* **Front/Diagonal sensor distance/power** – the top two sets of graphs

The reason I have both is that navigation is done using distance and wall/edge decisions are made using power.

The left Y axis is power and the right Y axis is distance.

I send info. for both. I could send the translation table / function and send raw and have the host do the math. Didn’t have time during the initial implementation to do this.

This is a visual version of the data in the grid on the bottom right.

* **Forward / Rotational Profiler and Servo** – the bottom two sets of graphs

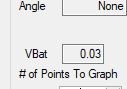
They show the velocity and error.

The left Y axis is velocity and the right Y axis is error.

This is a visual version of the data in the grid on the bottom right.

* **VBat** is captured every millisecond. It has helped me understand when the mouse suddenly stops with low battery voltage. I can look at it and work out that there isn’t a code bug but really the battery drooped.

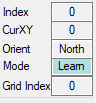
It is along the left edge, middle of the screen



* Just below VBat is **# of Points To Graph** – this is used to set the range for all four graphs. Sometimes I want a zoomed out view and other times I want to focus. It is **super** important to align all the data so you can get a complete picture



* This is mixture of info.



**Index** is the millisecond counter. Should have just called it that.

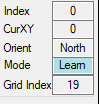
**CurXY** is the mouse’s current maze location

**Orient** which way is it pointing

**Mode** is it learning, backtracking, speedrun. The background color matches the color of the path displayed in the maze in the upper right.

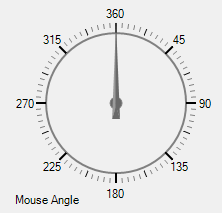
**Grid Index** is the message counter / line number in the grid of data on the bottom right.

In this example, notice that **Index** is zero; **Grid Index** is the line number within the data being presented.





* This tells me current mouse angle – basically rotational position. It is usually more cute then helpful but when single stepping through data it is very helpful.

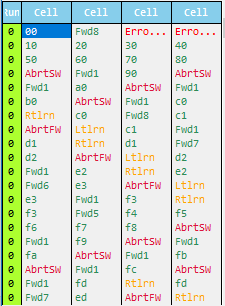


* This is a macro level view. It combines the:
  + Run number – left column in green. Used to quickly navigate between runs
  + Cell location – where was the mouse
    - Raw hex numbers
  + Motion command – what was is trying to do
    - Fwdn – move forward ‘n’ cells
    - Ltlrn – left learning turn
  + Maze learning decisions
    - AbrtSw – aborted path because we found a side wall that blocked it
    - AbrtFW – aborted path because we found a front wall that blocked it
  + Macro level
    - If something nasty happened, put it here.

It isn’t reasonable to scan the event log – ‘Comment’ column in the data grid because it consists of 17 columns and thousands of rows. So, put special things here.

I had a bug I didn’t understand. I added that to this display. It is shown as “Erro…”. By scanning this table, I can quickly see what I need to drill down into.

**All data is tied together**. Clicking on a cell in this table, move the graphs and the data grid and all other elements to the same event.



* The maze display is obvious. The path color changes based on learning VS backtracking VS speed runs. The color matches the background used for “**Mode”**.
* To the right of the maze display is errors output by the **host** environment. Examples are parsing errors
* The **data grid** / **grid of data** is super interesting because I use it to step through the minutiae.

The columns are:

**mm**: What mm is the mouse at

**LFP/LDP/RDP/RFP** – Left/Right Front/Diagonal Power i.e. raw sensor reading

**LFD/LDD/RDD/RFD** – Left/Right Front/Diagonal Distance i.e. how far the sensor thinks the wall is

**FVel/FErr/RVel/RPos/RErr** – Front/Rotational Velocity/Error/Position

**Lat** – state of the lateral corrector

**Long** – state of the longitudinal corrector

**Comment** – parameters associated with commands or events or?



* I always meant to annotate the graphs with lines that show me different events. If you look at the attached picture, I’ve circled in Orange these tick marks. I don’t remember what they correspond to but I think I only got one or two events in.