2014 IEEE International Conference on
Autonomous Robot Systems and Competitions (ICARSC)
May 14-15, Espinho, Portugal

Evaluation Of Signal Smoothing Algorithms for Stability of a Quadrotor MAV

Asheesh Ranjan, Pranav Jetley
Department of Computer Engineering
Delhi Technological University
Delhi, India

***Abstract*—The control of mobile robots involves the use of wireless signals. In this paper, we compare and evaluate the real time performance of multiple smoothing algorithms for a pulse width modulated digital signal on an Arduino Uno microcontroller board. The input signal is generated in a real world setting via a manually controlled RC transmitter-receiver setup. The signal smoothing is performed on the input signal and performance is evaluated in terms of standard deviation and signal to noise ratio. We also focus on the needs of a critical real-time system, a remotely piloted UAV, and perform a secondary analysis in terms of mean run time and signal accuracy to find the most suitable algorithm.**

I. Introduction

There has been a spurt of research and developmental activities in the field of unmanned aerial vehicles (UAVs) in the last decade. They are used to perform various types of missions ranging from fire fighting, surveillance and reconnaissance, locating survivors in natural disasters, to offensive military application. They offer the advantage of being un-piloted and can therefore be used for more dangerous missions, where the risks for piloted aircraft are high. Recent advancements in the field have focused on the development of smaller UAVs, which are called micro aerial vehicles or MAVs [1]. These are usually smaller in size which make them applicable for use in areas where larger UAVs cannot be used, such as being able to fly indoors. One type of MAV is a quadrotor, which contains four motors mounted at the ends of two beams, usually arranged in a plus or X configuration. The simple design and lack of any moving parts makes the quadrotor a popular choice for various missions.

For remote operation, MAVs are operated by a radio controlled transmitter receiver system. The control signals are usually transmitted across a fixed frequency range. As they travel through the channel, these signals are subject to interference by noise, which can corrupt the control signal and compromise the functionality of the MAV. A variation in the control signal to the MAV will cause abrupt changes in motor RPM, leading to varying thrust values. This variation in the thrust generated by the propellers can significantly change the orientation of the craft, in turn compromising flight stability. Thus, there is a need to filter out the noisy part of the signal to ensure that the MAV responds reliably to different control inputs.

This paper addresses the removal of noise in a control signal for a MAV using various time domain filters. Our test bed is a quadrotor MAV, which is controlled by an onboard Arduino Uno microcontroller board receiving control inputs from a 2.4 Ghz Avionic RCB7X transmitter receiver system. We will be implementing smoothing algorithms on the pulse width modulated (PWM) digital signals sent by the receiver to the Arduino Uno. The processed output will be sent to the four motors.

This paper is organized as follows; section II gives a background about different types of noise and their generation. It will also give a backdrop about different signal smoothing algorithms used for filtering noise from a signal, and the methods applied to compare the performance of different algorithms. Section III describes the hardware platform that has been used for conducting the experiment. Section IV describes the experiment in detail. The methods used for gathering data, factors behind the use of different algorithms, the statistical and application specific measures used to compare them are described. Section V focuses on the results and evaluation of our experiment. Finally, conclusion and future work are presented in section VI.

II. Background

A. Types of Noise and Noise Generation

The error introduced into a signal during transmission from the sender to the receiver is called noise. The presence of noise in a signal impairs the signal to noise ratio, which is a measure of signal quality, and causes fluctuation in the signal value. This in turn, makes the signal inaccurate and unreliable and may even lead to a loss of data.

There are various sources of noise in our experimental setup. Noise can either be generated at the transmitter side circuit, in the transmission channel, or at the receiver side circuit. The types of noise being generated can be categorized depending on the cause. Random fluctuations in the current flowing in the circuits of the transmitter and receiver may be a source of shot noise. Flicker noise may occur in electronic devices and can show up as irregularities in the conducting wires causing fluctuation in the voltage and current values. Another type of noise is transient noise, which occurs during signal transmission and is caused by interference in the transmission channel. These are short pulses followed by decaying low frequency oscillations. Another source of noise is the thermal agitation of electrons inside a conductor, called thermal noise. Lastly, a major source of noise in our experiment is quantization noise. The analog signal sent by the transmitter is converted to a (PWM) signal before being sent

978-1-4799-4254-1/14/$31.00 ©2014 IEEE

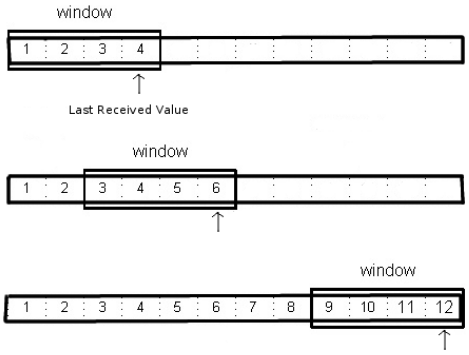

Fig. 1. Input channel with moving window

to the motors. The rounding off of errors that occur during this process of signal conversion is known as quantization error, which presents itself as noise in the original signal.

B. Signal Smoothing and Related Algorithms

Signal smoothing is a process that attempts to capture essential information while leaving out the noise in the signal, by interpolating the raw signal to estimate the original signal. As our system updates control signals in real time, we have considered filters in the time domain for signal smoothing [2] i.e. the simple moving average (SMA), cumulative moving average (CMA), exponential moving average (EMA), the Savitzky-Golay (SG) filter, and the Ramer-Douglas-Peucker (RDP) algorithm. These time domain filters usually have a finite window width and the window moves along the data set, as shown in the figure below. For our application, the most recently received signal value is the first value in the moving window. The data points in the smoothing window are assigned different weights and the actual value is interpolated. The weighting function is the primary factor differentiating the algorithms.

C. Comparison Methodology

The comparison methodology applied for the various algorithms is two fold: a statistical comparison and an application specific comparison. In the statistical comparison we compute the standard deviation (SD) of the data sets obtained for different algorithms. The SD is an indicator of the variation of a data set from its mean value. A lower SD in the control signal will result in more stable flight. We will also be comparing the ratios of the signal to noise ratio (SNR) of the smooth and noisy signal. The SNR ratio will provide us with a measure of the noise present in the signal.

For a signal consisting of a constant signal part a varying noisy part, the SNR is given by

$$SNR = \frac{S^2}{\sigma_n^2} \tag{1}$$

where S is the constant signal value and n^2 is the variance of the noise. Thus, the SNR ratio of the smooth and noise signal is given by

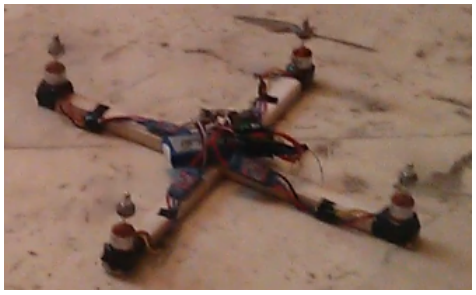

Fig. 2. Quadrotor MAV Test Platform

$$SNR\,Ratio = \frac{\frac{S^2}{\sigma_{n1}^2}}{\frac{S^2}{\sigma_{n2}^2}} \tag{2}$$

where σ_{n1}^2 is the variance of the noise of the original signal and σ_{n2}^2 is the variance of the noise of the noisy signal. As the input signal is the same in both the data sets, the useful portion of the signal denoted by S can be assumed to be equal. Thus, the above equation simplifies to

$$SNR\,Ratio = \frac{\sigma_{n2}^2}{\sigma_{n1}^2} \tag{3}$$

While the previous analyses provide an indication of the signal parameters in terms of statistics, these parameters alone do not provide sufficient information regarding real world performance of the algorithms. One of the constraints is that the processing power on the Arduino Uno is constrained [3], with an Atmega 328 running at clock frequency of 16 MHz. To run the motors stably, the Arduino Uno needs to update the control signal at a frequency of 80 Mhz. This means that the run time of each algorithm needs to be within reasonable limits. Therefore, we will compute the mean run time (MRT) for each algorithm and the ones having the least run times are the ones which are most suitable for a control system application.

As the smoothing algorithm attempts to interpolate the control signal from the received signal, we will also analyse the accuracy of the smoothed output by calculating the difference of the integrals (DOI) of the raw and smooth signals. This term will indicate the extent of signal deviation from the original signal after smoothing. In the real world, this translates to the system reacting as expected to the control signal.

III. HARDWARE PLATFORM

The MAV used in the experiment consists of two 45cm x 2.5cm beams arranged in a X configuration. The beams are made of balsa wood and have the electronics unit is housed at the center of the beams. Four 8x4 APC E propellers are mounted on Turnigy D2830-11 1000kV brushless outrunner motors, which are powered by a Turnigy 3000 mAh 3S1P lithium polymer battery and are

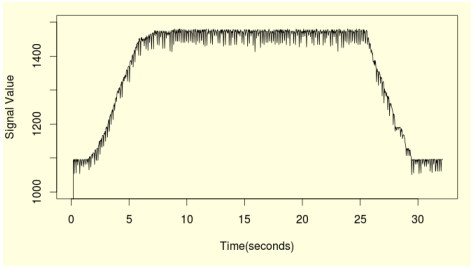

Fig. 3. Sample input *(motor activates at a signal value 1150)*

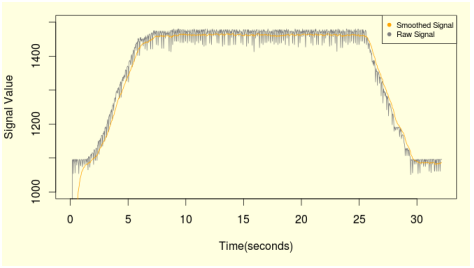

Fig. 4. Plot representing input signal and smoothed output for EMA 0.10

connected via a Hobbyking UBEC 30A electronic speed control. The electronic speed controllers are controlled by the Arduino microcontroller board. Our experiment makes use of the Avionic RCB7X 2.4Ghz 7-channel transmitter receiver system, which incorporates frequency hopping spread spectrum (FHSS) technology. FHSS transmits radio signals by switching between different frequency channels, using a particular sequence known to both the transmitter and receiver. This makes it highly resistant to narrow-band interference. The receiver converts the control signal into a digital PWM signal and sends it to the Arduino board.

The Arduino board being used here is an Arduino Uno Revision 3. This board forms the main controller of our quadrotor platform. The board is programmed using the official Arduino IDE, which comes packaged with platform specific libraries and configuration files. The Arduino Uno board has limited processing and memory resources. It consists of Atmega 328 microcontroller operating at 16 MHz, 1KB EEPROM, 2KB SRAM, and 32KB flash memory. It provides 14 digital and 6 analog general-purpose input-output pins, out of which 6 support PWM. We will be connecting the PWM pins to the receiver as the receiver generates PWM signals. It also has a Atmega 16U2 microcontroller which allows the board to communicate with a computer via a serial connection as well as allows the main microcontroller to be programmed.

IV. Experiment

We have considered a situation where the quadrotor starts from rest and accelerates to a throttle level of approximately 50%, which is sufficient for the quadrotor to hover. The quadrotor stays in the hover position for 15 seconds, and finally decelerates to idle throttle in the last 5 seconds. The throttle is controlled manually to closely resemble real world operation. The algorithms are implemented in C++. The Arduino board processes the digital PWM signals being sent by the receiver using the implemented smoothing algorithms. We monitor the system in real-time using a serial connection and record the input signal value, both raw and smoothed, and the times at which the smoothing function is called and returns. This process is repeated multiple times per algorithm to ensure a significant sized data set.

We have considered filters in the time domain for signal smoothing i.e. the simple moving average (SMA), cumulative moving average (CMA), exponential moving average (EMA), the Savitzky-Golay filter, and the Ramer-Douglas-Peucker (RDP) algorithm. The SMA is the mean of the set of data points distributed uniformly on either side of a central value. The computed mean is then set as the control value in place of the received value at that time instant. The number of data points, also known as the window size can be varied.

The CMA takes the mean of all the data that has arrived till that particular instant. It is given by the formula:

$$C_{k+1} = \frac{x_{k+1} + xC_k}{k+1} \tag{4}$$

$$C_k = \frac{\sum_{i=1}^{k} x_i}{k} \tag{5}$$

where C_k is the cumulative average of k data points and x_k is the value of k^{th} data point. C_0 is taken as zero. Similar to SMA, we again compute the CMA and set it as the control value, in place of the received value at that time instant.

The EMA is a type of weighted moving average with an exponential weighting function [4]. The weighting function uses a continuously decreasing exponential function. A constant factor, alpha, represents the degree of decrease in successive weights, and lies between 0 and 1. A smaller value of alpha takes more of the previous readings into account for calculating the current signal value. The EMA for a series S can be calculated as:

$$S_1 = C_1 \quad \text{for } t = 1$$
$$S_t = \alpha Y_t + (1-\alpha) S_{t-1} \quad \text{for } t > 1 \tag{6}$$

where Y_t is the data value at a time instant t. S_t is the value of the EMA at any time instant t.

The SG filter is a digital filter, which works on the method of fitting least squares, of a polynomial of a given order, to data points in a moving window. The polynomial is evaluated at the center of the moving window, and that value is the filtered value. The window is shifted over the entire data set and the central value of the window is calculated in each iteration of the algorithm. The degree of the polynomial used for least squares fitting and the moving window size can be varied and tested to obtain the most appropriate combination of parameters. The convolution coefficients for smoothing the signal can be obtained from the original paper of Savitzky and Golay [5].

The final algorithm implemented was RDP algorithm

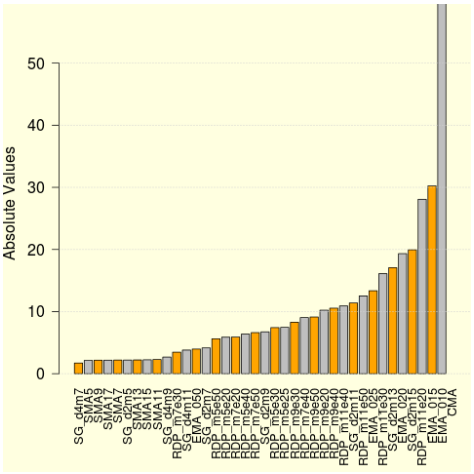

Fig. 5. SNR Ratios

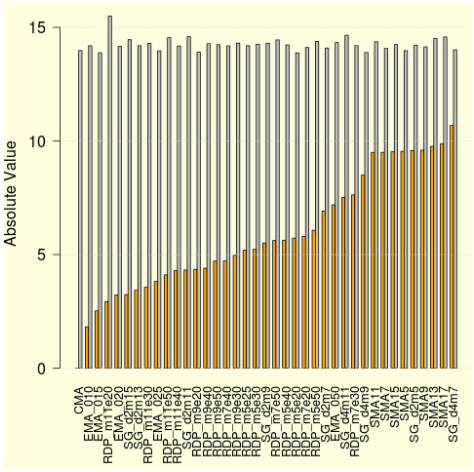

Fig. 6. Standard deviation for input and smoothed signal *(orange represents the smoothed signal and grey represents the original signal)*

[6]. Given a set of curves, it attempts to reduce the number of points required to best approximate the curve. The algorithm uses the perpendicular distance of points from the curve to estimate which points need to be ignored. The ones at a distance greater than a fixed value (ϵ) are considered significant and remain part of the final smoothed output. It can be applied for smoothing noisy signals by filtering the data set to include fewer points.

Now that the algorithms were implemented, we executed the program on the Arduino board and observed the performance in real time. An open serial connection between the Arduino board and the workstation was used to collect data for a period of 30 seconds using the input signal as described above. Each record of the data set collected consisted of four parameters: time of call of smoothing function, raw value of input signal, time of return of smoothing function, and the smoothed signal value. To compare the performance of the different algorithms, we first performed a statistical comparison. The firs part of this analysis involved computing the SD of the raw and smooth value of the data set. This was done using the SD function present in the stats package of R. Furthermore, the SNR ratio was also computed for the raw and smooth signal. The SNR values were computed using the formula given in section II. We used the variance function from the stats package in R for this analysis.

The next phase of analysis involved an application specific evaluation of the different algorithms on the quadrotor platform. For this, the MRT for each algorithm was computed. The run time was estimated by calculating the difference of the time of call of smoothing function and the time of return of smoothing function. The run time was then averaged over the entire data set. The second part of this analysis consisted of calculating the DOI of the smooth and noisy signal. To compute the integrals, we implemented an integral function.

V. Results and Evaluation

As mentioned above, the data was collected in real-time from the MAV using a serial connection. The Arduino was refreshing control signal to the motors at a variable frequency depending on the algorithm being used. The data obtained was then preprocessed to remove any instances of invalid characters. An example of the input signal versus the smoothed output of EMA for alpha 0.10 is shown in figure 8.

A. Statistical Evaluation

First, we calculated the SD of each data set in the time interval when the signal was expected to be stable i.e. from 10 to 25 seconds. It is expected that the SD of the smooth data set will be lower than that of the raw data set and that lower SDs will result in a more stable control signal. This will result in a uniform thrust being generated leading to a more stable flight.

The SDs showed a visibly improved output signal being generated in all cases. As can be seen in figure 6, certain algorithms performed markedly better than the rest, with CMA resulting in a SD of 0. EMA with alpha 0.10 and 0.15 performed well with low SDs. Thus, we observed that a higher dependence on the previous values leads to a smoother signal. The SG filter was most efficient for a quadratic fitting and a larger window size. For the quartic fitting, a larger window size gave a lower SD. All variations of window size and epsilon for RDP resulted in higher SDs than EMA, but were approximately close to each other. The worst performing algorithms were variations of SMA, resulting in consistently large SDs.

Next, we computed the ratio of SNRs for the smoothed and raw signal, which can be seen in figure 5. It is expected a higher SNR ratio of the control signal will indicate a signal with lower fraction of noise present in it. The variation in SNR was found to be significant ranging from 1.7 to 6. The CMA case in particular had a zero variance in the smoothed signal resulting in a extremely large SNR ratio. The highest SNR ratio was observed for EMA with alpha 0.10 followed by EMA with alpha 0.15. A large majority of variations of RDP were found to have SNR ratios distributed in a small range from 5.6 to 10.9. However, a window size of 11 and

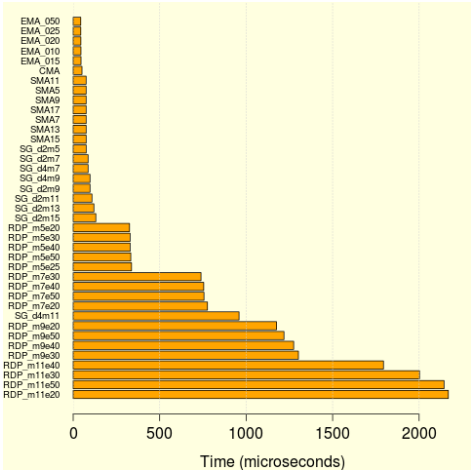

Fig. 7. Mean run time

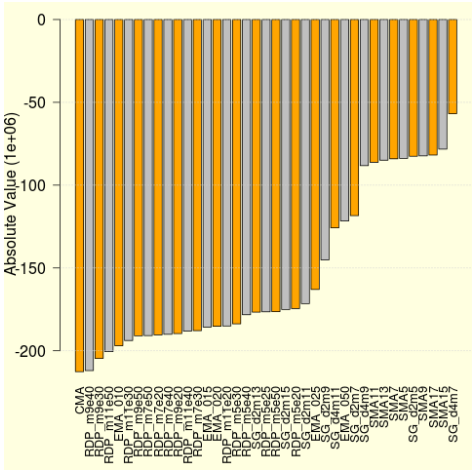

Fig. 8. Difference of integrals

an epsilon value of 20 gave the highest SNR ratio of 28. All variations of SMA resulted in very low SNR ratios. SG filter showed variable SNR ratios depending on the window size and degree of fitting. However, the highest SNR ratio was observed for a quadratic fitting and a window size of 15.

B. Application Oriented Evaluation

Our second phase of evaluation consisted of comparisons based on the requirements for our quadrotor MAV. The primary concerns as mentioned above were the accuracy of the signal being generated after smoothing as well as the time taken for the execution of the algorithm.

Our expectation from the DOI measure was that a value close to zero will indicate a more accurate output. The accuracy of the signal will translate to a more responsive MAV. On observing figure 8, we see that CMA has the lowest DOI, thus indicating that it was the least accurate. This makes it unfeasible for use in control systems. RDP for large window sizes also resulted in significantly low values. The EMA with alpha 0.10 performed poorly whereas EMA with alpha 0.15 performed reasonably well with a DOI of -185. Different variations of SG filter had a varying performance ranging from a DOI of -176 for quadratic fitting with window size 13 to -57 for a quartic fitting with window size 7. The SMA performed extremely well on this metric having the least DOI with no major difference across different window sizes.

Next, we compared the mean run time as the severely constrained requirement of the Arduino processor required smoothing to be performed efficiently. This will allow the Arduino Uno to update the control signal at a higher frequency. The mean run time for EMA was the lowest followed by CMA. SMA took approximately double the time required for EMA. SG filter performed reasonably well with MRT varying from 75 to 130 for the smallest degree and smallest window size. As the window size and the degree of fitting increased, there was an increase in the MRT. The worst performing algorithm was RDP as it involved multiple recursions, which made it unfeasible for use in our application.

Finally, we intended to combine the above four measures into an aggregate measure based on which we would estimate the best performing algorithm for our specific application. To do this, we normalized each data set on a zero-one scale with higher values indicating a better performance. A weighting function that would assign a priority to each measure was decided. The function assigned the following weights: 4-MRT, 3-DOI, 2-SNR ratio, and 1-SD. We assigned MRT the maximum priority as it would effect the responsiveness of the entire system to any input being provided. We calculated the aggregate measure or score by adding the four weighted values. The results of this analysis were as shown in table I.

TABLE I. FINAL SCORES OF ALGORITHMS

Rank	Algorithm	Score
1	EMA_010	9.707
2	EMA_015	8.371
3	EMA_020	7.916
4	SG_d2m15	7.577
5	SG_d2m13	7.509

VI. CONCLUSION AND FUTURE WORK

In this paper, multiple signal smoothing algorithms were implemented and tested on a quadrotor MAV platform. The control input was processed in real time using various signal processing algorithms and their performance evaluated. The evaluation was done on the basis of SD and SNR, followed by DOI and MRT. Exponential moving average filters were found to be the best performing algorithms with alpha parameters of 0.10 and 0.15. EMA was found to perform well on all metrics and came across as the best choice for application in a MAV.

In the future, we will explore other filters and experiment with different scenarios and larger data sets. Efficient signal smoothing on a low power platform such as Arduino Uno has the potential to be applied in many areas where control systems may be involved. Furthermore, these results are applicable to other areas utilising digital signal processing such as audio, SONAR and RADAR systems. Specifically, for MAVs, we

will incorporate an inertial measurement unit and evaluate the performance of the smoothing algorithms for varying flight patterns. Indoor flying is one of the core application areas of MAVs and stable flight with minimal vibrations is critical in closed and constrained environments.

REFERENCES

[1] C. Galiski and R. bikowskib *"Some problems of micro air vehicles development,"* Bulletin of the Polish Academy of Sciences, Technical Sciences, Vol. 55, No. 1, 2007

[2] Seng Hansun, *"A new approach of moving average method in time series analysis,"* New Media Studies (CoNMedia), 2013 Conference on , vol., no., pp.1,4, 27-28 Nov. 2013

[3] Koswatta, R.; Karmakar, N.C., *"Moving average filtering technique for signal processing in digital section of UWB chipless RFID reader,"* Microwave Conference Proceedings (APMC), 2010 Asia-Pacific , vol., no., pp.1304,1307, 7-10 Dec. 2010

[4] Wilson, A., *"Event Triggered Analog Data Acquisition Using the Exponential Moving Average,"* Sensors Journal, IEEE , vol.PP, no.99, pp.1,1

[5] Abraham. Savitzky and M. J. E. Golay; *Smoothing and Differentiation of Data by Simplified Least Squares Procedures.* Analytical Chemistry 1964 36 (8), 1627-1639

[6] Douglas, D. H. and Peucker, T. K. (2011) *Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or its Caricature*, in Classics in Cartography: Reflections on Influential Articles from Cartographica (ed M. Dodge), John Wiley & Sons, Ltd, Chichester, UK.

[7] Chen, H.C.; Chen, S.W., *"A moving average based filtering system with its application to real-time QRS detection,"* Computers in Cardiology, 2003 , vol., no., pp.585,588, 21-24 Sept. 2003

[8] Ziluan Liu; Yuehui Jin; Yidong Cui; Qiyao Wang, *"Design and implementation of a line simplification algorithm for network measurement system,"* Broadband Network and Multimedia Technology (IC-BNMT), 2011 4th IEEE International Conference on , vol., no., pp.412,416, 28-30 Oct. 2011