

# Data Analysis Tool

---

TOOL OVERVIEW

# Data Logging Format

Data is logged in a text file in the following format:

```
#Profile = ./Profiles/GAUIquadPID.profile
#UAV Constants      P      I      D      Trim
#Pitch              -1980      0      -409      0
#Roll               0      0      0      0
#Yaw                0      0      0      0
#Altitude           0      0      0      0
#Longitude          0      0      0      0
#Latitude           0      0      0      0
#Orientation:      OptiTrack-VRPN
#Position:         OptiTrack-VRPN
#Direction:        OptiTrack-VRPN
#Communication:    6-Channel Trainer
#Control:          None
```

Experiment configuration

Data headers

```
#Time  Lat Lon Alt Roll Pitch Yaw PIDpitchIn PIDpitchOut setThrottle matrixInPitch Motor_1 Motor_3
0      +0.1306 +0.3377 +1.4709 +0.28 +3.48 -0.03 0.00 2606.05 0.00 434.34 0 1300
7      +0.1306 +0.3377 +1.4709 +0.28 +3.48 -0.03 0.00 2606.05 0.00 434.34 0 1300
11     +0.1303 +0.3380 +1.4709 +0.30 +3.53 +0.00 0.00 155.69 0.00 25.95 0 325
```

Data

# Components

---

DataAnalysis	← script
params	← structure
expData	← structure
parse_log()	← function
plot_data()	← function
plot_separate()	← function
plot_multi()	← function
plot_sub()	← function

# How to Use the Tool

---

1. Open up the `DataAnalysis` script
2. Set the options for the analysis
3. Run the `DataAnalysis` script
4. If needed, use the plotting functions for extra plots

**IMPORTANT:** The data header of time values in the log file must be named "Time"

# DataAnalysis Script

---

This is the top level script that user will first interface with

## Layout

- Set analysis configuration options
- Parse logged data
- Plot data, if required

```
%% Initial options

params.file.name = 'sampleLogFile.txt'; % name of the logfile

params.time.scaler = 1000; % value by which time is scaled
params.time.units = 's'; % units of time after scaling

params.angle.units = 'rad' % specify the units in degrees

params.plotting.plot = 0; % switch to choose plot type
params.plotting.type = 0; % to choose the type of plot

params.plotting.colsToPlot = [6 9]; % column numbers to plot
% if empty & p...
params.plotting.multiColsToPlot = []; % column numbers to plot

save params params;

%% parsing the log file specified

expData = parse_log(InitParams.file.name); % parse the log file

save expData expData;

%% plotting routines

% this is where the initial plotting functions will be called
```

# Analysis Configuration Options

---

`params`

All the analysis configuration options set in the `DataAnalysis` script are stored in the `params` structure

## Parameters overview

- Log file name
- Plotting
  - Quantities to plot
  - Type of plotting
    - Separate plots for multiple quantities
    - Single plot with super-imposed quantities
    - Sub-plots of multiple quantities
- Units for common quantities

# Analysis Configuration Options

---

`params.file.name`

- Name of the file

`params.file.path`

- Path of the file with a trailing file name

If the user knows the file name and the log file is in the current directory

- the user can set the file name in the `fname` variable

else

- the user can leave the value of `fname` blank
- an explorer window will pop up to locate the file to be parsed

# Analysis Configuration Options

---

`params.time.units`

- Unit of time to be displayed while plotting

`params.time.scaler`

- Value by which the time vector in the log file needs to be multiplied by to obtain the time in the unit mentioned in `params.time.units`

`params.angle.units`

- Unit in which the angles have been recorded



# Analysis Configuration Options

---

`params.plotting.plot`

- Switch to generate plots

`params.plotting.separate`

- Switch to generate individual plots

`params.plotting.separateData`

- Cell array of headers to be plotted individually. If empty, all headers are plotted on separate plots.

`params.plotting.multi`

- Switch to generate a plot with multiple data in the single plot

`params.plotting.multiData`

- Cell array of headers to be plotted together on the single plot. If empty, the `params.plotting.separateData` vector is used.

`params.plotting.subPlots`

- Switch to generate 2x1 subplots

`params.plotting.subPlotsData`

- Cell array of headers be plotted in the sub plots. If empty, the `params.plotting.multiData` vector is used.

# parse\_log() Function

---

```
function [loggedData] = parse_log(filename, params)
```

filename

- Either the name of the log file OR the path to the log file

params

- The analysis configuration options

loggedData

- A structure that contains the data contained in the log file
- This is the `expData` specifically

This function parses the data stored in the log file and returns a structure containing the data

# expData Structure

---

This structure will store all of the logged data in the following format:

## expData

- <header-name>
  - name
  - data
  - units

*Example:*

## expData

- Pitch
  - name = "Pitch"
  - data = <array of data logged for Pitch>
  - units = "rad"

To access an attribute of any logged quantity, use the following:

`expData.<header-name>.<attribute>`

`expData.Pitch.data`

# plot\_data() Function

---

```
function plot_data(expData, plotParams)
```

expData

- The structure containing the data in the log file

plotParams

- The plotting options that were set in the DataAnalysis script
- This is `params.plotting` specifically

This function plots the data according to the parameters mentioned in `plotParams`.

This function will most likely be used in the `DataAnalysis` script only.

# plot\_separate() Function

---

```
function plot_separate(expData, varargin)
```

expData

- The structure containing the data in the log file

varargin

- The name of headers to be plotted on the same plot with optional **plot formatting** options

This function plots separate plots for the data headers mentioned.

**Example:** `plot_multi(expData, 'Pitch', 'go', 'Roll', 'Yaw', 'rs')`

# plot\_multi() Function

---

```
function plot_multi(expData, numOfHeaders, varargin)
```

expData

- The structure containing the data in the log file

numOfHeaders

- number of data sets to be plotted on the same plot

varargin

- The name of headers to be plotted on the same plot with optional **plot formatting** options

This function is used to plot multiple headers on the same plot.

Example: `plot_multi(expData, 3, 'Pitch', 'go', 'Roll', 'Yaw')`

# plot\_sub() Function

---

```
function plot_sub(expData, varargin)
```

expData

- The structure containing the data in the log file

varargin

- The data headers to be plotted in the sub plots along with optional **plot formatting** options

This function generates 2x1 sub plots of data headers specified

**Example:** `plot_sub(expData, 'Pitch', 'go', 'Roll')`

# Plot Formatting Options

---

These are character strings like those that are used with MATLAB's plot function.

The character string made from one element from any or all the following 3 columns:

b	blue	.	point	-	solid
g	green	o	circle	:	dotted
r	red	x	x-mark	-.	dashdot
c	cyan	+	plus	--	dashed
m	magenta	*	star	(none)	no line
y	yellow	s	square		
k	black	d	diamond		
w	white	v	triangle (down)		
		^	triangle (up)		
		<	triangle (left)		
		>	triangle (right)		
		p	pentagram		
		h	hexagram		

Examples: `'go'`, `'rs'`, `'--rs'`



# Examples

---

# Log File Used

---

Using sampleLogFile.txt

The log file contains the following headers:

```
#Time   Lat Lon Alt Roll   Pitch   Yaw PIDpitchIn  PIDpitchOut setThrottle matrixInPitch  Motor_1 Motor_3
```

# Example 1

---

# Analysis Configuration Options

```
%% Initial options

fname = ''; % set file name here
fpath = ''; % leave this blank
if(isempty(fname))
    [fname, fpath] = uigetfile('.txt','Select log file'); %
end
params.file.name = fname;
params.file.path = [fpath fname];

params.time.units = 's'; % unit to time to be
params.time.scaler = 1/1000; % value by which the

params.angle.units = 'deg'; % specify the unit

params.plotting.plot = 1; % switch to choose
params.plotting.separate = 1; % to generate separate
params.plotting.multi = 0; % to super impose multiple
params.plotting.subPlots = 0; % to generate sub plots
params.plotting.separateData = {'Pitch','Roll','Yaw'};

params.plotting.multiData = {}; % if empty but switch
params.plotting.subPlotsData = {}; % if empty but switch

save params params;
```

empty file name

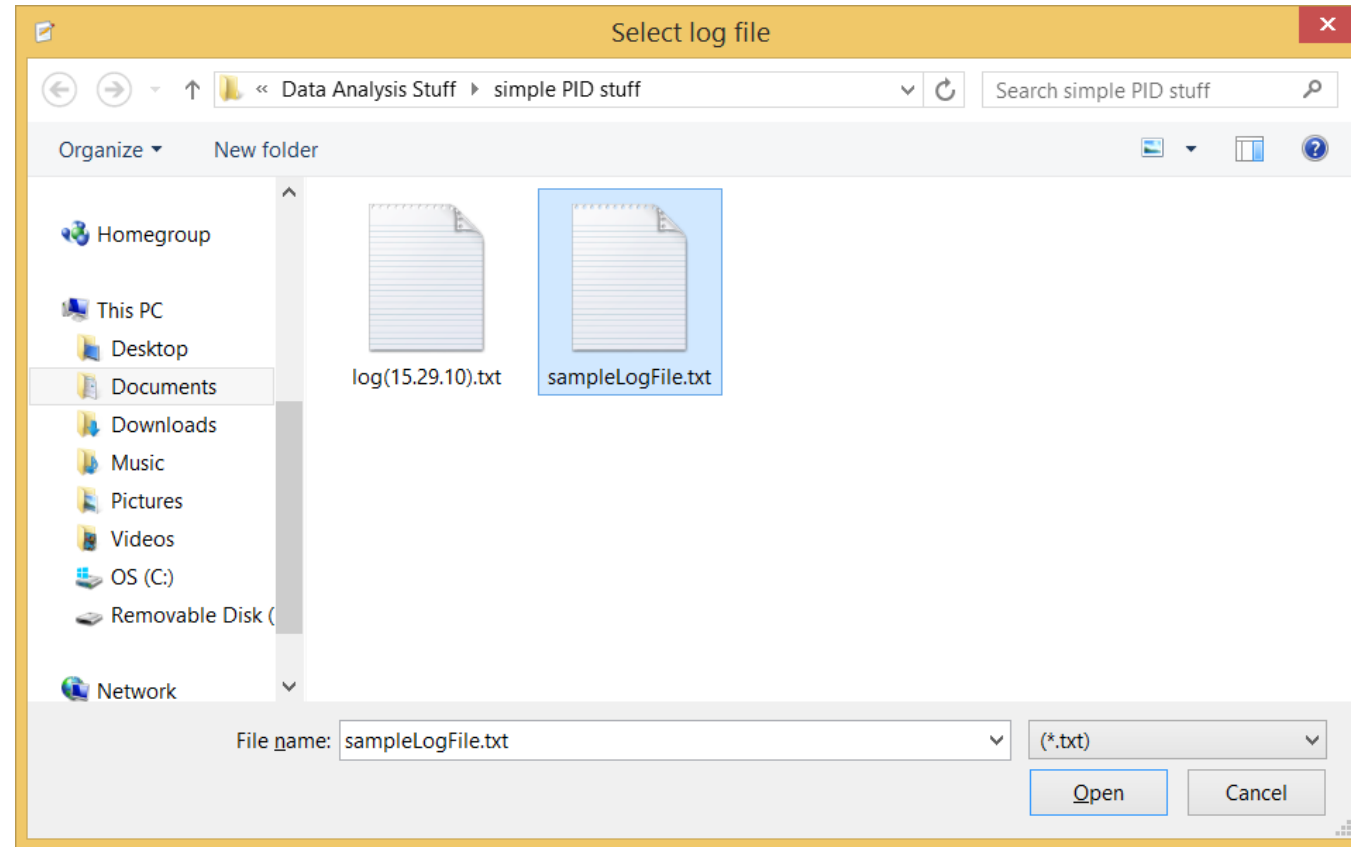
plotting switch set to ONE

plotting separate plots

data headers to plot on separate plots

After running this script →

# Select File



# Generated params Structure

---

```
params  
file  
time  
angle  
plotting
```

```
params.file  
name: 'simpleLogFile.txt'  
path: 'C:\Users\Rohit\Documents\...'
```

```
params.time  
units: 's'  
scaler: 1e-03
```

```
params.angle  
units: 'deg'
```

```
params.plotting  
plot: 1  
separate: 1  
multi: 0  
subPlots: 0  
separateData: {'Pitch' 'Roll' 'Yaw'}  
multiData: {}  
subPlotsData: {}
```

# Generated expData Structure

---

expData

Time

Lat

Lon

Alt

Roll

Pitch

Yaw

PIDpitchIn

PIDpitchOut

setThrottle

matrixInPitch

Motor\_1

Motor\_3

expData.Time

data: [733x1 double]

units: 's'

type: 'Time'

expData.Pitch

data: [733x1 double]

units: 'deg'

type: 'Angle'

expData.setThrottle

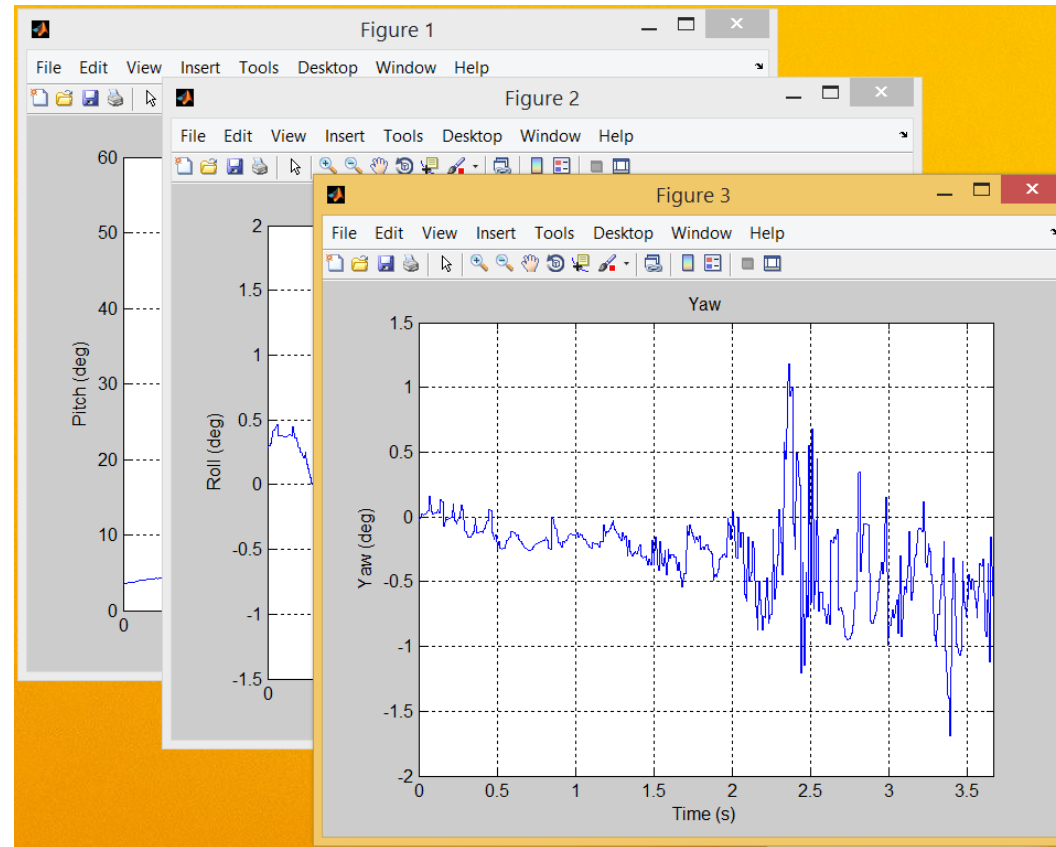
data: [733x1 double]

units: ''

type: 'Other'

# Generated Plots

---





# Example 2

---

# Analysis Configuration Options

```
%% Initial options

fname = 'sampleLogFile.txt';

fpath = '';
if(isempty(fname))
    [fname, fpath] = uigetfile('.txt','Select log file');
end
params.file.name = fname;
params.file.path = [fpath fname];

params.time.units = 's';
params.time.scaler = 1/1000;

params.angle.units = 'deg';

params.plotting.plot = 1;
params.plotting.separate = 1;
params.plotting.multi = 1;
params.plotting.subPlots = 0;
params.plotting.separateData = {'Pitch','Roll','Yaw'};

params.plotting.multiData = {};
params.plotting.subPlotsData = {};

save params params;
```

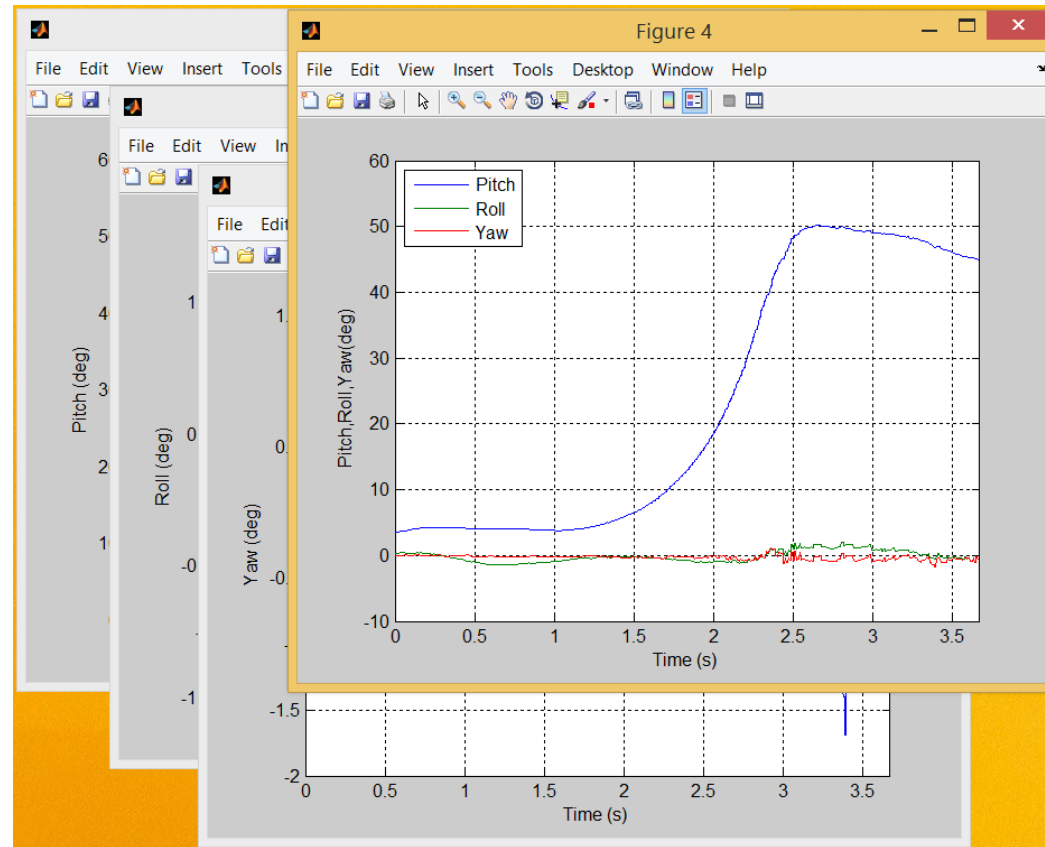
file name mentioned

plotting switch set to ONE

plotting separate plots and multi plots

data headers to plot on multi plots is empty  
So, the separateData vector will be used

# Plots Generated



# Example 3

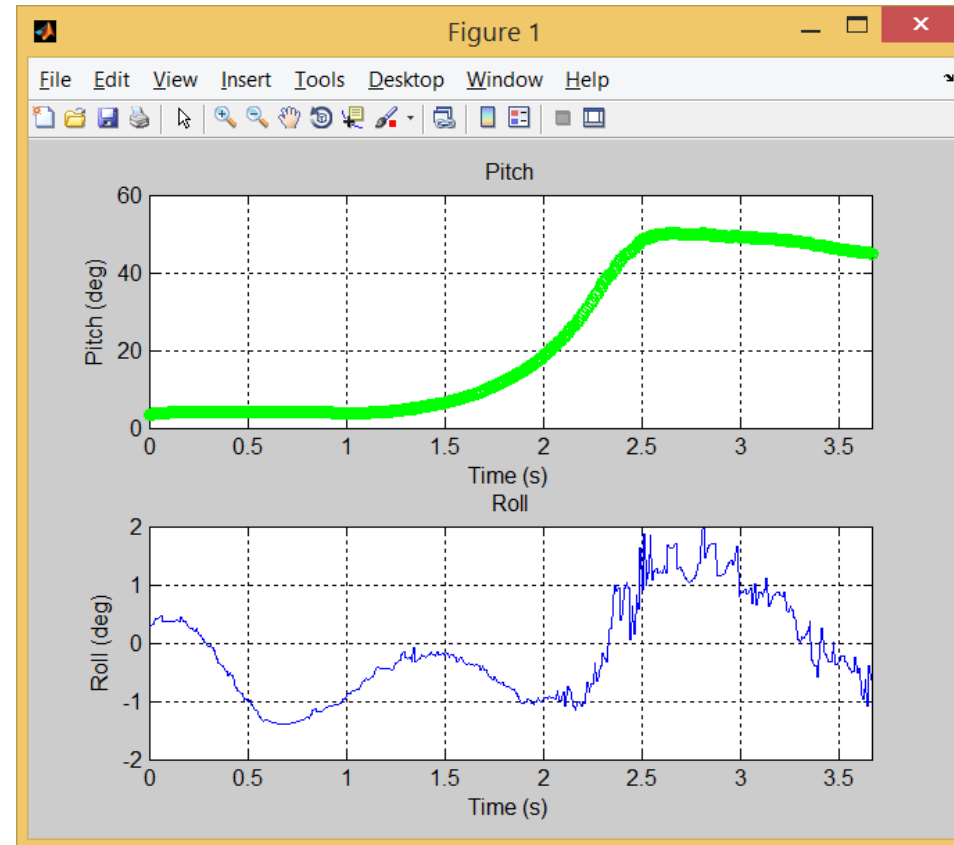
---

# Independently Using Plot Functions

---

```
plot_sub(expData, 'Pitch', 'go', 'Roll');
```

# Plot Generated



# Potential New Additions

---

Add the following functions to the tool

- `plot_separate_vectors()`
- `plot_multi_vectors()`
- `plot_sub_vectors()`

The current plotting functions allow only data from the data structure, `expData`, to be used. If any data needs to be modified (scaled, etc.), the original data in the data structure needs to be modified the plotting functions are to be used.

These functions will take in variable names instead of header names to allow for plotting of data that is not in the `expData` structure.