Classification of Structural Failure for Multi-rotor UAS
CS289A Final Project

Chris Echanique, Sunil Shah

May 2014

Abstract

This paper covers our investigation into the use of machine learning techniques to classify 350 samples of accelerometer data collected from a multi-rotor unmanned aerial system. We use the Fast Fourier Transform of the data to produce a spectrum in the frequency domain. We then investigate various features that may be used to characterise this data efficiently and apply various standard classification algorithms to the dataset. In the end, we gain accuracies of xx% when using binary classification and xx% when using multi-class classification.

1 Introduction

There has been a sharp increase in the popularity of unmanned aerial systems (UAS) that are built and flown by hobbyist pilots. The popularity of these aircraft has been driven by three underlying trends: 1) the growth and popularity of open source hardware and software (e.g. the Arduino platform or ROS, the robotics operating system), 2) the maker movement (e.g. 3D printing) and 3) the availability of low cost sensors (e.g. the inertial measurement units which are used by nearly every smartphone). These allow hobbyists to assemble, economically, aircraft in their own homes. The most popular type of aircraft design amongst hobbyists is the versatile multi-rotor vertical take off and landing aircraft. These typically have 3 or more arms attached to small electric motors. Control is similar to a helicopter but, coupled with an autopilot running a basic loop feedback controller, is easier for an untrained pilot.

The inherent versatility and low cost of this design is disruptive to the status quo of large, military grade UAS. Thus, as these aircraft have become more capable and robust, there has been an increase in the number of commercial entities formed who are attempting to build products and services based around these low cost vehicles. While commercial operation of these aircraft is currently prohibited, there is a federal mandate for the Federal Aviation Administration (FAA) to have rules permitting businesses to operate these in place by 2015. However, air travel within the national airspace is incredibly safe - due to the strict regulations and rules that must be followed by entities operating in all but the least congested class of airspace. It is unlikely that the legalisation of UAS will come without similarly restrictive requirements surrounding safety and operation.

This is directly at odds with the *hacker* mindset that is used to build disruptive technologies such as this. While the use of open source software and hardware has accelerated the development of new features, the fast pace of change arguably leaves these systems open to complex failures that, in the context of a high powered flying aircraft, can often have extreme, undesirable, collateral damage. Therefore, we attempt to use our newly acquired machine learning knowledge to further the development of a structural health monitoring system for UAS that can use this data to, at the very least, ascertain whether or not it is safe to fly.

We use data collected as part of a Master's project where a system using accelerometers mounted to the arms of an accelerometer was built. This approach is used for other complex mechanical systems such as jet engines, helicopter gearboxes and various large scale industrial motors but has not been used for a small UAS previously. The frequency-domain data produced by this system characterise vibration in three dimensions and show, clearly, the effects of mechanical failure.

Section 2.1 gives a brief overview of the system itself and section 2.2 characterises the data that was collected. In section 3 we describe the various feature vector components we considered and in section 4 we describe the various approaches we took to classify our data. Our results are discussed in section 5 and we conclude with an overview of the limitations of our project and suggestions for future work in section 6.

2 Data Collection

Despite the multitude of sensors onboard current UAS designs, there is actually very little data reported back which can be used to detect problems with the UAS. Sensors typically detect the position of the aircraft, the current battery voltage and the status of the radio link. Fail-safes can be triggered in the autopilot which cause the UAS to return to a known *home* location in various error conditions - such as a low battery, lost radio link or if the UAS breaches a pre-programmed geographic *fence*. In order to detect mechanical failures in flight, additional data is needed.

2.1 System Design

Accelerometers are used here since the data they provide is the most versatile and can be used to characterise multiple types of mechanical failure. We consider a handful of failures here based on the typical construction of a multi-rotor UAS. As mentioned previously, these have a number of arms which are connected by a central plate (upon which the autopilot and radio hardware are usually affixed). At the end of each of these arms is an electric motor, typically brushless, that is attached to a propeller. Any single component just described can fail without collision. For instance, propellers may be chipped or any of the screws that are used to connect components together may work themselves loose. This results in inadequate control performance and, at worst, may cause the aircraft to stop flying altogether. Each of these failures causes various forms of vibration which is easily picked up by the accelerometer.

Figure 1 shows this system in action. In this initial version of the system, collected data is transmitted wirelessly to a laptop which logs the data for later processing. Preferably this data would be fed straight into the autopilot where it could be tested against when arming or disarming the vehicle.

2.2 Data Description

The data used for this project was collected from a single UAS arm with a motor attached. This arm was

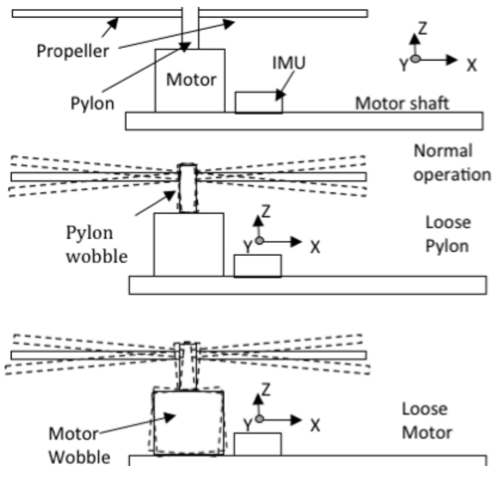

Figure 1

tethered to a bench. The motor was made to spin up under various normal and erroneous conditions and data was collected over a 10 second interval. The pulse width was also varied (the amount of time each motor is powered on for) from 900 μs to 2300 μs. The higher this time, the more power is supplied to the motor and the faster it spins. Data is collected for when there is:

- no propeller (unloaded)
- a propeller (loaded)
- a propeller mounted on a loose pylon
- a propeller mounted on a loose motor
- a broken propeller

The accelerometers used operate at 1 KHz (i.e. 1,000 points a second). Therefore 10 sets of data of 1,000 points each were collected for each failure mode and pulse width combination. Each of these sets of data was run through the Fast Fourier Transform algorithm to convert the data from the time domain to the frequency domain. This gives us data representing the magnitude of vibration at various frequencies for the x, y and z directions.

Table 1 shows a summary of the data that was collected. Note that the struck out pulse widths are tests that could not be performed for safety reasons. All-in-all, we have 35 different combinations of failure mode (including normal operation - i.e., no failure) and pulse width, with 10 samples for each combination.

	No load	**Loaded**	**Loaded**	**Loaded**	**Loaded**	
	No issue	No issue	Loose pylon	Loose motor	Broken propeller	
Pulse width (us)	800	800	800	800	800	Low speed
	900	900	900	900	900	
	1600	1600	1600	1600	~~1600~~	Medium speed
	1700	1700	1700	~~1700~~	~~1700~~	
	2200	2200	2200	~~2200~~	~~2200~~	High speed
	2300	2300	2300	~~2300~~	~~2300~~	

Table 1: Overview of data collected

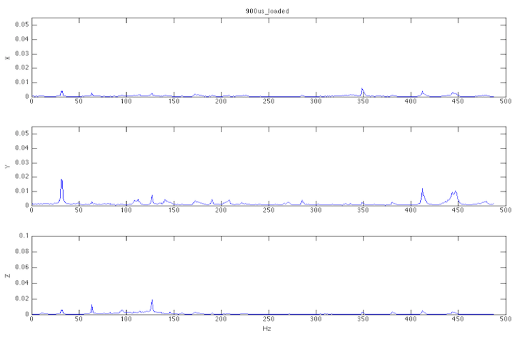

Figure 2: 900us pulse width no issue FFT

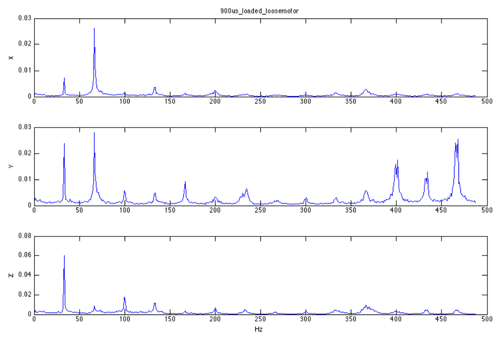

Figure 4: 900us pulse width loose motor FFT

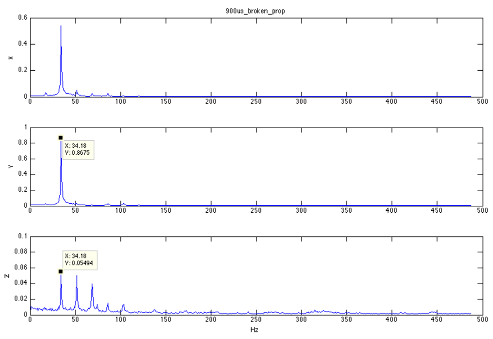

Figure 3: 900us pulse width broken propeller FFT

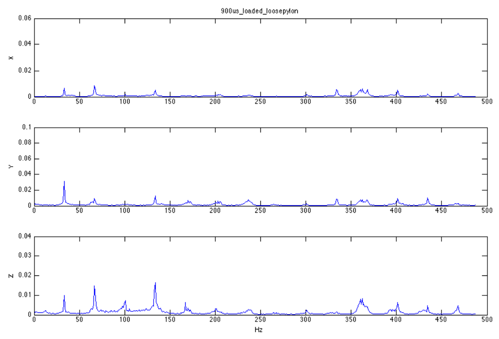

Figure 5: 900us pulse width loose pylon FFT

3 Feature Vector Design

The idea of performing classification based on frequency domain data is very well tested. Primarily the data used is derived from audio, where the computed FFT gives the frequency distribution of a sample of music or voice [4].

3.1 Preprocessing

Typically when classifying audio, it is necessary to preprocess the data to normalise for differences in volume (due to distance from the source) and to remove silence (for example, during speech). However, our data was collected in a controlled environment such that this it not strictly necessary - the accelerometer was a fixed and constant distance away from the source of vibration and the motor was on continuously for the duration of the test.

3.2 Dimensions

We considered working with x, y and z axis data - although give more weight to the y and z axes because vibrations typically happen laterally for each arm (i.e. y axis) or vertically (i.e. z axis).

3.3 Spectral Features

3.3.1 Mean Magnitude

We calculate the basic mean magnitude of our frequency spectrum which characterises how much overall vibration there is.

3.3.2 Peak to Mean Ratio

We use the ratio of the magnitude of the top peak to the mean magnitude as a feature. This lets us identify the relative divergence of the peak. For example, if the peak value is high but so is the mean value, this suggests that this is a very noisy set of data. However, if the peak value is high but the mean value is low, this suggests that the peak is exceptional. Thus a high peak to mean ratio should mark when there is a very strong vibration at a single peak frequency.

3.3.3 Number of Prominent Peaks

The number of prominent peaks is a potentially interesting feature to incorporate. We noticed that some of the erroneous conditions resulted a different number of peaks in the data. We calculated the prominent peaks by counting the number of peaks larger than three standard deviations from the mean of the data set.

3.3.4 Spectral Centroid, Spectral Rolloff

The spectral centroid is the magnitude weighted mean of the frequency spectrum for each signal (or alternatively known as "the centre of mass" of a spectrum. The spectral rolloff point is similar, except it considers the point at which 85% of the power is at lower frequencies, essentially measuring how much the power spectrum is skewed to the right. This feature has more pertinence for audio signals due to the way human ears work. We therefore chose to use the spectral centroid as a feature.

3.3.5 Spectral Bandwidth

This is the absolute difference between the highest and lowest frequencies in our frequency spectrum. In our data, the FFT output is fixed between 0 and 500 Hz and there is a modest amount of activity at each of these frequencies. This is most likely because the full spectrum is limited by the output of the accelerometer used and it produces noise at all frequencies. Therefore, we disregard this is a feature.

3.4 Sub-band Analysis

A common method of composing features for classification of audio is to perform sub-band analysis [1] - where the overall frequency spectrum is broken down into bands that are deemed pertinent.

3.4.1 Mel Frequency Cepstral Coefficients

Mel Frequency Cepstral Coefficients (MFCC) are regarded as one of the best performing ways to summarise audio that is classified by humans [1]. MFCCs are computed from the power cepstrum of a sound and involve taking sampling the original FFT at frequencies taken from the mel scale. The mel scale is a distribution of pitches that has been empirically determined to be considered as equally spaced pitch bands by humans. Intuitively, this would help when classification of the audio in question is down to the human listeners. However, for our particular problem, this is unlikely to work well since our classification does not depend on a human listener.

3.4.2 Spectral Histogram

For our particular data, we do not perform any specific sub-band analysis since the pertinent bands tend to shift with motor and propeller choice, as well as with the pulse width. We do however use a spectral histogram that summarises the full frequency spectrum. This allows us to reduce the dimensionality of our feature vector while still characterising the distribution for classification.

3.5 Temporal Features

It's worth noting that most feature vectors for audio use typically include some kind of temporal feature that "capture the temporal evolution of a signal" [1]. This is important for audio identification where the signal is not continuous and where the temporal aspect is crucial to proper identification or classification. However, since we again have a controlled set of data, we do not attempt to use this feature. Theoretically it might prove useful for situations where there is a sudden change, such as where a propeller strikes an object and suddenly comes to a halt. However, this is a rare occurrence.

4 Classification Algorithms

Based on common approaches to classification of frequency based data [1] [3] [2], we attempted to classify our data via support vector machines, K-NN classification, decision trees, random forests, and AdaBoost.

For simplicity, we primarily collected results for binary classification first (where 0 is working and 1 is not working) and then tried multi-class classification (for 4 distinct classes: working, loose pylon, loose motor, broken propeller).

4.1 Implementation Details & Cross Validation

The data was provided to us as multiple comma separated value text files. These were parsed and loaded into SQLite. We used the popular **scikit-learn** machine learning library in python.

Due to the lack of a separate test set, we used k-fold cross validation to gauge the accuracy of each of our approaches. The results below are shown for where $k = 5$.

5 Results

Table 3 shows our overall results. We managed to reach a maximum test accuracy of **99.6%** using a random forests classifier with 10 estimators.

We experimented with various combinations of feature vectors (results of which are omitted, for brevity), in particular the bin size of histograms. We found this had a noticeable effect on classification accuracy - too large (i.e. few bins) or too small (i.e. many bins) would result in more misclassifications. For reference, table 2 shows the classification accuracy when using purely the raw frequency spectrum values as features. This works surprisingly well for the k-NN classifier but poorly for other algorithms - perhaps due to its high dimensionality. We suspect that with less clean data, using the raw frequency spectrum values as features would not work as well since the data would not necessarily be normalised to the same degree.

SVM	65.2
k-NN	98
Decision Trees	87.6
Random Forests	91
Adaboost	84

Table 2: Classification accuracy when using raw frequency spectrum as features.

Altering the SVM kernels showed us that RBF kernels gave us the best results. Linear and polynomial kernels took far too long to train. We also varied the hyper parameter C but the default value of 1 proved to work mostly well enough.

Varying the weights accorded to samples when classifying using our k-NN classifier had little effect on the overall accuracy - presumably because the values clustered together fairly closely.

Given the immediately obvious success of the random forests classifier, we collected results for varying numbers of estimators, shown in table **??**. 100 estimators gives us the best accuracy of **100%**!

6 Conclusion & Limitations

Ideally we would like to be able to perform the classification in real-time. While our results indicate that random forest have the best performance, the memory overhead required to store these trees may not be ideal for embedded computers. Thus, we may have

Number of Bins	0	10	50	100
SVM	65.2	70.4	81.8	96
k-NN	92	92	92	92
Decision Trees	92.8	95.6	94.4	90.8
Random Forests	95.2	98.8	**99.6**	97.4
Adaboost	89.6	94.4	96.2	93

Table 3: Overall results: average test set classification accuracy

n_estimators	Training	Test
1	95.4	88.6
2	94.4	90.6
5	99.6	96.8
10	100	99.2
20	100	99.4
50	100	99.6
100	100	100
200	100	99.2

Table 4: Number of estimators for random forest classifier: effect on average training and test set accuracy

to consider optimizing parametric classification methods that utilize less memory.

Additionally, we would like to extend this binary classification of the data to a multi-class problem. Instead of classifying the drone as functional or non-functional, it would be more ideal to classify what issue is causing the unmanned aerial vehicle to not function properly. This would require a larger set of data for each category to improve classification accuracy.

Here we have shown it is possible to very accurately classify the data collected through this setup to diagnose mechanical failures on an unmanned aerial system by using standard frequency spectrum features. These features and the algorithms used are very similar to other classification problems in the frequency domain - such as classification of music.

To ensure the robustness of this classification, more data needs to be collected whilst flying - in particular while multiple motors are running concurrently. These may introduce vibrations at various harmonic frequencies which may require altered feature vectors or better normalisation (or pre-processing) to maintain classification accuracy.

7 Acknowledgements

Thank you to Yong Keong Yap for providing us with the data used for this project and the diagram of the testing setup used in this paper.

References

[1] Zhouyu Fu et al. “A survey of audio-based music classification and annotation”. In: Multimedia, IEEE Transactions on 13.2 (2011), pp. 303–319.

[2] Adam Głowacz, Witold Głowacz, and Andrzej Głowacz. “Sound Recognition of Musical Instruments with Application of FFT and K-NN Classifier with Cosine Distance*”. In: AUTOMATYKA 14 (2010).

[3] Michael Haggblade, Yang Hong, and Kenny Kao. “Music genre classification”. In: Department of Computer Science, Stanford University (2011).

[4] Tomi Kinnunen. “Spectral features for automatic text-independent speaker recognition”. In: Licentiatesthesis, Department of computer science, University of (2003).