Conflict Detection & Resolution Simulink Model

Aidan Wallace. University of Maryland, College Park. May 2021.

Contents

Model Overview

The main Simulink file for modeling fixed-wing UAV flight and CD&R is titled ‘metareasoning.slx’ and was created in Matlab R2020a. It borrows heavily from the Simulink aerospace blockset example, “Multiple Aircraft with Collaborative Control”. It contains three main blocks: “Fixed-Wing UAV [All]”, “Conflict Detection”, and “Conflict Resolution”. Coordinates for each aircraft are output to the Matlab workspace. The signal “States” contains busses of the velocities, flight path angles, headings, thrusts, bank angles, load factors, x, y, and h position of each aircraft in that order. The signal “Commands” contains busses of the thrust commands, bank angle commands, and load factor commands.

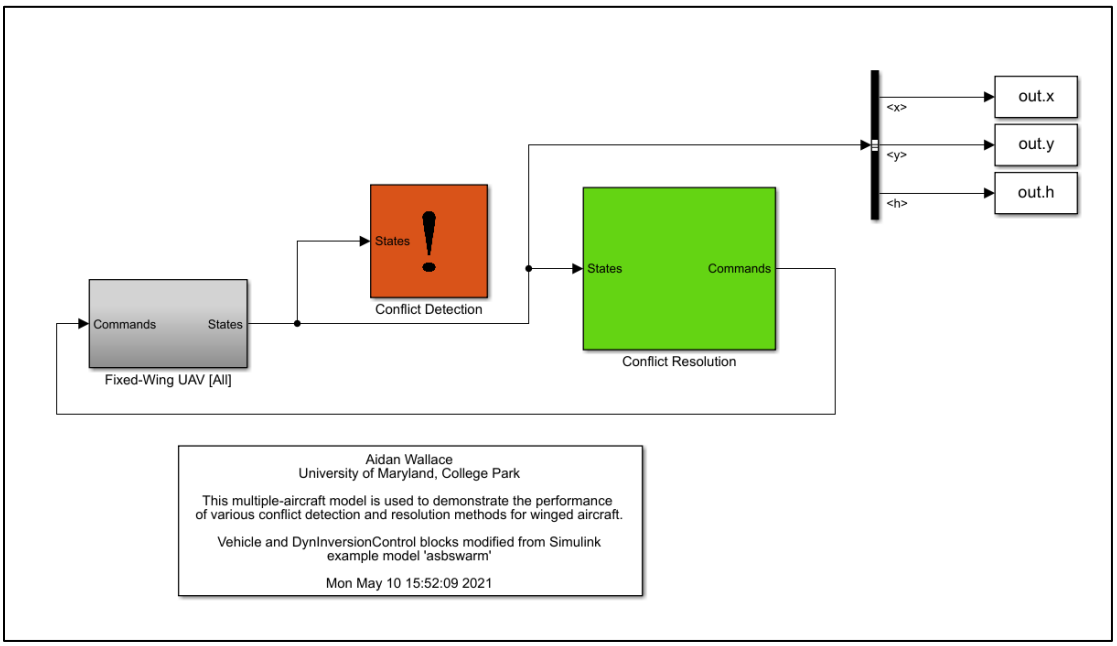

Figure 1: Model Overview

States: Velocity, gamma, chi, thrust, mu, loadfactor, x, y, h

Commands: Thrust_command, mu_command, loadfactor_command

Fixed-Wing UAV Block

"Fixed Wing UAV" models the aircraft dynamics. It receives a bus of **all** the aircraft commands and outputs a bus of **all** aircraft states to the detection and resolution blocks. This block is mostly untouched from the aerospace blockset example, aside from the small block "wrap", which converts the heading from an incorrect East-based heading to a North-based heading between -pi and pi. The red blocks are directly from the aerospace blockset.

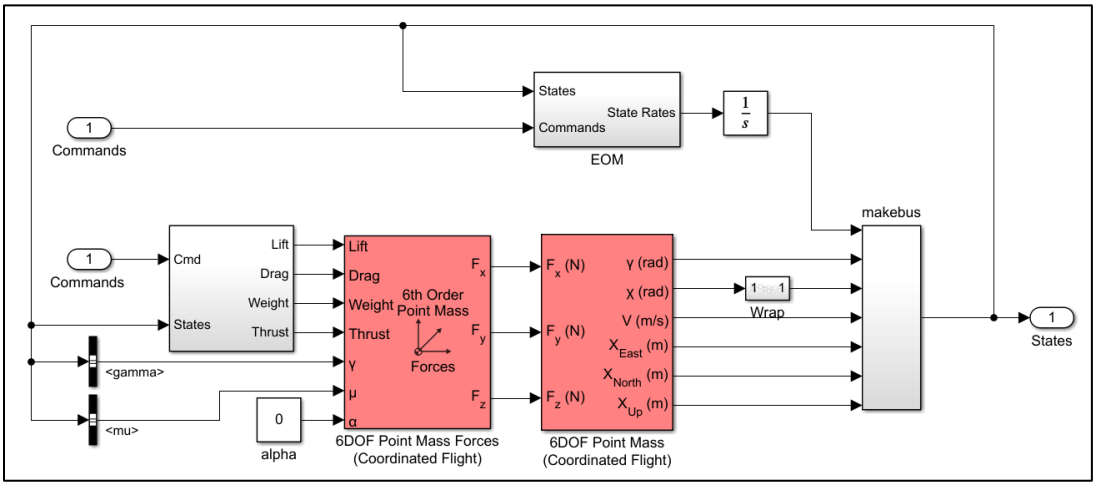

Figure 2: "Fixed-Wing UAV" Block

Conflict Detection Block

The conflict detection block produces two n x n (n = no. of aircraft in the scenario) containing information on time until los of separation (tlos) and time until the closest point of approach (tcpa). For example, the time in the (1,2) position or (2,1) position corresponds to the interaction between aircraft 1 and 2. These matrices are symmetrical with NaN on the diagonals. The main function used for this conflict detection is titled “statebasedCD”. For information on how this works, see the [BlueSky project on Github] (bluesky/docs/ASAS-CD&R-info.pdf, also available in the drive and Github repository). Currently, the outputs are unused, but will likely be of help in implementation for further resolution algorithms, such as the unfinished Erzberger 2010. A critical part of modeling single agent behavior is something called the “for each” block in Matlab. This allows ‘simultaneous’ calculation and scales down the problem substantially such that the problem doesn’t require large matrices and an only quasi-distributed approach.

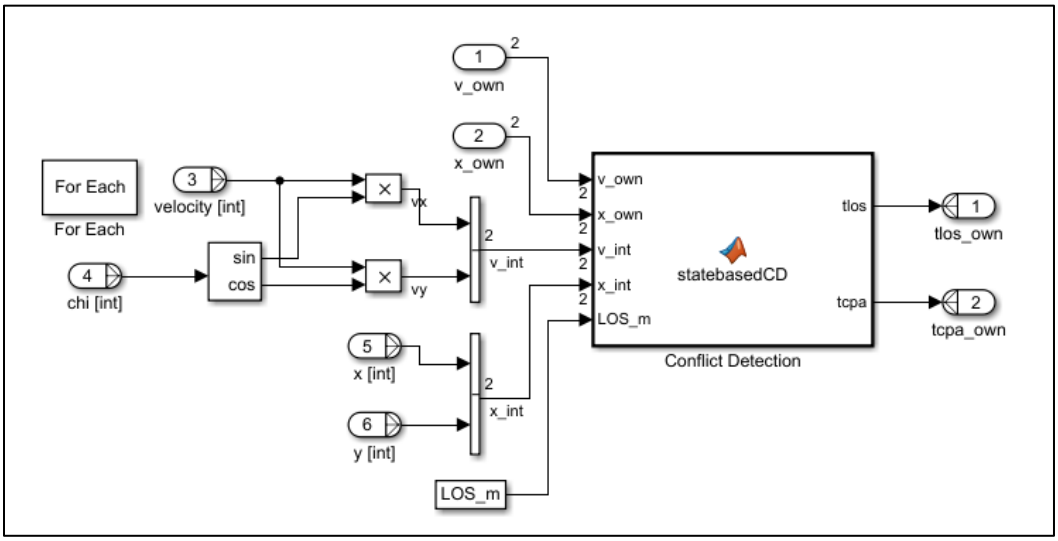

Figure 3: Third Layer of the "Conflict Detection" Block. The Magic Happens Here.

Conflict Resolution Block

The main goal of the “Conflict Resolution” block is to take the states of every aircraft in the scenario and output the next command for each aircraft. There are three main sections.

- “Metareasoning Policy” is currently unused and anticipates using the states and conflict information to output some type of indicator for the preferred resolution method.
- “Conflict Resolution Techniques” contains the BYPASS block, which when active will cause the aircraft to ignore any conflicts and automatically updates the desired states to be the current state (normal cruising aircraft). The other blocks are different resolution methods which will be discussed in a later section.
- “Desired States to Commands” contains the “DynInversionControl” block which directly comes from the aerospace blockset example. This block must be active for any resolution technique that issues desired states rather than direct commands of bank angle, loadfactor, and thrust. It is important to note that **this block includes complex dynamics like wing stall**. Therefore, it is recommended that the reader closely inspect this block.

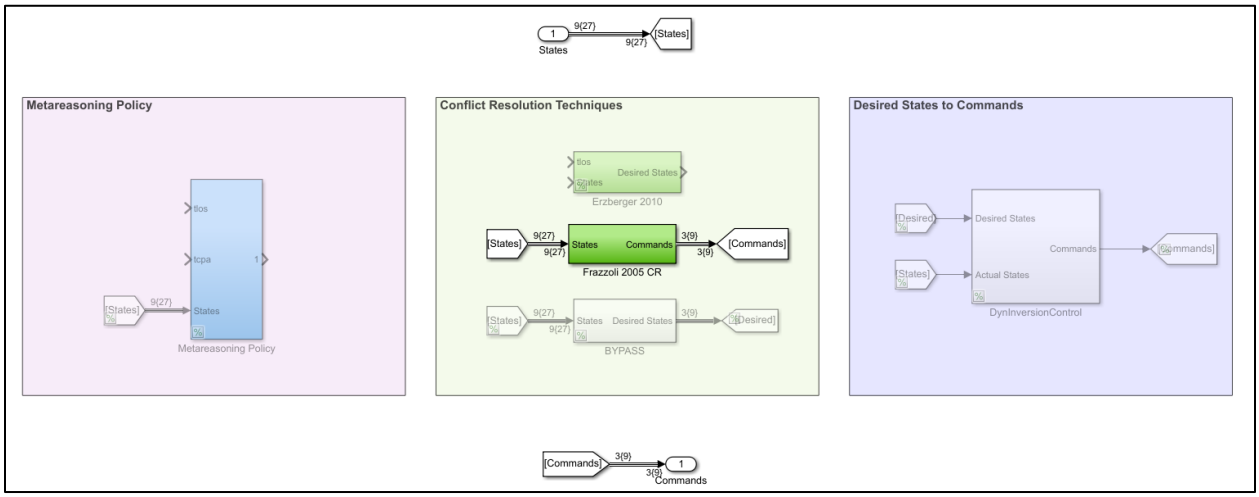

Figure 4: "Conflict Resolution" Block

Callback Functions

Callback functions are critical for the proper functionality of the model. Callback functions are Matlab functions which are run at various points in the execution of the model. To view and assign callback functions, in Simulink go to Model Explorer in the "Modeling" tab. Select the model in the model hierarchy on the left. Then, on the right under "Model Properties" select the "Callbacks" tab. The current callbacks in use are:

- PreLoadFcn –Loads necessary variables into the Matlab workspace. Only executed once upon opening the model. Current function is "Metareasoningpreloadfcn.m".
- InitFcn –Executed whenever the model is run. Current function is "MetareasoningInitFcn.m". Used to declare initial conditions for the aircraft and desired parameters specific to that run. **Note: Heading in the InitFcn must currently be defined based off East.**
- StopFcn –Executes after a simulation run is terminated. Current function is "trajectory_plot.m".

See the Simulink documentation for more information on callback functions if you intend to alter these or implement more.

Resolution Methods

Frazzoli 2005

The Frazzoli 2005 resolution method uses an agreed-upon "hold" radius called the reserved region. The reserved region is a circle centered R meters to the right of the aircraft with radius R + $d_s/2$. The act of holding entails indefinitely turning right in a circle with radius R. The purpose of this reserved region is to ensure that separation is never lost between aircraft and to provide an entity that is perfectly maneuverable, unlike the aircraft with certain turn radii. For this resolution method to work, goal positions must be input for these reserved regions (**Note: not the aircraft!)**. In the current implementation, aircraft will hold unless their reserved region is not immediately obstructed (an obstructed reserved region is blocked from its goal by another that it is directly touching). They will fly straight if their heading matches the goal bearing and the reserved region is unobstructed. Currently, a third state called "roll" is not implemented, which expedites resolution by having aircraft turn left on the boundary of another's reserved region.

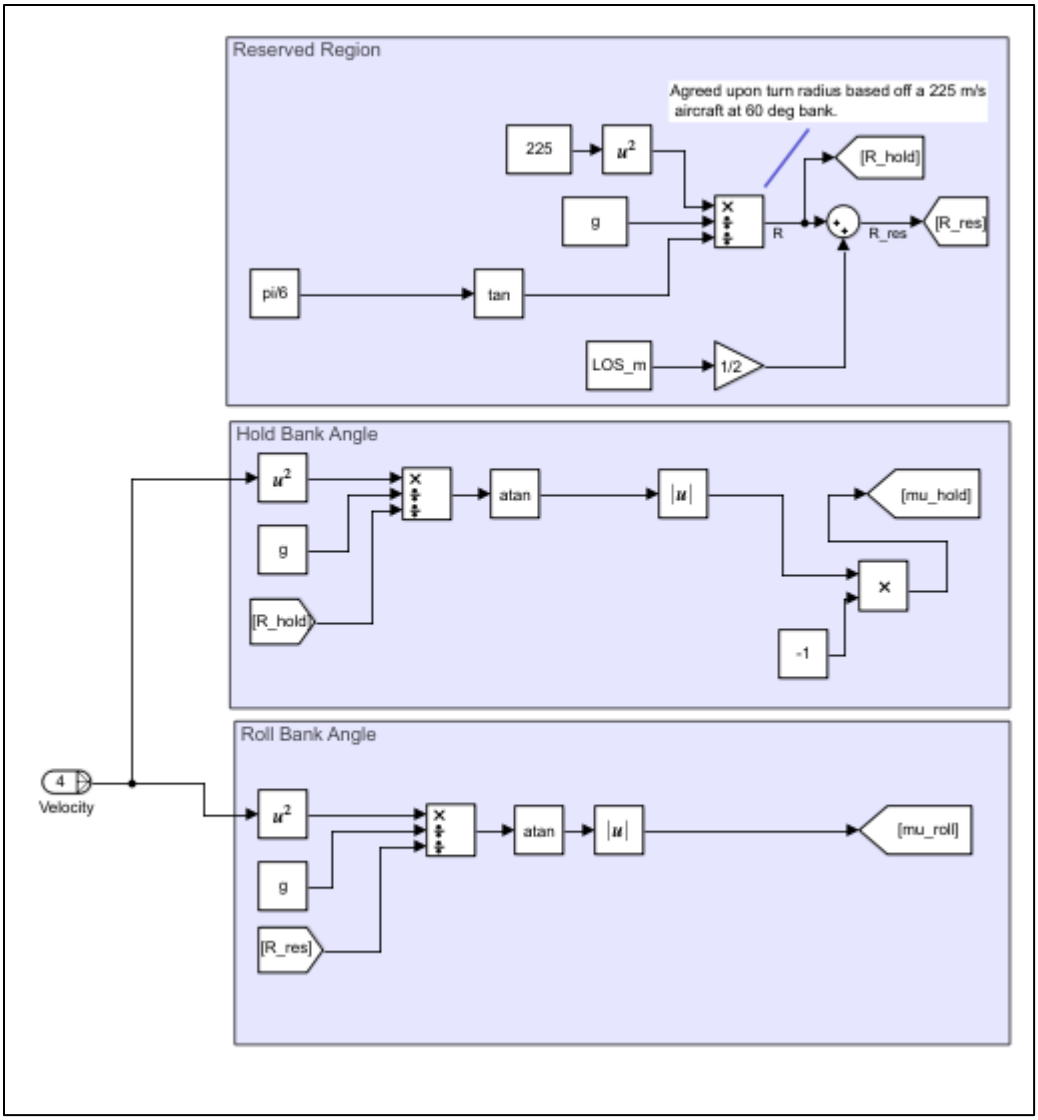

Figure 5: Scenario-Wide Resolution Parameters

First, as shown in figure 5, an agreed upon reserved region radius and holding turn radius are calculated based off a reference aircraft. Using these radii, a holding bank angle and rolling bank angle (unimplemented) are calculated for each aircraft.

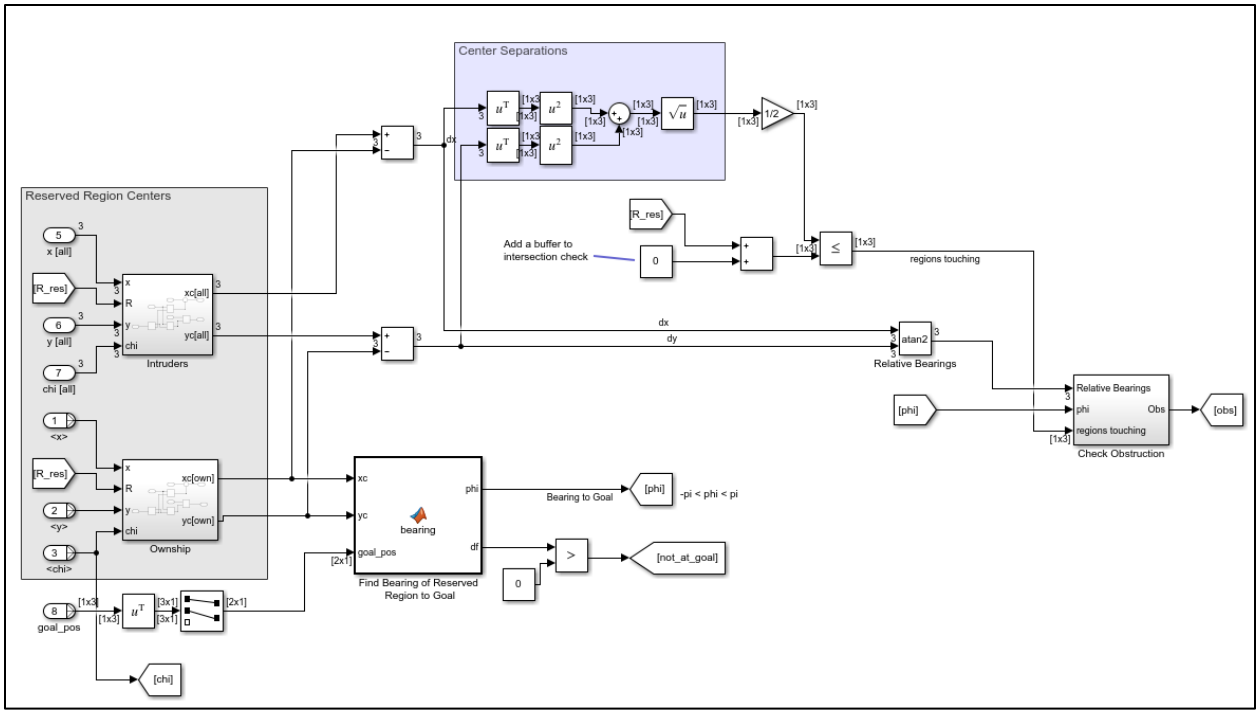

Figure 6: Checking Reserved Region Obstruction

First, the coordinates of the reserved regions are all calculated for the focus aircraft (ownship) and all of them (ownship + intruders). Using the center of the ownship's reserved region, the bearing to the goal and the distance to the goal > 0 are then calculated and stored in "phi" and "not_at_goal", respectively. Next, the separation magnitude between the center of the ownship's reserved region and the center of the intruders' reserved regions are calculated and halved. If this distance is less than the radius of the reserved region (remember, these radii are uniform) then that indicates the reserved regions are touching.

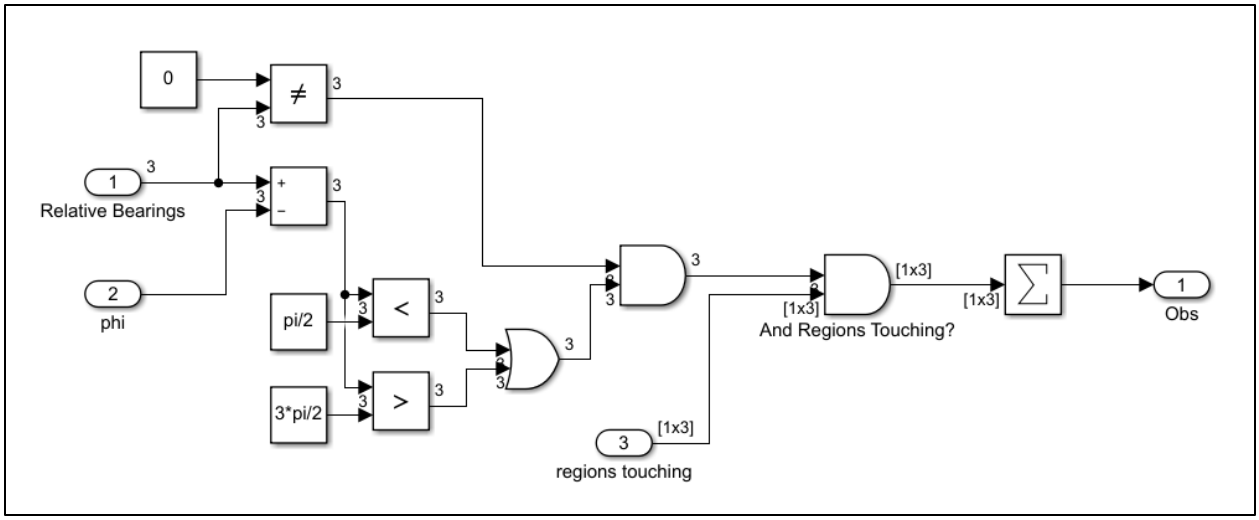

Figure 7: "Check Obstruction" Block

Next, we check to see if the ownship's reserved region is obstructed by another. First, the relative bearing of each reserved region to the ownship's is found (bound between -pi and pi). Next, these relative bearings are compared to the aircraft's heading. Since the circles must be touching, any heading within 90 degrees of the goal bearing would indicate that the path is obstructed. Then, the relative bearing of zero is excluded from the logic, as this is one reserved region being compared to itself. Lastly, the logic gate of path obstruction and touching regions is executed. Since this will create an obstruction bool for each pair, the vector is summed. Any summation not equal to zero means that the ownship's reserved region is obstructed. This summation is stored in "Obs".

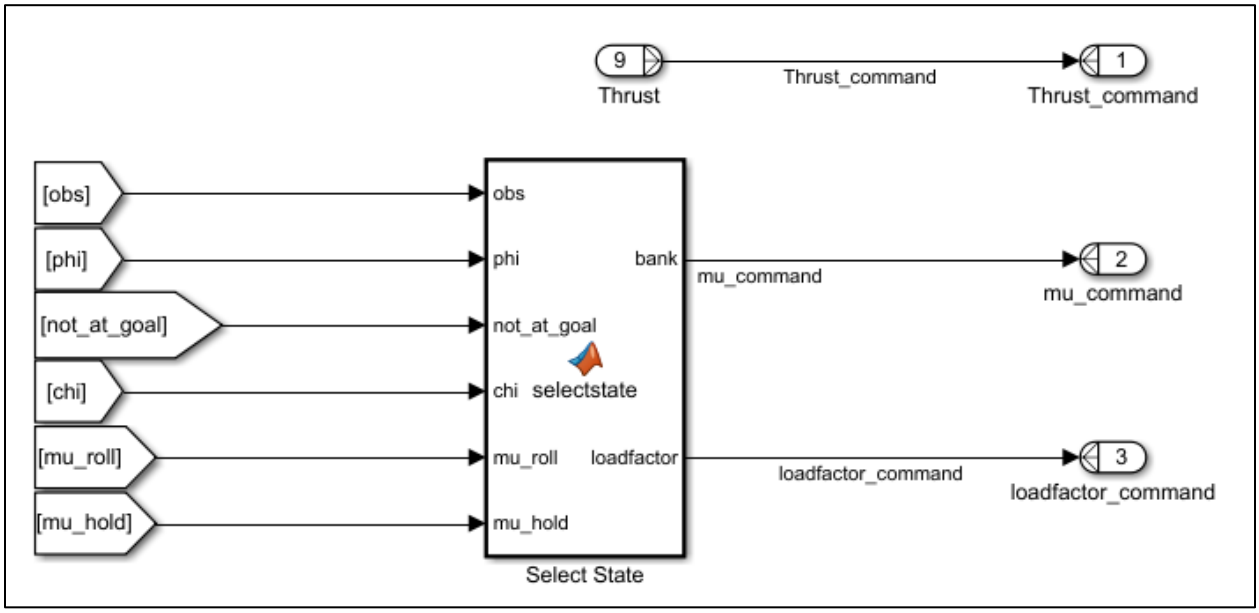

Figure 8: Assigning Aircraft State

Lastly, the “Select State” Matlab function takes in all the previously calculated variables and assigns the necessary commands corresponding to a hold or straight state.

It is recommended that the reader take the time to read *Decentralized Cooperative Conflict Resolution for Multiple Nonholonomic Vehicles* by Frazzoli, Pallottino, Scordio, and Bicchi. Summary slides are also located in the drive and Github repository.

Erzberger 2010 (Unfinished)

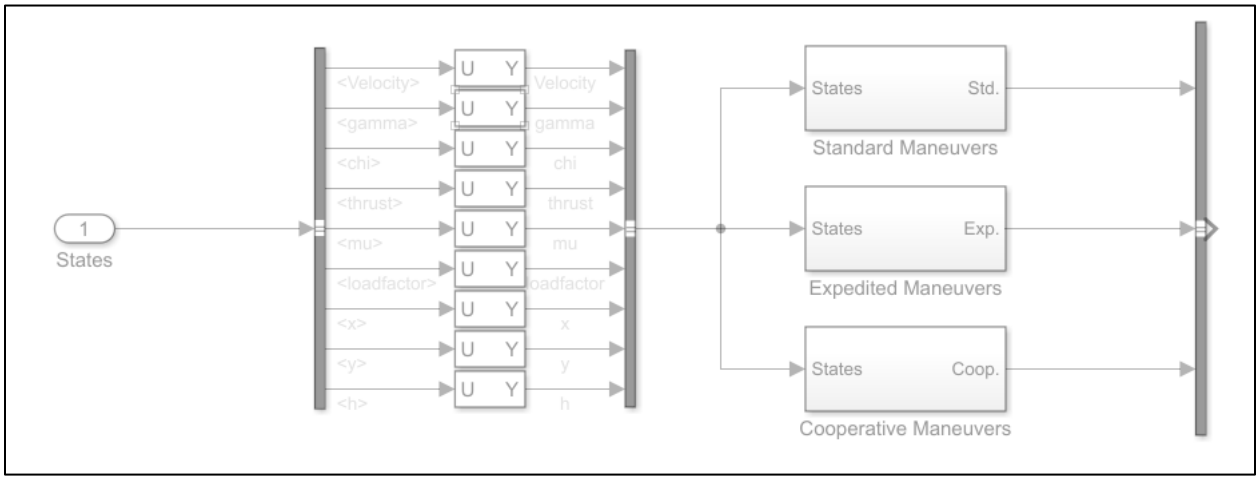

Figure 9: Unfinished Erzberger 2010 Resolution Block

Summaries for this algorithm can be found in the drive/Github repository. Essentially, parameters for 12 resolutions are found. Depending on the characteristics of the resolution, it is either ranked 1, 1a, 2, or 2a and chosen based off these rankings. The 12 resolutions are:

- Standard (15 deg bank)
    - A left, B straight
    - B left, A straight
    - A right, B straight
    - B right, A straight
- Expedited (30 deg bank)
    - A left, B straight
    - B left, A straight
    - A right, B straight
    - B right, A straight
- Cooperative (Both move @ 30 deg bank)
    - A right, B right

- A left, B right
- A right, B left
- A left, B left

The current implementation finds correct resolution parameters for these 12 types in their relevant Matlab functions. However, it is only designed for a n = 2 scenario, and at this time does not issue commands based off the highest ranked resolution. This resolution block was created before the full utility of the “for each” block was recognized, thus only being able to consider 2 aircraft. For future work, this would likely be a useful tool.

Here are some anticipated key considerations:

- Downstream conflicts.
- Aircraft must only partake in a single resolution with another.
- Aircraft must execute resolution maneuver once and flag that resolution as in progress so as not to keep updating with a new conflict geometry. This resolution is not a feedback loop.
- Type 2 resolutions (failures) are not yet implemented.
- Complex maxima checking detailed by paper not yet implemented.