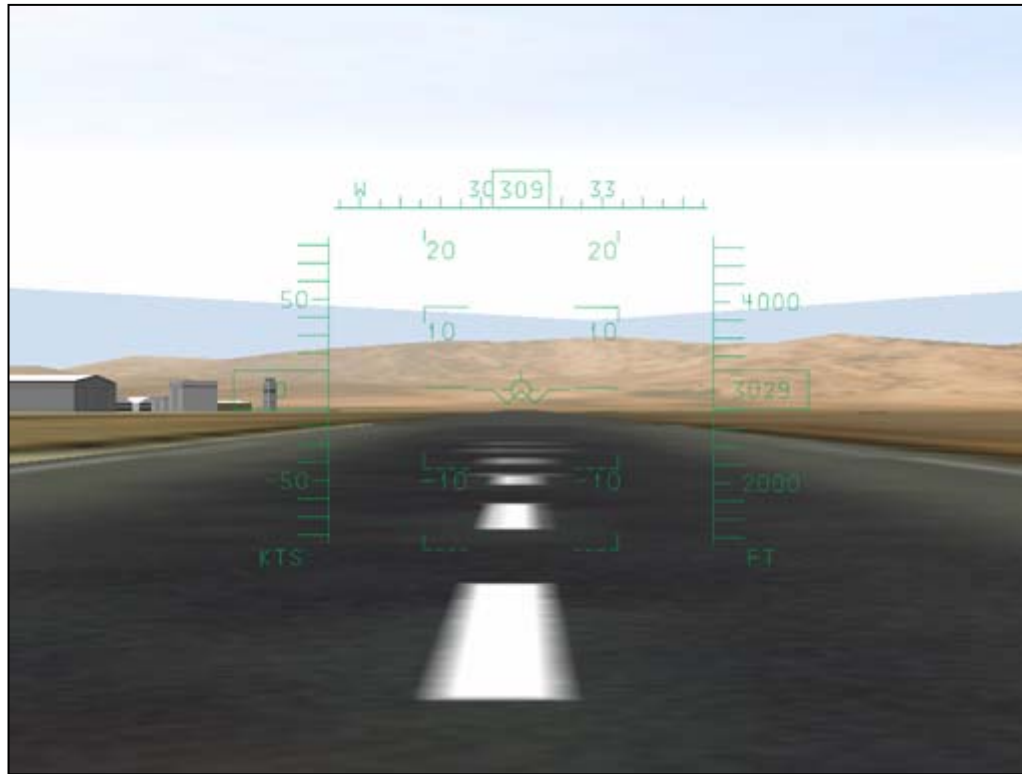


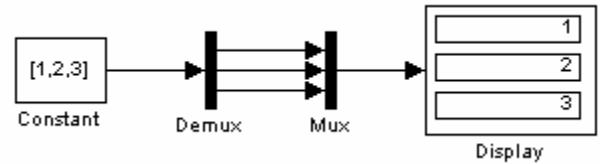
Simulink in the Pheagle Simulator



Chris Atkinson

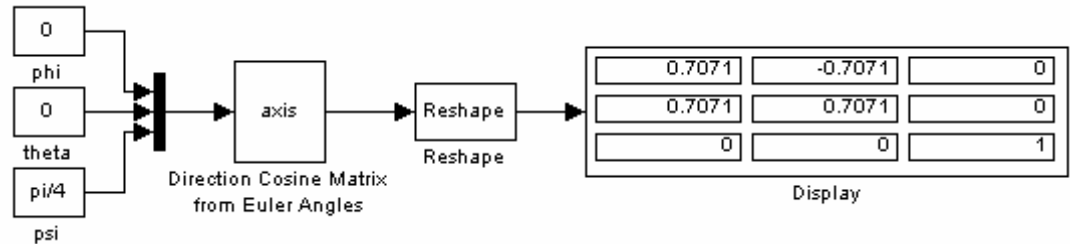
Aero 400 - Flight Simulation

Vectors



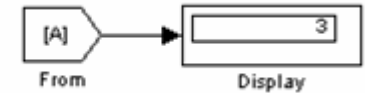
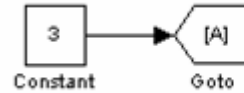
- Multiple values contained in one signal
- Constant vectors can be made by putting an array into a constant block using Matlab format... `[1,2,3]`
- Use Mux and Demux blocks under Signals & Systems to convert single signals to a vector and vice versa
- Bus selectors access one or more signal from a vector
 - They only work if the vector comes directly from a Mux or another Bus Selector
- Most signals in the simulator are three element vectors representing x,y,z axes (uvw, LMN, Fxyz)
- Other signals like buttons and engine data are Muxed for simplicity
 - The contents of the signals are listed near the block the signal originated from
- Vectors can be viewed directly with a display just like normal signals

Matrices



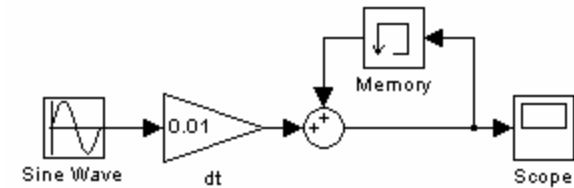
- Matrices can be made by Muxing multiple vectors
- Matrices can also be put in as constants in Matlab form [1,2,3;2,3,4;3,4,5]
- Transformation matrices can be made using the Simulink Direction Cosine Matrix block under Simulink Extras/Aerospace Blocks/Axes Transformations
 - The results come out as a 9 element vector
 - Use a Reshape block under Signals & Systems to convert it to a 3x3 matrix
 - If the inputs are [phi, theta, psi] then this will result in a body to earth axis transformation
- Set a Product block to do a matrix multiply to apply the transformation to a vector

Flags



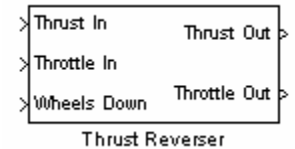
- In the simulator, flags are used to organize signals without using too many lines
- You can use a Goto block under Signals & Systems to create a flag for the source of the signal and one or more From blocks to access the signal
- Flags are identified by a tag
- The tag in the From block has to match the tag in the Goto block

Memory Blocks



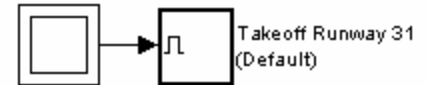
- If you create a loop in Simulink, it tries to solve it by iteration unless it is a continuous block like a transfer function or integrator
- Since the simulator already iterates to solve the equations of motion we want to use the value from the previous time step
- Memory blocks store the value of the signal for the next iteration
- They also stop the Simulink iteration process
- The initial value stored in the block when the simulation begins can be set in the block parameters

Subsystems



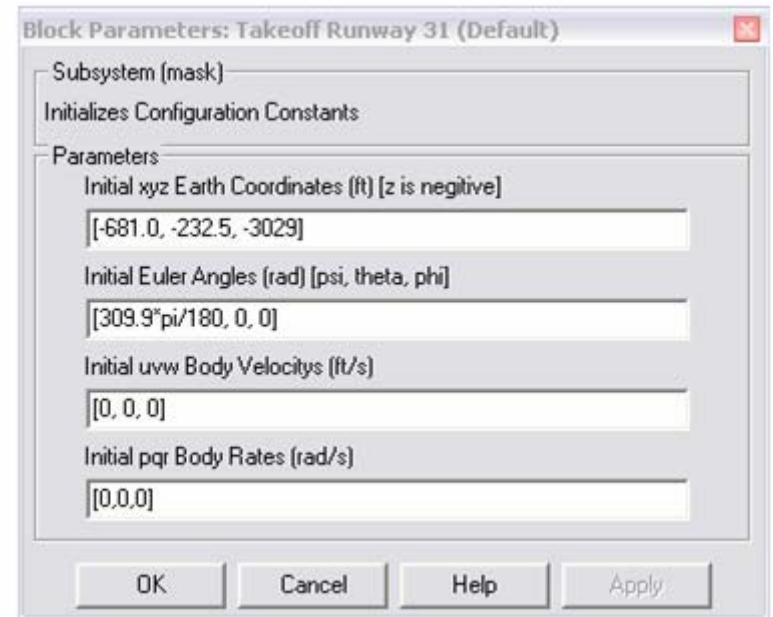
- To simplify the simulator model, subsystems are used to separate different parts of the model
- To make a subsystem, grab a Subsystem block from Signals & Systems and double click on it
- It will open a new Simulink window
- You can create the model for you subsystem there
- To make inputs and outputs for the subsystem, use Out1 and In1 blocks
- You can also put Triggered or Enabled blocks into the subsystem to create subsystems that only run at certain times
- You can create a mask for your subsystem so that someone can enter parameters in a GUI. The parameters are stored as variables for the subsystem to use (you can even customize the little picture on the block)

Push Buttons



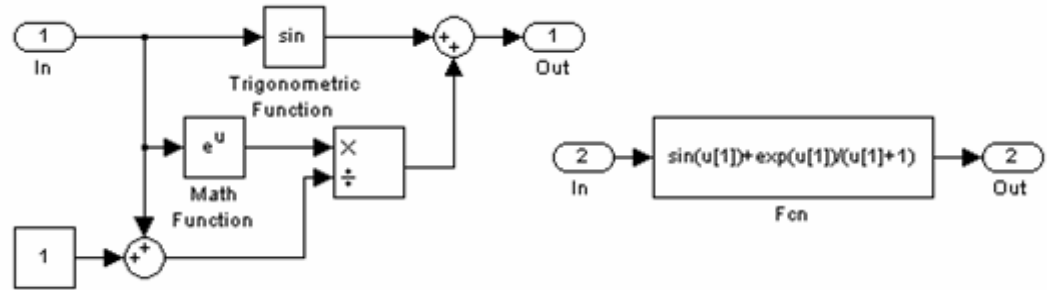
- The little blocks with squares in them are push buttons
- If you double click on them they will output 1 for one time step otherwise they output 0
- They are used to set different configurations or to reset the 6 DOF
- If you want a button to be activated when the model starts to set a default configuration, right click on it, go to mask parameters and check the “Activate on First Time Step” box
- You can’t find them in the Simulink library because I invited them, so just copy one from the simulator model
- If you want it to toggle between 1 and 0, you can take it apart (look under mask) and figure out how to make it do that
- You need “button.m” in the working directory for the buttons to work

Initial Conditions



- The initial conditions fed into the 6 DOF are done with Data Store blocks
- They are like flags except they can store data like a memory block
- The configuration blocks are enabled subsystems that write data to the data store blocks when they are activated by a button
- The blocks in the lower left corner of the model are the data store blocks they write to
- The blocks going into the 6 DOF read the data from the data store blocks
- The “Takeoff Runway 31” configuration is default, so its button is activated when the model starts

Math



- You can implement math functions using Simulink blocks, but it becomes confusing for complicated expressions
- You can use a Fcn block under Functions & Tables to type in the function in Matlab form
- You can access elements of the input vector using $u[n]$ where n starts numbering at 1

