

# The alogview Tool for Analyzing Mission Log Files

June 2018

Michael Benjamin, mikerb@mit.edu  
Department of Mechanical Engineering, CSAIL  
MIT, Cambridge MA 02139

---

<b>1</b>	<b>Overview</b>	<b>1</b>
<b>2</b>	<b>The Five Primary Viewing Windows</b>	<b>3</b>
<b>3</b>	<b>The NavPlot Window and Controlling Replay</b>	<b>3</b>
3.1	Controlling the Current Time in Paused Mode . . . . .	3
3.2	Controlling the Current Time in Streaming Mode . . . . .	4
<b>4</b>	<b>Video Capture</b>	<b>4</b>
<b>5</b>	<b>The LogPlot Window For Viewing Time-Series Data</b>	<b>4</b>
5.1	Additional Buried but Useful Hot-Keys in the LogPlot Window . . . . .	5
<b>6</b>	<b>The IPFPlot Window For Viewing Helm Objective Functions</b>	<b>6</b>
6.1	Launching an IPFPlot Window . . . . .	6
6.2	Stepping Through Time and Replay Control in the IPFPlot Window . . . . .	6
6.3	Selecting the IvP Function in the IPFPlot Window . . . . .	7
6.4	Rendering the Collective IvP Function in the IPFPlot Window . . . . .	7
6.5	Variable Scoping Within the IPFPlot Window . . . . .	8
6.6	Additional Buried but Useful Hot-Keys in the IPFPlot Window . . . . .	8
<b>7</b>	<b>The HelmPlot Window For Viewing Helm State</b>	<b>9</b>
7.1	Launching the HelmPlot Window . . . . .	9
7.2	Stepping Through Time and Replay Control in the HelmPlot Window . . . . .	9
7.3	Behavior States in the Helm Plot Window . . . . .	9
7.4	Behavior Warnings, Errors, Mode History and Life Events . . . . .	10
7.5	Additional Buried but Useful Hot-Keys in the HelmPlot Window . . . . .	11
<b>8</b>	<b>The VarPlot Window For Viewing Variable Histories</b>	<b>11</b>
8.1	Launching the VarPlot Window . . . . .	11
8.2	Viewing the Variable History in the VarPlot Window . . . . .	12
8.3	Adding and Removing Variables from the History List . . . . .	13
8.4	Additional Buried but Useful Hot-Keys in the VarPlot Window . . . . .	13
<b>9</b>	<b>Command Line Usage for the alogview Tool</b>	<b>14</b>

---

## 1 Overview

The **alogview** application is a post-mission analysis utility loaded with one or more alog files, typically one per vehicle. It is more than a log file playback tool. A full state is maintained across all vehicles for both playing back sequentially and rewinding or jumping to any arbitrary point in

time. The main window may look a bit like the run-time **pMarineViewer** window, but many of the GUI components in **alogview** are available solely for post-mission analysis. A snapshot of the tool, with no pop-up sub-windows open, is shown in Figure 1.

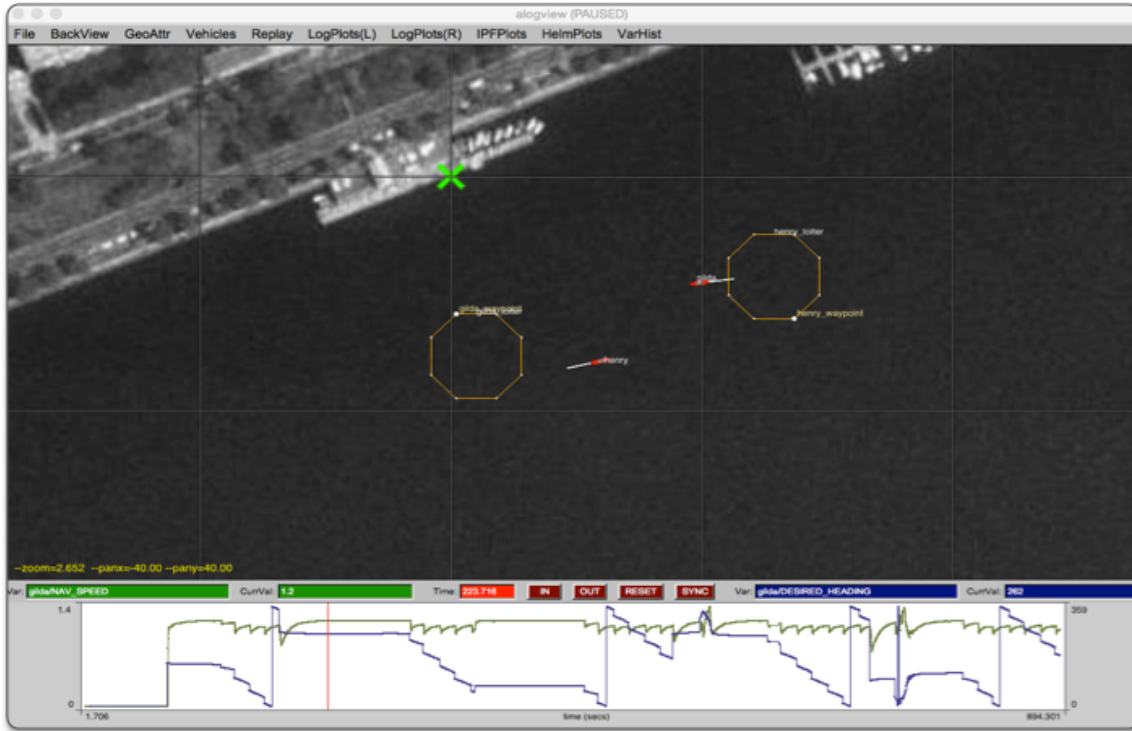


Figure 1: **The alogview tool**: used for post-mission rendering of alog files from one or more vehicles and stepping through time to analyze helm status, IvP objective functions, and other logged numerical data correlated to vehicle position in the op-area and time.

Quick Start: Launch **alogview** with one or more alog files on the command line, for example the two vehicle alog files from the **m2\_berta** example mission:

```
$ alogview henry.alog gilda.alog
```

Upon launch, the viewer should look similar to Figure 1. Step forward or backward through time with the '[' or ']' keys, or click anywhere in the time-series in the bottom part of the screen. Replay can also be automated by accelerating or decelerating replay with the 'a' or 'z' characters. Many other replay and visualization options are available via the pull-down menus and are introduced in later sections.

## Recent History of Development with Releases

- Pre-2011: This tool existed as **logview**, was not documented and had limited support for rendering objective functions and helm information. Mostly used in-house by developers.

- 2011: Better support for IvP function rendering. Initial thin documentation. Support added for geometric log entries. Still very buggy but useful enough to share with other adventurous users. Biggest limitations: (a) initial full-alog load into memory too huge for anything but small missions. (b) no general scoping tool available for non-numerical data, (c) many little bugs or incomplete features.
- 2015: First real attempt to support users generally outside core developers. Better memory management for large and/or multi-vehicle missions. New scope for non-numerical data. Pop-up implementation of sub-windows. Unlimited sub-windows supported for multi-vehicle missions. Overall much less buggy and more feature-rich.

## 2 The Five Primary Viewing Windows

The `alogview` tool can be described by its five GUI components, listed below. The first two are always open upon startup, and the remaining three are upon user request via the pull-down menus:

- The *NavPlot* window: This is the main vehicle viewing area shown at the top in Figure 1. It primarily shows the current vehicle position(s) and perhaps recent trajectory and other geometric artifacts produced by the vehicle apps or behaviors, such as waypoints or sensor artifacts. More detail in Section 3.
- The *LogPlot* window: This is the time-series window shown at the bottom in Figure 1. The variable's rendered are selected by the user from the list any logged numerical data from any vehicle. More detail in Section 5.
- The *IPFPlot* window: (pronounced "i-p-f-plot") This is a pop-up window launched from the `IPFPlots` pull-down menu, for a chosen vehicle. It will render the prevailing IvP objective functions responsible for latest helm decision. More detail in Section 6.
- The *HelmPlot* window: This is a pop-up window launched from the `HelmPlots` pull-down menu, for a chosen vehicle. It shows the helm state in terms of active, running, idle behaviors, as well as behavior warnings, errors and life events. More detail in Section 7.
- The *VarPlot* window: This is a pop-up window launched from the `VarPlots` pull-down menu, for a chosen vehicle, and chosen logged MOOS variable (string or double). It shows an ordered list of recent and pending postings for the given variable. Additional variables can be added resulting in an interleaved sequence. More detail in Section 8.

In all windows, the current time can be altered with the same keyboard hot keys, found in the `Replay` pull-down menu and described in Section 3.

## 3 The NavPlot Window and Controlling Replay

The *NavPlot* window is the main viewing space, shown upon startup, with the current and recent vehicle positions rendered for each vehicle alog file. It is the top part shown in Figure 1.

### 3.1 Controlling the Current Time in Paused Mode

By default `alogview` is paused upon startup. The options for stepping forward and backward through time may be seen from the `Replay` pull-down menu. There are three pairs of options:

- The '[' and ']' keys for stepping forward and backward by 1 second.
- The '' and '' keys for stepping forward and backward by 5 seconds.
- The 'ctrl-[' and 'ctrl-]' keys for stepping forward and backward by 0.1 seconds.

### 3.2 Controlling the Current Time in Streaming Mode

Replay may also be put into a streaming mode at any time by accelerating or decelerating the streaming speed. Streaming is controlled by the following three keys:

- The 'a' key will accelerate the time warp, doubling the warp upon each successive hit, up to 64Hz.
- The 'z' key will decelerate the time warp, halving the warp upon each successive hit, down to 1/8Hz.
- The 'SPACEBAR' key will pause or unpause the streaming mode at any time.

The time warp is always shown in the very top of the `alogview` window. When streaming at say 16Hz, you should see (Time Warp = 16) in the window title bar. When not streaming, i.e., paused, this is indicated instead of the time warp, as in Figure 1.

Note that in streaming, between each update, there is work going on under the hood in addition to the rendering tasks. This includes the update of the current state of what is being rendered in the main and pop-up windows. Often it is not possible to refresh at the time warp requested. In this situation, `alogview`, and your machine, will do the best it can to keep up. Instead of seeing (Time Warp = 16) in the title bar, you may see something like (Time Warp = 16)(Actual:8.73). The latter number will vary, but represents the best warping rate possible given what is being updated between refreshes.

A faster observed time warp can be achieved by altering the *streaming redraw interval*. By default this interval is 0.5 seconds. So, for example, if the replay time warp is 4, then 8 re-draws will need to be handled per second. This may or may not be achievable. If the redraw interval is instead 1.0 seconds, then half as many re-draws are required for the same effective time warp. The drawback will be that the motion of the vehicle and objects will appear more "jumpy". The *streaming redraw interval* may be altered in the `Replay` pull-down menu.

## 4 Video Capture

Video capture is no longer natively supported in `alogview` due to the common availability of third party video capturing software. Just launch `alogview`, put it into streaming mode (Section 3.2) at the desired rate, and launch your favorite video capture program.

## 5 The LogPlot Window For Viewing Time-Series Data

The *LogPlot* window allows the rendering of any numerical data from any loaded `alog` file to be plotted against time, as shown in Figure 1 and up close in Figure 2 below. The user may zoom in

to see finer resolution in the postings, by clicking either the **IN** or **OUT** buttons, or by the 'i' or 'o' keys while the mouse is in the *LogPlot* window. To return to the original full apperture zoom, hit the **RESET** button. The user may left-click at any point in the time series to adjust the current time (shown by the red bar in the viewer). This time adjustment is propagated not only to the *NavPlot* viewer but to all open pop-up windows.

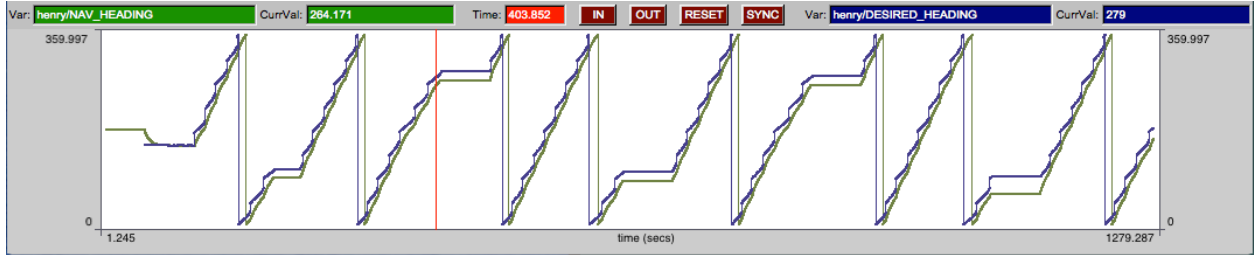


Figure 2: **The logplot window:** is used for viewing any numerical data found in any of the alog files loaded into *alogview*. Here the desired and actual vehicle headings are logged side-by-side.

Two variables, and two y-axes are available for selection. By default they are drawn on independent scales to allow variables with starkly different ranges, such as speed and heading, to be meaningfully rendered. The user may choose to synchronize the two variable ranges rendered to allow for more meaningful comparisons when the two variables have the same range, such as desired and observed speed. The synchronization is done by hitting the **SYNC** button at the top of the logplot window. The user chooses which two variables are rendered from the pull-down menu as shown in Figure 3.

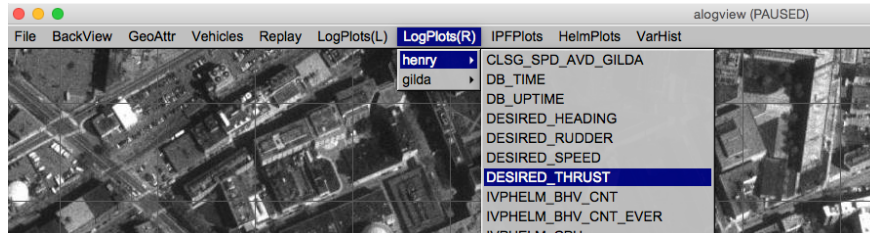


Figure 3: **The logplot pull-down menu:** is used for first selecting the vehicle and then the variable to be plotted.

If you find yourself wanting to repeatedly open an alog file with same one or two variables chosen for the *LogPlot* window, these variables can be chosen and at launch time passed as command line arguments:

```
$ alogview --lp=henry:NAV_SPEED --rp=gilda:NAV_SPEED
```

## 5.1 Additional Buried but Useful Hot-Keys in the LogPlot Window

There are a few hot-key functions available (only when the main window is active and the mouse is over the *LogPlot* window, that may not be apparent from simply interacting with the window:

- The 'i' key: will zoom in the resolution on the plot. This can also be done by hitting the IN button.
- The 'o' key: will zoom out the resolution on the plot. This can also be done by hitting the OUT button.
- The 'LEFT' and 'RIGHT' keys: will step forward and backward through time by 1 second.
- The 'UP' and 'DOWN' keys: will zoom in and out on the plot, essentially the same as the 'i' and 'o' keys.

The above hot-keys are in addition to the time control hot-keys listed in Section `sec_alv_time_replay` and Section 3.1.

## 6 The IPFPlot Window For Viewing Helm Objective Functions

The *IPFPlot* window allows the rendering of an IvP function (IPF) from any vehicle and any behavior found in any of the alog files loaded into *alogview*.

### 6.1 Launching an IPFPlot Window

An *IPFPlot* pop-up window is launched from the *alogview* main pull-down window as shown in Figure 4, by choosing a vehicle name. Separate windows can be opened for different vehicles, or even for the same vehicle.

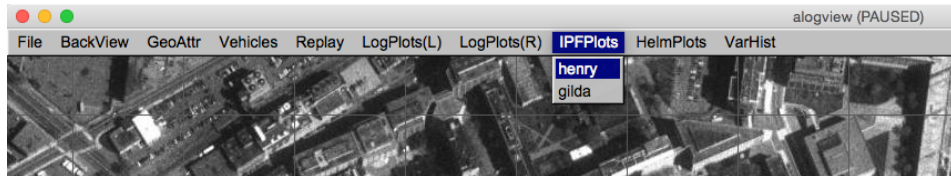


Figure 4: The *ipfplot* pull-down menu: is used for first selecting the vehicle and then the variable to be plotted.

A newly launched pop-up window loads the relevant information from disk into the *alogview* memory space. This includes all IvP functions for all behaviors at all times, for the given vehicle. This can be large. For some (larger) missions and some computer configurations, the larger memory footprint will begin to be noticed in terms of application responsiveness. Likewise, the internal data structure updates, each time the current time is changed, increases for each pop-up window. This may result in a slower streaming rate when in replay. Closing the pop-up window releases the memory and should immediately boost responsiveness and playback rate.

### 6.2 Stepping Through Time and Replay Control in the IPFPlot Window

The current time is shown in the upper left corner and is synchronized with the main viewing window and any other pop-up windows that may be open. The same time keys are functional when this window is active, e.g., '[', ']', ',', '>', '<', for stepping forward and backward in time. The replay control keys, e.g., 'a', 'z', are also functional, and the replay state and rate is also shown in the title bar of this window.

### 6.3 Selecting the IvP Function in the IPFPlot Window

The user may choose any single objective function at a time from the menu on the left side of the window shown in Figure 5. The current behavior weight is shown alongside the behavior name in the menu. If a behavior presently has a weight of zero, no function will be rendered when that behavior is selected. Multiple objective functions may be viewed simultaneously by launching more pop-up windows.

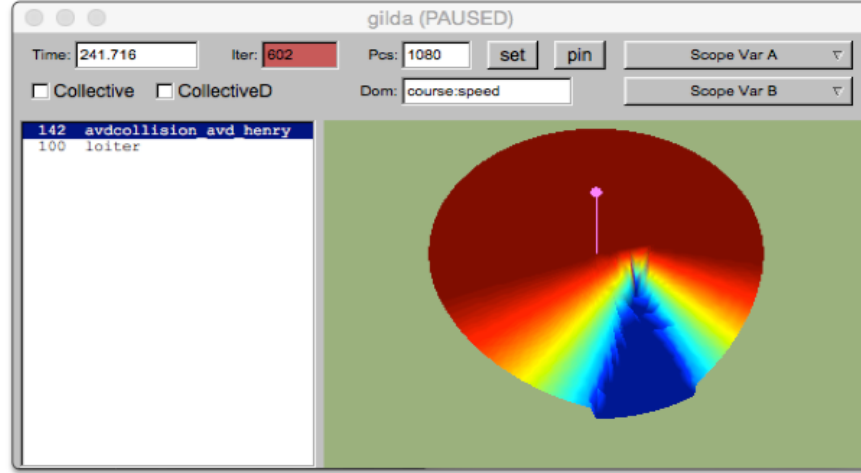


Figure 5: **The ipfplot window:** used for examining IvP functions for the chosen vehicle at the current time.

Since each window is associated with a particular vehicle, the helm iteration may be shown unambiguously in the redish window in the upper left. The rendering of the IvP function may be rotated or tilted for different perspectives by using the arrow keys. The perspective may be returned to the bird's eye view by hitting the SET button.

### 6.4 Rendering the Collective IvP Function in the IPFPlot Window

IvP functions of two forms are renderable - those defined over course and speed, and those defined over depth. In both cases, a rendering of the collective function is possible by clicking on the **Collective** or **CollectiveD** check-boxes on the left. In Figure 6 below, the collective objective function is shown, showing the weighted sum of the two behaviors. The purplish pin rendered in Figures 5 and 6 shows the actual chosen decision for the present helm iteration. When rendering the collective function as in Figure 6 the pin should be visually consistent with the peak of the function. When rendering a single objective function, as in Figure 5, the pin may not visually correlate to the peak of the function since other functions may be in play. For example in Figure 5 the pin identifies one of many points on the red plateau. The pin can be toggled on and off by hitting the PIN button.



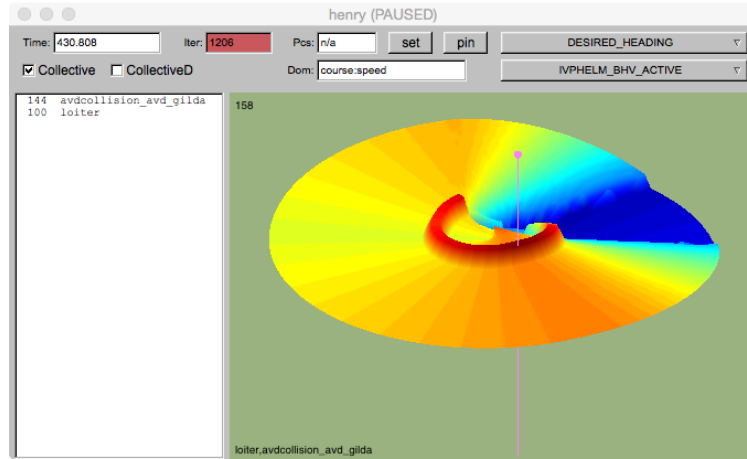


Figure 6: **The ipfplot window:** showing here the collective objective function representing the weighted sum of the two active behaviors.

## 6.5 Variable Scoping Within the IPFPlot Window

Often it is useful to scope on a couple key variables related to the rendered objective function, to observe how they change and perhaps influence the function as time evolves. Although a general scoping ability exists via the *VarPlot* window described below in Section 8, it can be handy to see the variable value(s) right in the *IPFPlot* window. The two pull-down menu buttons in the upper right in Figure 6 allow the selection of two variables whose values are output in the upper and lower left hand corners of the window as shown. The choice of these two variables is specific to the chosen behavior, or collective function. A different set of variables may be associated with each behavior, and they will be refreshed accordingly as the user switches between behaviors.

## 6.6 Additional Buried but Useful Hot-Keys in the IPFPlot Window

There are a few hot-key functions available that may not be apparent from simply interacting with the window:

- The '+' key: will increase the font size in the browser pane, and the font size of any scoped variable.
- The '-' key: will decrease the font size in the browser panes, and the font size of any scoped variable.
- The 'c' key: will toggle the IvP function rendering from an individual selected function to the collective objective function.
- The 'f' key: will put the window in a "full-window" mode showing only the rendered objective function, without the browser pane.
- The 'UP' and 'DOWN' keys: will tilt the rendered objective functions.
- The 'LEFT' and 'RIGHT' keys: will rotate the rendered objective functions.

The above hot-keys are in addition to the time control hot-keys listed in Section `sec.alv_time_replay`



and Section 3.1.

## 7 The HelmPlot Window For Viewing Helm State

The *HelmPlot* window renders information regarding the present helm state, including which behaviors are in which state and for how long, summary of update attempts for each behavior, behavior warnings, behavior errors, a history of helm mode changes, and a history of helm life events.

### 7.1 Launching the HelmPlot Window

The *HelmPlot* pop-up window is launched from the *alogview* main pull-down window as shown in Figure 7, by choosing a vehicle name. Separate windows can be opened for different vehicles, or even for the same vehicle.



Figure 7: **The helmplot pull-down menu:** is used for first selecting the vehicle and launching the *HelmPlot* window.

A newly launched pop-up window loads the relevant information from disk into the *alogview* memory space. For some (larger) missions and some computer configurations, the larger memory footprint will begin to be noticed in terms of application responsiveness. Likewise, the internal data structure updates, each time the current time is changed, increases for each pop-up window. This may result in a slower streaming rate when in replay.

### 7.2 Stepping Through Time and Replay Control in the HelmPlot Window

The current time is shown in the upper left corner and is synchronized with the main viewing window and any other pop-up windows that may be open. The same time keys are functional when this window is active, e.g., '[' , ']', ',', '>', '<', for stepping forward and backward in time. The replay control keys, e.g., 'a', 'z', are also functional, and the replay state and rate is also shown in the title bar of this window.

### 7.3 Behavior States in the Helm Plot Window

Each behavior known to the helm up to the present time is grouped into a list of either active, running, idle or completed modes as shown in Figure 8.

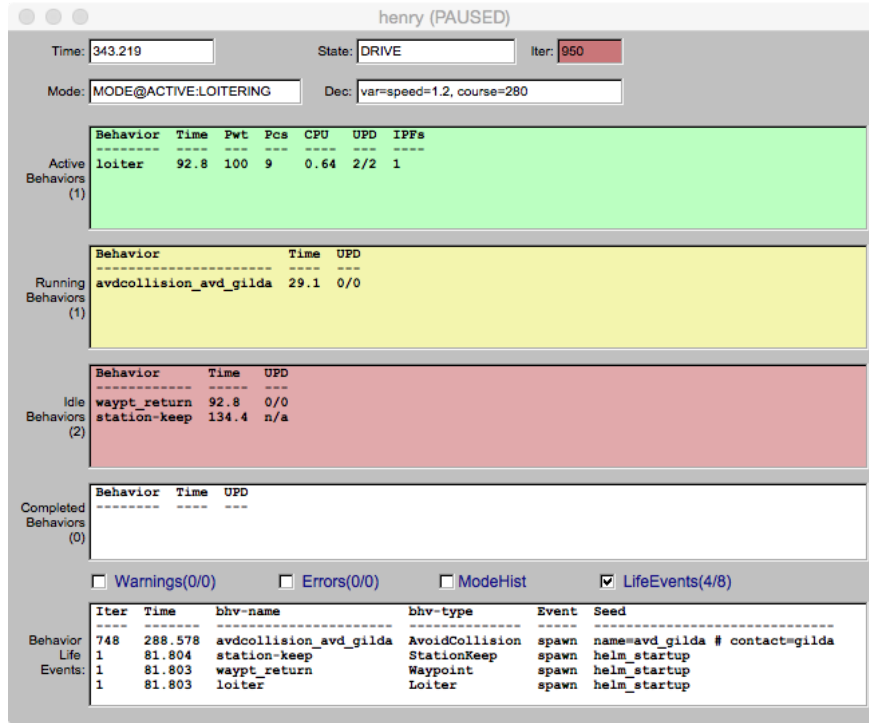


Figure 8: **The helmplot window:** used for analyzing the helm state and behavior modes.

For each behavior, the behavior name is listed, along with the elapsed time since the behavior has been in that mode. If the behavior is configured with the **updates** parameter, then the number of successful vs attempted updates are also listed. If the behavior does not have the **updates** parameter set, then **n/a** will instead be listed in that column as with the station-keep behavior in the figure below. For active, behaviors, the present priority weight and number of pieces is also shown.

## 7.4 Behavior Warnings, Errors, Mode History and Life Events

The bottom pane of the *HelmPlot* window may have one of four content modes, selectable with the check-box shown.

- *Behavior Warnings:* In this mode, all behavior warnings, in the form of messages written to **BHV\_WARNING**, will be listed in chronological order with the most recent at the top. One handy feature is that the existence of possible future warnings can be see in the check-box label. In parentheses, (a/b) indicates that *a* warnings are shown in the list out of *b* warnings total for the mission.
- *Behavior Errors:* In this mode, all behavior errors, in the form of messages written to **BHV\_ERROR**, will be listed in chronological order with the most recent at the top. In the check-box label, in parentheses, (a/b) indicates that *a* errors are shown in the list out of *b* errors total for the mission.
- *Mode History:* In this viewing mode, all helm mode changes will be listed in chronological order with the most recent at the top.

- *Life Events*: In this viewing mode, all helm life events (behavior spawning and deaths) will be listed in chronological order with the most recent at the top.

## 7.5 Additional Buried but Useful Hot-Keys in the HelmPlot Window

There are a few hot-key functions available that may not be apparent from simply interacting with the window:

- The '+' key: will increase the font size in the browser pane, and the font size of any scoped variable.
- The '-' key: will decrease the font size in the browser panes, and the font size of any scoped variable.
- The 'UP' and 'DOWN' keys: will tilt the rendered objective functions.
- The 'LEFT' and 'RIGHT' keys: will rotate the rendered objective functions.

The above hot-keys are in addition to the time control hot-keys listed in Section `sec.alv_time_replay` and Section 3.1.

## 8 The VarPlot Window For Viewing Variable Histories

The *VarPlot* window renders the full history of one or more chosen MOOS variables, allowing the user to see the present value, recent postings, and upcoming pending postings. This holds for both numerical and string variables.

### 8.1 Launching the VarPlot Window

The *VarPlot* pop-up window is launched from the *alogview* main pull-down window as shown in Figure 9, by choosing a vehicle name, and MOOS variable name. Separate windows can be opened for different vehicles, variables, or even for the same vehicle and variable. Although one variable is chosen upon launch, any number of variables can be inserted later from the VarPlot pull-down buttons



Figure 9: **The VarPlot pull-down menu:** is used for first selecting a vehicle and variable name, and launching the *VarPlot* window.

A newly launched pop-up window loads the variable history information from disk into the *alogview* memory space. For some (larger) missions and some computer configurations, the larger memory footprint will begin to be noticed in terms of application responsiveness. Likewise, the internal

data structure updates, conducted each time the current time is changed, increases for each pop-up window. This may result in a slower streaming rate when in replay. Closing the pop-up window releases the memory and should immediately boost responsiveness and playback rate.

## 8.2 Viewing the Variable History in the VarPlot Window

The *VarPlot* window contains two primary panes. The top pane shows everything in the past, up to the most recent post. The bottom pane shows everything in the future, starting with the next pending post. The "current time" can be regarded as being in between the two panes.

Time	Node	VarName	Source		
218.960	gilda	DESIRED_SPEED	pHelmIvP	1.2	
218.964	henry	DESIRED_HEADING	pHelmIvP	104	
218.998	henry	NAV_SPEED	uSimMarine	1.2	
219.109	henry	NAV_SPEED	uSimMarine	1.2	
219.212	henry	NAV_SPEED	uSimMarine	1.2	
219.236	gilda	DESIRED_SPEED	pHelmIvP	1.2	
219.240	henry	DESIRED_HEADING	pHelmIvP	104	
219.338	henry	NAV_SPEED	uSimMarine	1.2	
219.445	henry	NAV_SPEED	uSimMarine	1.2	
219.525	gilda	DESIRED_SPEED	pHelmIvP	1.2	
219.535	henry	DESIRED_HEADING	pHelmIvP	104	
219.548	henry	NAV_SPEED	uSimMarine	1.2	
219.680	henry	NAV_SPEED	uSimMarine	1.2	
219.804	henry	NAV_SPEED	uSimMarine	1.2	
219.834	gilda	DESIRED_SPEED	pHelmIvP	1.2	
219.835	henry	DESIRED_HEADING	pHelmIvP	104	
219.911	henry	NAV_SPEED	uSimMarine	1.2	
220.020	henry	NAV_SPEED	uSimMarine	1.2	
220.128	gilda	DESIRED_SPEED	pHelmIvP	1.2	
220.133	henry	DESIRED_HEADING	pHelmIvP	104	
220.135	henry	NAV_SPEED	uSimMarine	1.2	
220.262	henry	NAV_SPEED	uSimMarine	1.2	
220.364	henry	NAV_SPEED	uSimMarine	1.2	
220.379	gilda	DESIRED_SPEED	pHelmIvP	1.2	
220.384	henry	DESIRED_HEADING	pHelmIvP	104	
220.470	henry	NAV_SPEED	uSimMarine	1.2	
220.577	henry	NAV_SPEED	uSimMarine	1.2	
220.633	gilda	DESIRED_SPEED	pHelmIvP	1.2	
220.637	henry	DESIRED_HEADING	pHelmIvP	104	
220.686	henry	NAV_SPEED	uSimMarine	1.2	
220.789	henry	NAV_SPEED	uSimMarine	1.2	
220.891	henry	NAV_SPEED	uSimMarine	1.2	
220.918	gilda	DESIRED_SPEED	pHelmIvP	1.2	
220.922	henry	DESIRED_HEADING	pHelmIvP	104	
221.016	henry	NAV_SPEED	uSimMarine	1.2	
221.117	henry	NAV_SPEED	uSimMarine	1.2	
221.211	gilda	DESIRED_SPEED	pHelmIvP	1.2	
221.221	henry	NAV_SPEED	uSimMarine	1.2	
221.222	henry	DESIRED_HEADING	pHelmIvP	104	
221.350	henry	NAV_SPEED	uSimMarine	1.2	
221.451	henry	NAV_SPEED	uSimMarine	1.2	
221.472	gilda	DESIRED_SPEED	pHelmIvP	1.2	

Figure 10: **The VarPlot window**: used for scoping on a set of MOOS variables into the past and upcoming in the future.

Each variable posting has six components, separated into six columns in both panes:

- *Time*: The timestamp of the posting. This is always shown in the first column.
- *Node*: The node, i.e., vehicle name is shown in the second column. Since the scope may contain values across nodes, this can be useful to see. If only scoping on one variable, or if all variables are from the same node, then this column can be hidden by toggling the *Node* check-box.
- *Variable Name*: If scoping on only one variable, this third column may be hidden to provide more visibility to the other columns.

- *Source*: The source, i.e., MOOS App that generated the posting, is shown in the fourth column. This may be toggled on or off.
- *Auxiliary Source*: The auxiliary source is shown in the fifth column. Most MOOS apps do not make postings with an auxiliary source. The helm does however, and this typically indicates the particular behavior that produced the posting.
- *Variable Value*: The last column is just the value of the posting. This column cannot be hidden.

### 8.3 Adding and Removing Variables from the History List

Upon launching the *VarPlot* window, a single variable is chosen. After launching, additional variables may be added (from any vehicle) from the **Add Variable** pull-down menu button as shown in Figure 11. The **Set Variable** button presents the same set of choices, but will essentially remove all previously selected variables and replace it with the chosen one. The **Del Variable** button will remove the selected variable from the current set of variables.

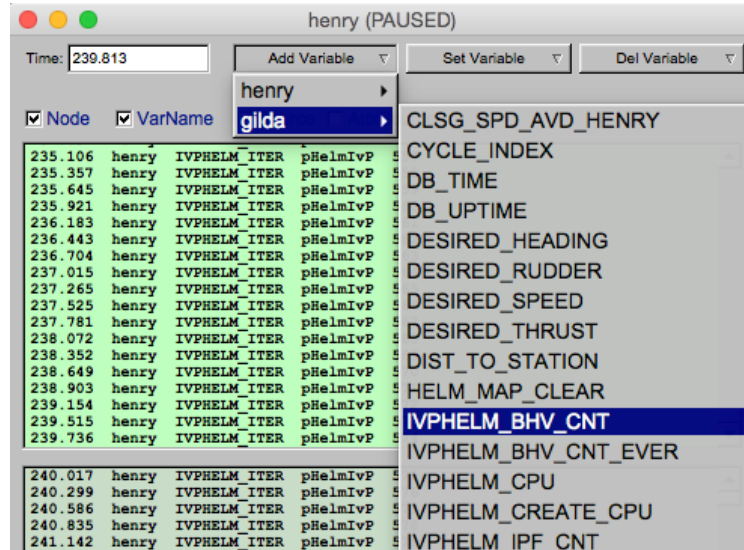


Figure 11: The *VarPlot* window: Any variable from any vehicle may be added any time after the window is opened.

### 8.4 Additional Buried but Useful Hot-Keys in the VarPlot Window

There are a few hot-key functions available that may not be apparent from simply interacting with the window:

- The '+' key: will increase the font size in the browser panes.
- The '-' key: will decrease the font size in the browser panes.

The above hot-keys are in addition to the time control hot-keys listed in Section `sec_alv_time_replay` and Section 3.1.

## 9 Command Line Usage for the alogview Tool

The `alogview` tool is run from the command line with one or more given `.alog` files and a number of options. The usage options are listed when the tool is launched with the `-h` switch:

```
$ alogview --help or -h
```

*Listing 9.1: Command line usage for the `alogview` tool.*

```
1  Usage:
2    alogview file.alog [another_file.alog] [OPTIONS]
3
4  Synopsis:
5    Renders vehicle paths from multiple MOOS .alog files. Replay
6    logged data or step through manually. Supports several
7    specialized pop-up windows for viewing helm state, objective
8    functions, any logged variable across vehicles. If multiple
9    alog files are given, they will be synchronized. Upon launch,
10   the original alog files are split into dedicated directories
11   to cache data based on the MOOS variable name.
12
13  Standard Arguments:
14    file.alog - The input logfile.
15
16  Options:
17    -h,--help      Displays this help message
18    -v,--version    Displays the current release version
19
20    --bg=file.tiff  Specify an alternate background image.
21
22    --lp=VEH:VAR    Specify starting left log plot.
23    --rp=VEH:VAR    Specify starting right log plot.
24                   Example: --lp=henry:NAV_X
25                   Example: --rp=NAV_SPEED
26
27    --nowtime=val   Set the initial startup time
28    --mintime=val   Clip all times/vals below this time
29    --maxtime=val   Clip all times/vals above this time
30
31    --quick,-q      Quick start (no geo shapes, logplots)
32    --altnav=PREF    Alt nav solution prefix, e.g., NAV_GT_
33
34    --zoom=val       Set initial zoom value (default: 1)
35    --panx=val       Set initial panx value (default: 0)
36    --pany=val       Set initial pany value (default: 0)
37
38    --geometry=xsmall Open GUI with dimensions 770x605
39    --geometry=small  Open GUI with dimensions 980x770
40    --geometry=medium Open GUI with dimensions 1190x935
41    --geometry=large   Open GUI with dimensions 1400x1100
42    --geometry=WxH     Open GUI with dimensions WxH
43
44  Further Notes:
45    (1) Multiple .alog files ok - typically one per vehicle
46    (2) See also: alogscan, alogrm, alogclip, aloggrep
47        alogsort, alogiter, aloghelm
```

The order of the arguments passed to `alogview` do not matter. The `--mintime` and `--maxtime` arguments allow the user to effectively clip the alog files to reduce the amount of data loaded into RAM by `alogview` during a session. The `--geometry` argument allows the user to custom set the size of the display window. A few shortcuts, "large", "medium", "small", and "xsmall" are allowed.