- How to fuse together Data from Sensors that have different Taus, especially those who are slower than the Loop-Time ?
- How to integrate AirBreaks/Drag Force of Air/ Trust of Motor a input value?
- What are the different noise factors and when do they occur ?
- The up-flight is rather short: about 25 seconds, so the Fusion should have a small settling time
- The Micro-Chip on which it is used is no the fastest : 168 MHz clock
- The Ram on the Chip is not endless: Maximal space for the Sensor fusion is about 10kB
- The Sensor Fusion should be as modular as possible so that it also can be used in the next competition
- The Sensor Fusion has to be as sturdy as possible so that it will not fail if a problem occurs
- The Fusion should make a state Estimation as precise as possible.
- There are a lot of different variables: 3 Positions, 1 Speed, 3 Accelerations, 3 Lagen, Time, Pressure, Tempterature, Humidity, Up-/Downforce.
- Especially the Input Value u which is the force onto the rocket is difficult to define (Drag, Trust = acceloration depends on wheigt which changes over time).

- The different Sensor have different weaknesses:
  - Accelerometer: Offset, drift, weak to vibrations
  - Gyro: Weak to Vibrations
  - Barometer: Many uncertenties, unpercise
  - GPS: Slow (max 5Hz)

1.4 Requirements

Requirement	**Rating**	**Aim**	**Importance**
System Load	# Calculation steps per loop	< 1000	Critical
Precision	Error between estimation and ground truth	$<$ 2m in Z	High
Settling time	Time from first reliable to optimal estimation	$<$ 1 s	High
Reliability	Functioning Estimation with #failed sensors	2-3 sensors	Medium
Modularity	Effort needed to change a sensors	$<$ 10 h work	Desirable

Table 1.1: Requirements table

As seen in the table 1.1 five requirements were drown out of the problem analysis. First of all, there is critical requirement the system load. This is given as critical cause it is needed that the algorithm is small enough to be run on a embedded system. Any solution that would not fit this requirement would be pointless in the frame of this thesis. Secondly there are two requirements which are tied together, the precision and settling time. Where the precision describes what a optimal estimation is under the context of this thesis, the settling time relies on this to be defined. The desired precision will be needed to ensure a possible good control.

1.5 Desired Solution

The desired solution should met the given requirements as optimal as possible. While doing this it should also not be more complicated than needed to make a future use as easy as possible. Cause of this the modularity is an important requirement to ensure this.

Chapter 2

Approach

How I want to get to the Solution and why I choosed it also how it works.

2.1 Tipps

- Don't forget point of origin / reference system - Don't forget the pitch angle

2.1.1 Barometer

-Use density as a statevector
-Use exponential atmosphere method.
-Increasing the R measurement noise matrix when rocket is ascending access the rising uncertainties

2.2 Possibilities

Which ones are there and what are their positives and negatives, in short how do they work I will use kalman filter with time depending system and sensor noise, but i have yet to define how i do this in peticullar. I should also make some pictures and such things.

2.3 Choosing

Why i choosed Kalman Filter

2.4 Explenation

How it 'should' work in detail

Chapter 3

Implementation

How it will be Implemented

3.1 Sensor Model

3.2 System Model

3.3 Simulation

Chapter 4

Tests

Here come the tests

Chapter 5

Conclusion

5.1 Thanks

Thanks to the Aris Team espacially Thomas and Fabian Also Thanks to Lukas and Jossely

Bibliography

Appendices

This is the Appendix