5. Effector Combination Logic

The effector combination or mixing logic is a matrix (Kmix) that connects between the flight control system outputs and the actuator inputs. Its purpose is to convert the FCS acceleration demands to TVC or aerosurface deflections or to thrust variations commands. Flight vehicles are in general controlled by multiple and different types of effectors that produce moments and forces in 3 or more directions, mainly 3 rotations and optionally some translations. The effectors are thrust vector control (TVC) engines, thrust varying (throttling) engines, control surfaces, and reaction control jets (RCS) that provide the "muscle" power to maneuver the vehicle. The mixing logic combines the vehicle effectors together as a system and becomes an integral part of the flight control software. In the event of an effector failure it is the mixing logic matrix that must be adjusted instead of the FCS gains. Figure 5.1 shows a mixing logic matrix for a typical flight vehicle that is controlled by different types of effectors. The inputs are acceleration demands coming from the flight control system, and they are converted to TVC pitch and yaw deflections, throttle commands, and aerosurface deflections that drive the control actuators. The FCS demands are functions of commands minus measurements that control the vehicle attitude and flight direction. They are mainly 3 rotational acceleration demands and may also include some translational demands, such as, accelerations along X, Y and Z. Translational control is used when translation or velocity control is necessary independently of rotations, such as, during vehicle separation, hovering at low speeds, or controlling the rate of descent. This is possible, of course, when the vehicle has the effector capability to generate translations, such as, a throttling engine, jets, body-flap, or a speed-brake to provide linear control along those directions.

The effector sizing is based on requirements defined by vehicle performance goals. The effectors as a system must be capable of providing the required accelerations for maneuverability and the control authority to react against disturbances in the controlled directions, which are at least 3 rotations, plus some translations. The Flixan mixing logic algorithm described in this section optimizes the actuator effectiveness, because it takes into consideration the vehicle geometry, thrusts, angle of attack, mass properties, aero-surface coefficients and the capability of each effector in the required directions. It maximizes the vehicle response in the commanded directions using minimum deflections. It uses pseudo-inversion to determine an optimal combination of the controls that achieve the demanded accelerations while reducing cross-coupling between the control axes. When the matrix is connected open-loop in series with the vehicle model, as shown in Figure 5.2, it attempts to diagonalize the plant which means that the vehicle accelerations approximate the accelerations requested by flight control. This, of course, is true when we ignore the aerodynamics of the base vehicle. The matrix will provide the proper accelerations. However, the vehicle will eventually diverge if it is open-loop unstable. That is why we need feedback stabilization. This pre-multiplication of the vehicle with the effector mixing matrix creates a plant model that is more efficient for control design because it already includes cross-axes decoupling. An efficient mixing logic should be time-varying because the control authority of the effectors changes as a function of geometry, dynamic pressure, angle of attack, thrust, and CG location. The derivation of a mixing logic matrix for a vehicle that is controlled by gimbaling engines, throttling engines or jets, and control surfaces is presented in the next section.

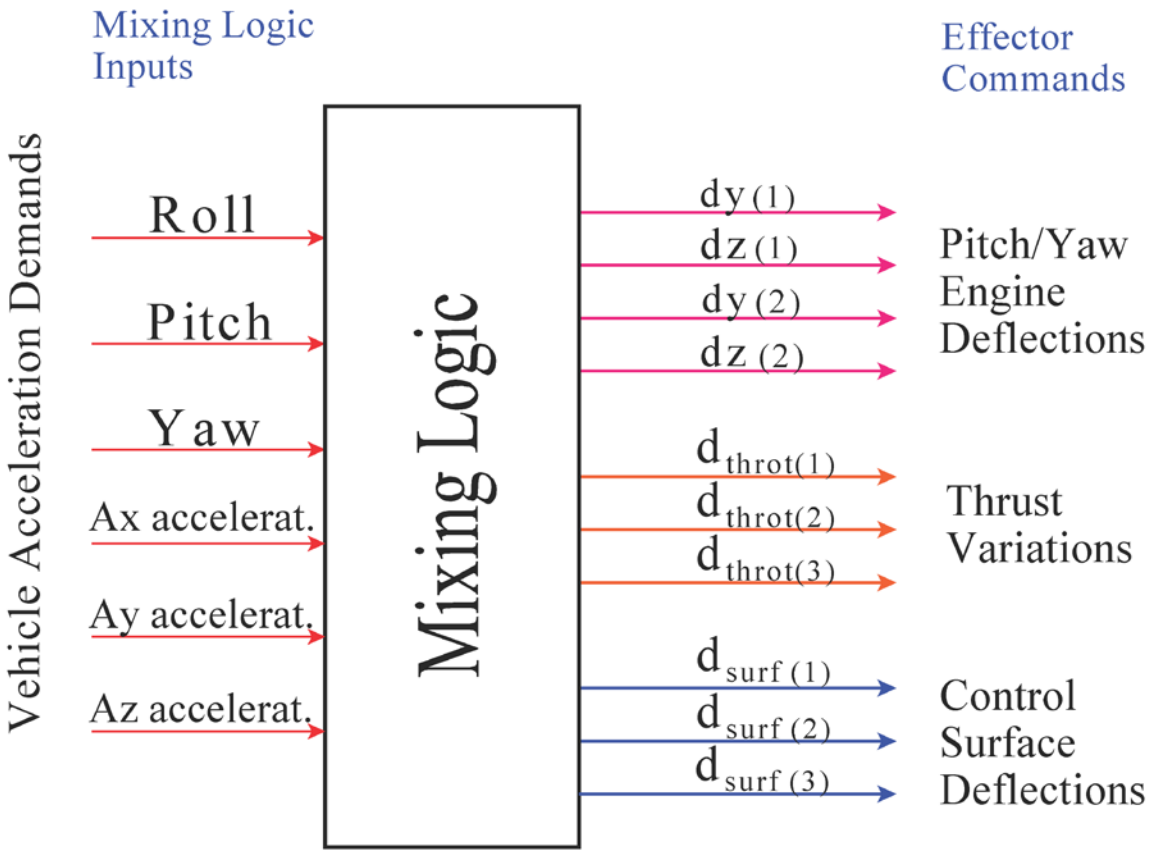


Figure 5.1 Effector Combination Matrix

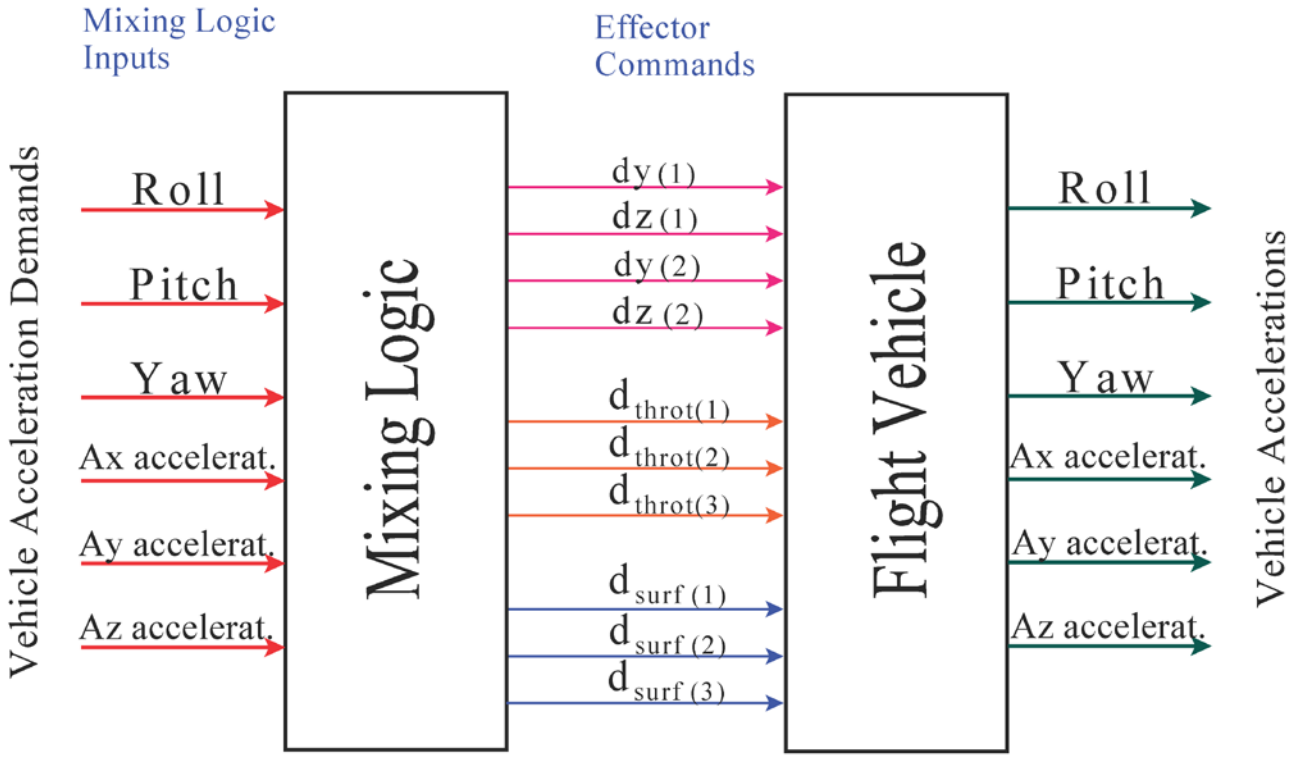


Figure 5.2 When the Mixing Logic Matrix is connected in Series with the Vehicle Model (Open-Loop) the Vehicle Accelerations should be approximately equal to the accelerations demanded by the FCS.

5.1 Forces and Moments Generated By a Single Engine

The following equation calculates the forces generated by a single thruster engine (i) mounted on a vehicle at fixed orientation angles (or trimmed at those angles): Δ_E in pitch (elevation angle with respect to the x y plane), and Δz in yaw (azimuth angle about the body z axis), see Figure (5.3). The forces along the body x, y, and z axes are:

$$\begin{aligned} F_{Xe(i)} &= T_{e(i)} \cos(\Delta_E) \cos(\Delta_Z) \\ F_{Ye(i)} &= T_{e(i)} \cos(\Delta_E) \sin(\Delta_Z) \\ F_{Ze(i)} &= -T_{e(i)} \sin(\Delta_E) \end{aligned} \qquad (5.1.1)$$

Let us define the throttle control $D_{th(i)}$ for engine (i) to be the ratio of thrust variation divided by the nominal engine thrust.

$$D_{th(i)} = \frac{\delta T_{e(i)}}{T_{e(i)}} \quad where: \delta T_{e(i)} \; is \; the \; Thrust \; Variation \qquad (5.1.2)$$

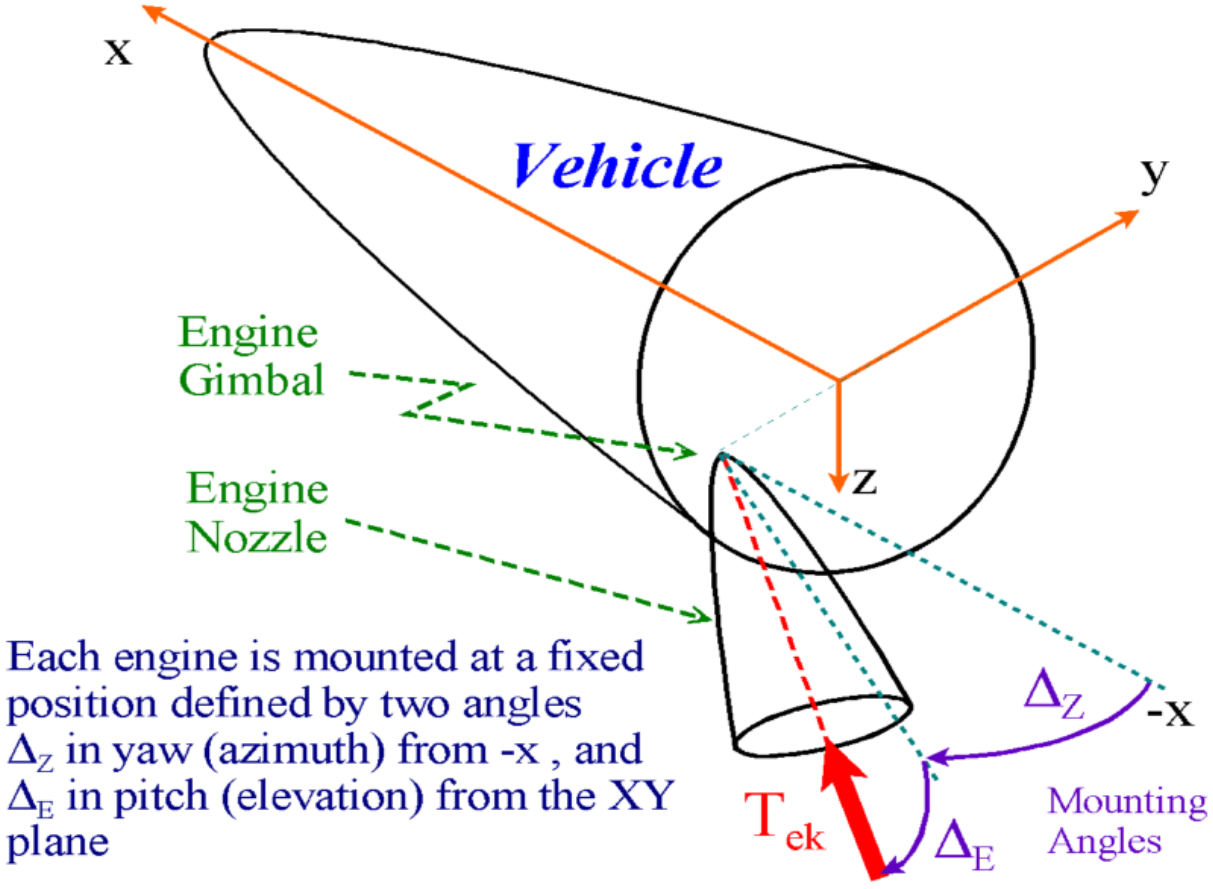


Figure 5.3 Engine Orientation Angles (Δy and Δz) with respect to the Vehicle Body Axis

The product Dth(i)*$T_{e(i)}$ = $\delta T_{e(i)}$ is the variation of engine thrust force above or below its nominal thrust value $T_{e(i)}$. Equation 5.1.3 calculates the force variation at the gimbal of an engine (i) due to the combined effects of gimbaling and throttling, resolved along the vehicle x, y, and z axes.

$$
\begin{aligned}
F_{xe(i)} &= T_{e(i)}\left[-s(\Delta_E)\,c(\Delta_Z)\,\delta y_{(i)} - c(\Delta_E)\,s(\Delta_Z)\,\delta z_{(i)} + c(\Delta_E)\,c(\Delta_Z)\,Dth_{(i)}\right] \\
F_{ye(i)} &= T_{e(i)}\left[-s(\Delta_E)\,s(\Delta_Z)\,\delta y_{(i)} + c(\Delta_E)\,c(\Delta_Z)\,\delta z_{(i)} + c(\Delta_E)\,s(\Delta_Z)\,Dth_{(i)}\right] \\
F_{ze(i)} &= T_{e(i)}\left[-c(\Delta_E)\,\delta y_{(i)} - s(\Delta_E)\,Dth_{(i)}\right]
\end{aligned}
\qquad (5.1.3)
$$

Let us define the distances between the engine (i) gimbal to the vehicle CG, {$l_{xe(i)}$, $l_{ye(i)}$, $l_{ze(i)}$} as follows

$$
l_{xe(i)} = X_{e(i)} - X_{CG} \qquad l_{ye(i)} = Y_{e(i)} - Y_{CG} \qquad l_{ze(i)} = Z_{e(i)} - Z_{CG} \qquad (5.1.4)
$$

The roll, pitch, and yaw moments on the vehicle resulting from the forces generated by a single engine (i) are obtained from the following matrix equation

$$
\begin{bmatrix} L_{e(i)} \\ M_{e(i)} \\ N_{e(i)} \end{bmatrix} = \begin{bmatrix} 0 & -l_{zei} & l_{yei} \\ l_{zei} & 0 & -l_{xei} \\ -l_{yei} & l_{xei} & 0 \end{bmatrix} \begin{bmatrix} F_{xe(i)} \\ F_{ye(i)} \\ F_{ze(i)} \end{bmatrix} \qquad (5.1.5)
$$

We will now calculate the moment and force variations in the vehicle body axes generated by each individual effector and combine them together as a system. This is, due to gimbaling, throttling, and also due to the control surface deflections. The contribution of each effector will be included and we will derive an expression for the total vehicle moments and forces as a function of the contributions from all effectors. One more detail that will be considered in the mixing logic calculations is the maximum capability of each effector. This consideration is important because the various engines or aero surfaces may have different maximum deflection angles or throttling capabilities. We must derive, therefore, a mixing law that will take into consideration the effector capabilities according to their peak contributions in each direction, by spreading the control authority evenly among the effectors proportionally, according to their capabilities. For example, if two engines have equal thrust but different gimbaling capabilities, the engine with the larger rotational capability should be allowed to deflect at a larger angle than the engine with the smaller rotation range. Ideally, they should all reach to their saturation limits together when the control demand is exceeded. This maximizes the control effectiveness.

5.2 Moments and Forces Generated by a Single Engine Gimbaling in Pitch and Yaw

Consider an engine (i) which is mounted at fixed elevation and yaw angles D_E(i) and D_Z(i) respectively, see figure (5.3). The engine is further gimbaling at small angles δy(i) and δz(i) in pitch and yaw directions with respect to the mounting positions. The moment variations on the vehicle are obtained from the equation (5.2.1).

$$
\begin{pmatrix} L_{g(i)} \\ M_{g(i)} \\ N_{g(i)} \end{pmatrix} = T_{e(i)} \begin{pmatrix} 0 & -l_{zei} & l_{yei} \\ l_{zei} & 0 & -l_{xei} \\ -l_{yei} & l_{xei} & 0 \end{pmatrix} \begin{pmatrix} -c(\Delta_Z)s(\Delta_E) & -c(\Delta_E)s(\Delta_Z) \\ -s(\Delta_Z)s(\Delta_E) & +c(\Delta_E)c(\Delta_Z) \\ -c(\Delta_E) & 0 \end{pmatrix} \begin{pmatrix} \delta_{y(i)} \\ \delta_{z(i)} \end{pmatrix} \qquad (5.2.1)
$$

This equation can be normalized by dividing the pitch and yaw engine deflections with the max deflection capabilities in both directions, so that the normalized inputs can vary between {0 and ±1} as follows:

$$\begin{pmatrix} L_{g(i)} \\ M_{g(i)} \\ N_{g(i)} \end{pmatrix} = T_{e(i)} \begin{pmatrix} 0 & -l_{zei} & l_{yei} \\ l_{zei} & 0 & -l_{xei} \\ -l_{yei} & l_{xei} & 0 \end{pmatrix} \begin{pmatrix} -c(\Delta_Z)s(\Delta_E)\delta_{y\max} & -c(\Delta_E)s(\Delta_Z)\delta_{z\max} \\ -s(\Delta_Z)s(\Delta_E)\delta_{y\max} & +c(\Delta_E)c(\Delta_Z)\delta_{z\max} \\ -c(\Delta_E)\delta_{y\max} & 0 \end{pmatrix} \begin{pmatrix} \delta_{y(i)}/\delta_{y\max} \\ \delta_{z(i)}/\delta_{z\max} \end{pmatrix}$$

By multiplying out the matrices in the above equation, it be expressed in a simplified form as follows:

$$\begin{pmatrix} L_{g(i)} \\ M_{g(i)} \\ N_{g(i)} \end{pmatrix} = \begin{pmatrix} | & | \\ V_{gyi} & V_{gzi} \\ | & | \end{pmatrix} \begin{pmatrix} \delta_{y(i)}/\delta_{y\max} \\ \delta_{z(i)}/\delta_{z\max} \end{pmatrix} \qquad (5.2.3)$$

Where: $V_{gy(i)}$ and $V_{gz(i)}$ are column vectors that correspond to the pitch and yaw engine deflections respectively.

Forces of an Engine Gimbaling in Pitch and Yaw Directions

Similarly, the forces applied at the gimbal due to an engine (i) gimbaling in pitch and yaw can be resolved along the body x, y, and z axes and normalized by dividing the pitch and yaw deflections with the max deflections as shown in the following equation, written also in column vector form:

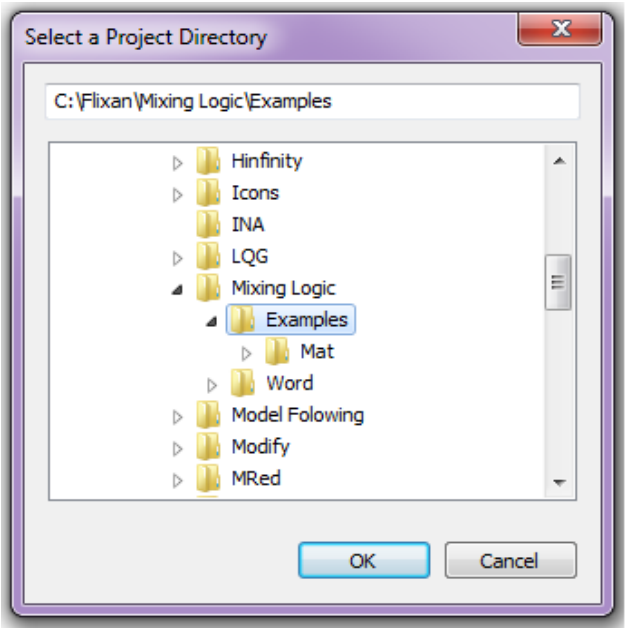
$$\begin{pmatrix} F_{X(i)} \\ F_{Y(i)} \\ F_{Z(i)} \end{pmatrix} = T_{e(i)} \begin{pmatrix} -c(Dz)s(Dy)\delta_{y\max} & -c(Dy)s(Dz)\delta_{z\max} \\ -s(Dz)s(Dy)\delta_{y\max} & +c(Dy)c(Dz)\delta_{z\max} \\ -c(Dy)\delta_{y\max} & 0 \end{pmatrix} \begin{pmatrix} \delta_{y(i)}/\delta_{y\max} \\ \delta_{z(i)}/\delta_{z\max} \end{pmatrix}$$

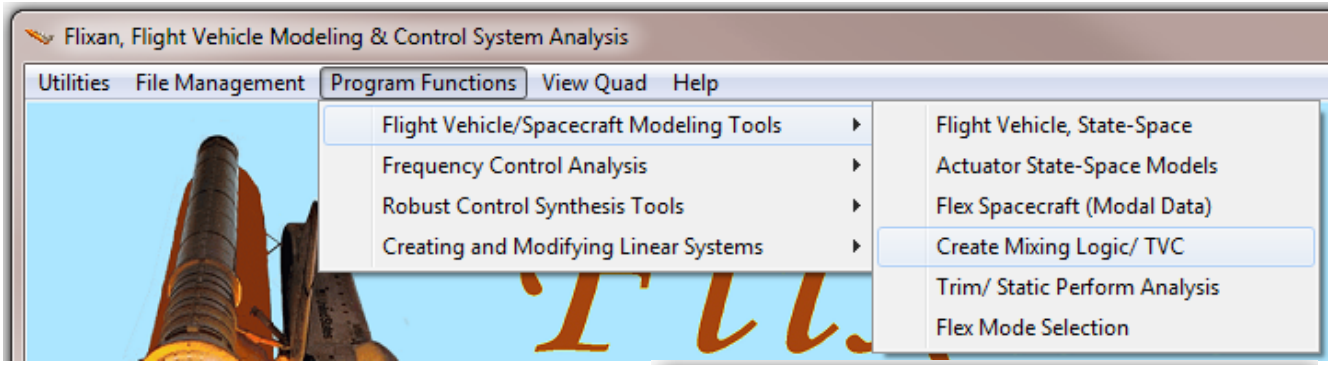
$$\begin{pmatrix} F_{X(i)} \\ F_{Y(i)} \\ F_{Z(i)} \end{pmatrix} = \begin{pmatrix} | & | \\ U_{gyi} & U_{gzi} \\ | & | \end{pmatrix} \begin{pmatrix} \delta_{y(i)}/\delta_{y\max} \\ \delta_{z(i)}/\delta_{z\max} \end{pmatrix} \qquad (5.2.4)$$

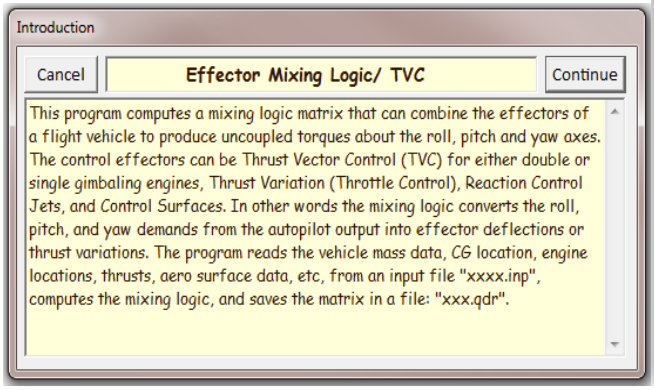
Where: $U_{gy(i)}$ and $U_{gz(i)}$ are column vectors that correspond to the pitch and yaw engine deflections respectively.

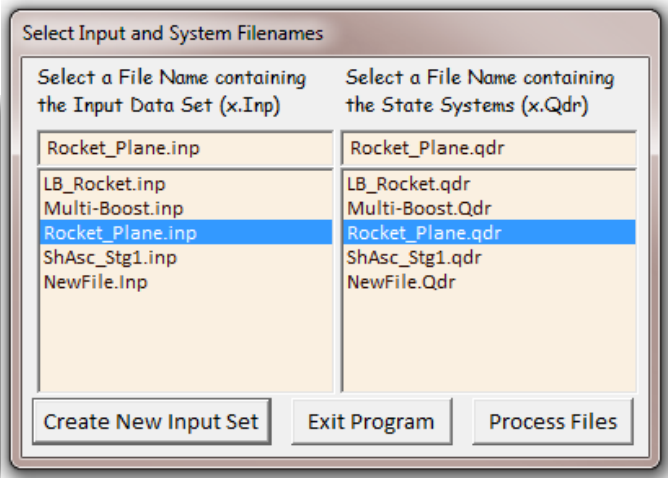
5.7 The Mixing Logic Program

The mixing logic calculation algorithm is an option in the Flixan program. It requires a vehicle data-set, such as the set used by the vehicle modeling program, and it uses the mass properties and the effector data information to calculate the mixing logic matrix. To run it, start the Flixan program and select the folder that contains the vehicle data. Then go to the Flixan main menu, and select "*Program Functions*", *"Flight Vehicle/ Spacecraft Modeling Tools"*, and then "Create *Mixing Logic/ TVC*". The filenames selection menu comes up, and the user selects two filenames: the input data file (Rocket-Plane.Inp) that contains the vehicle data, and the systems file (Rocket-Plane.Qdr) for saving the effector mixing matrix. Click on "*Process Files*" button to continue.

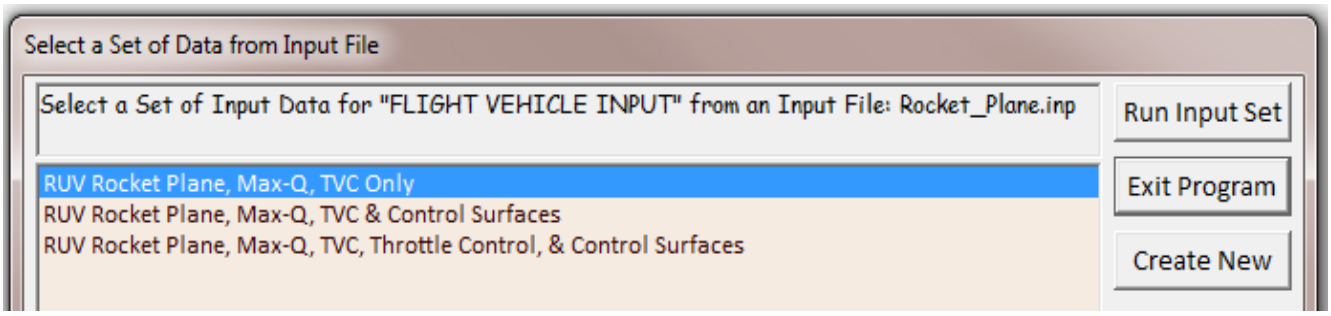




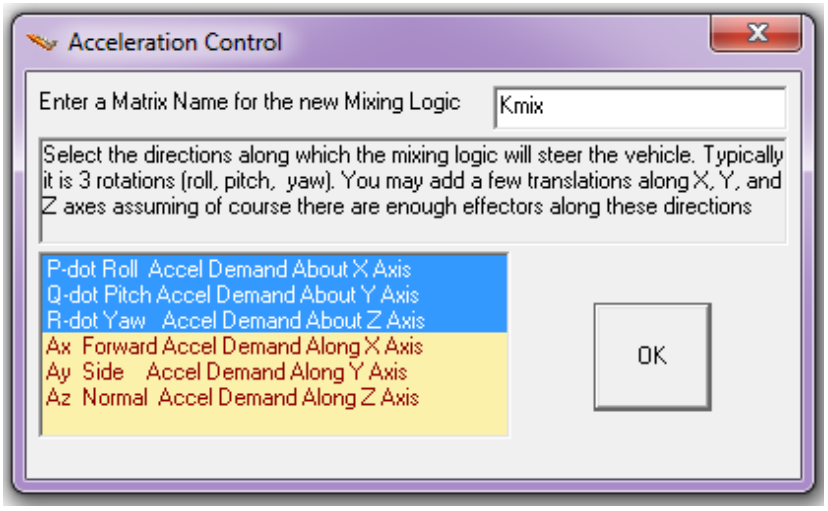




The program looks inside the input data file and searches for flight vehicle data sets. Then it presents a menu that shows the titles of all vehicle data-sets which are in this file. The vehicle data-sets are normally intended to create flight vehicle systems, but they are also used for generating mixing logic matrices. The user selects one of the vehicle sets for generating the mixing logic matrix in systems file "Rocket_Plane.Qdr", and clicks on "*Run Input Set*".

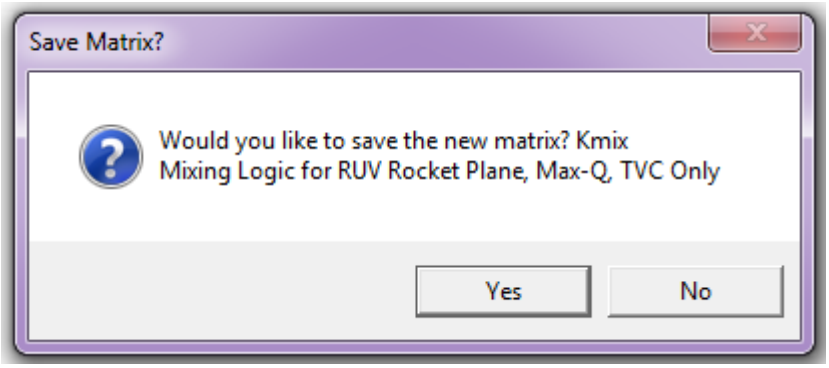


Use the next dialog to enter a name for the mixing logic matrix, in this example it is “Kmix”, and also the degrees of freedom or directions to be controlled by the effectors system. Typically 3 rotations are selected with some translations. The translational directions are optional and they should be chosen only if they can be directly accessible by the effectors. The number of vehicle degrees of freedom should be limited to the directions which are accessible by the effectors as defined in the vehicle model. We are normally interested in controlling 3 rotations: roll, pitch, and yaw. Translations are often indirectly controlled through pitching, rolling and yawing. Direct translational control may also be included if the vehicle has the effectors capability to provide control directly along those axes, such as a throttle control, speed brake, flaps, etc. For example, if the vehicle has throttle control or a speed-brake you may also include the x-axis acceleration to regulate speed. Flaps or RCS can also be used to control the z-axis acceleration. The control designer should know ahead of time which vehicle directions are controllable and select those directions in the degrees-of-freedom menu.



The program reads the vehicle mass properties, the CG location, engine locations, thrusts, maximum deflections, aero coefficients for the control surfaces, etc. and it calculates the mixing logic matrix using the pseudo-inverse method which was described in the previous section. The inputs to the mixing logic matrix come from the flight control system outputs, as shown in figure (4.1.1). The matrix outputs command the vehicle effectors. That is, the gimbaling engines, throttling engines, the reaction control jets, and the aero surfaces, as specified in the vehicle input data.

Finally, the program calculates the mixing logic matrix and wants to know if you wish to save the matrix in the selected systems file (.Qdr), and if the answer is "Yes" it saves it as a gain matrix. A gain matrix has a title and a short name. The number of columns in the mixing logic matrix corresponds to the acceleration directions (inputs) which are demanded from the FCS (min=3 and max=6). The number of rows is equal to the number of effectors (outputs) which are commanded by the matrix. The output sequence begins with the pitch deflections of engine numbers: 1, 2, 3,... n, followed by the yaw deflections of engine numbers: 1, 2, 3,...n, followed by the single gimbaling engine deflections along directions (γ_i), followed by the thrust variations of engines: 1, 2, 3,...n, and finally with the deflections of control surfaces: 1, 2, 3,... n. The definitions of the matrix inputs (control DOFs) and matrix outputs (effectors) are also included below the matrix. The labels can be edited and modified.



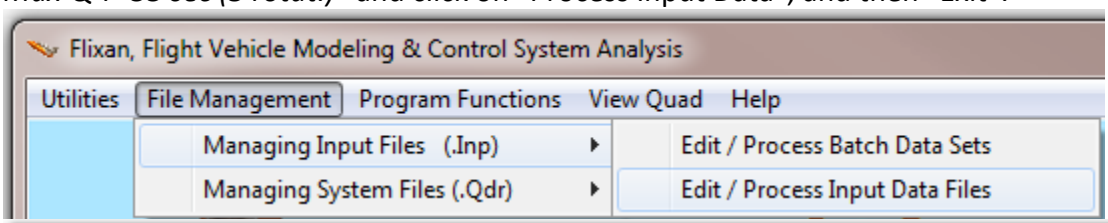
When a vehicle requires direct control in all 3 rotations and 3 translations the mixing matrix must have 6 columns. The 6 columns are the matrix inputs coming from the flight control acceleration demands. The first 3 correspond to the roll, pitch, and yaw angular acceleration demands, and the next 3 correspond to translational accelerations along x, y, and z. For a system with only three rotations and no translations the matrix has only three columns corresponding to the roll, pitch, and yaw demands. Fewer than 3 directions may also be selected in some cases. Selecting, for example, only the roll and yaw directions to be used in a truncated system that includes only lateral states. You may also choose roll and yaw rotations in combination with the y-axis acceleration. You may also select pitch rotation in combination with x-axis and z-axis accelerations to be used in a truncated pitch vehicle model.

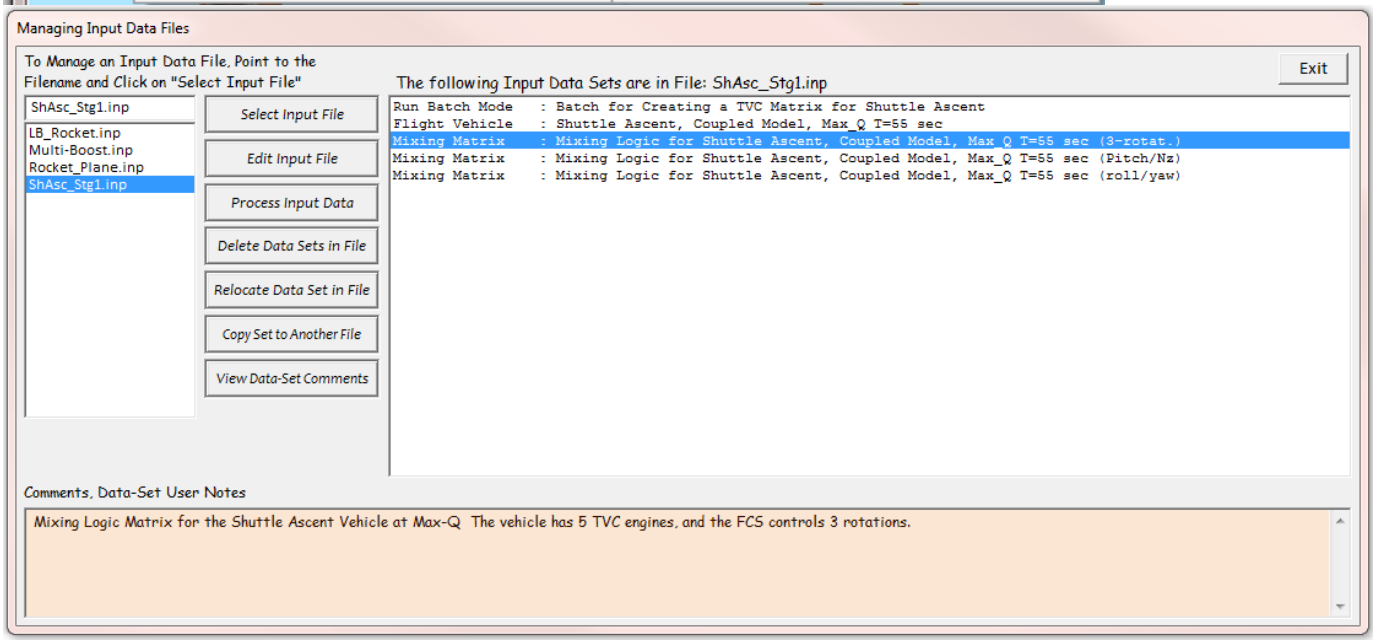
5.8 Mixing Logic Examples

We will now present some examples that illustrate how to apply this program to generate mixing logic matrices for different flight vehicle applications. They are included in directory "*C:\Flixan\Mixing Logic\Examples*". There is a Space-Shuttle ascent example, a Rocket Plane that combines aero-surfaces with TVC, a booster that uses single directional "constrained" gimbals, and a Shuttle with liquid boosters using multiple engines.

5.8.1 First Stage Shuttle Ascent Example

In the first example we have a Shuttle Ascent vehicle during first stage. The vehicle data are in file "ShAsc_Stg1.Inp" and its title is "*Shuttle Ascent, Coupled Model, Max-Q, T=55 sec*". This vehicle has 5 engines of different thrusts (three SSME's and two SRB's). The SSME's are mounted at some tilted angles with respect to the vehicle body axes. All engines can gimbal in the pitch and yaw directions but they are not throttling. We will use the mixing logic program to calculate three TVC matrices for different control directions and save them in file "ShAsc_Stg1.Qdr". The first TVC is a (10x3) matrix "*Mixing Logic for Shuttle Ascent, Coupled Model, Max-Q T=55 sec (3 rotat.)*" that converts the three (roll, pitch, and yaw) rotational acceleration demands to pitch and yaw gimbal deflections for the 5 engines. A mixing logic data-set for this configuration already exists in file "*ShAsc_Stg1.Inp*". To process it, go to "*File Management*", "*Manage Input Files*", and "*Edit/Process Input Data*". In the following dialog select the input file "*ShAsc_Stg1.Inp*" from the left menu and click on "Select Input File". From the right menu select the mixing logic set "*Mixing Logic for Shuttle Ascent, Coupled Model, Max-Q T=55 sec (3 rotat.)*" and click on "Process Input Data", and then "Exit".





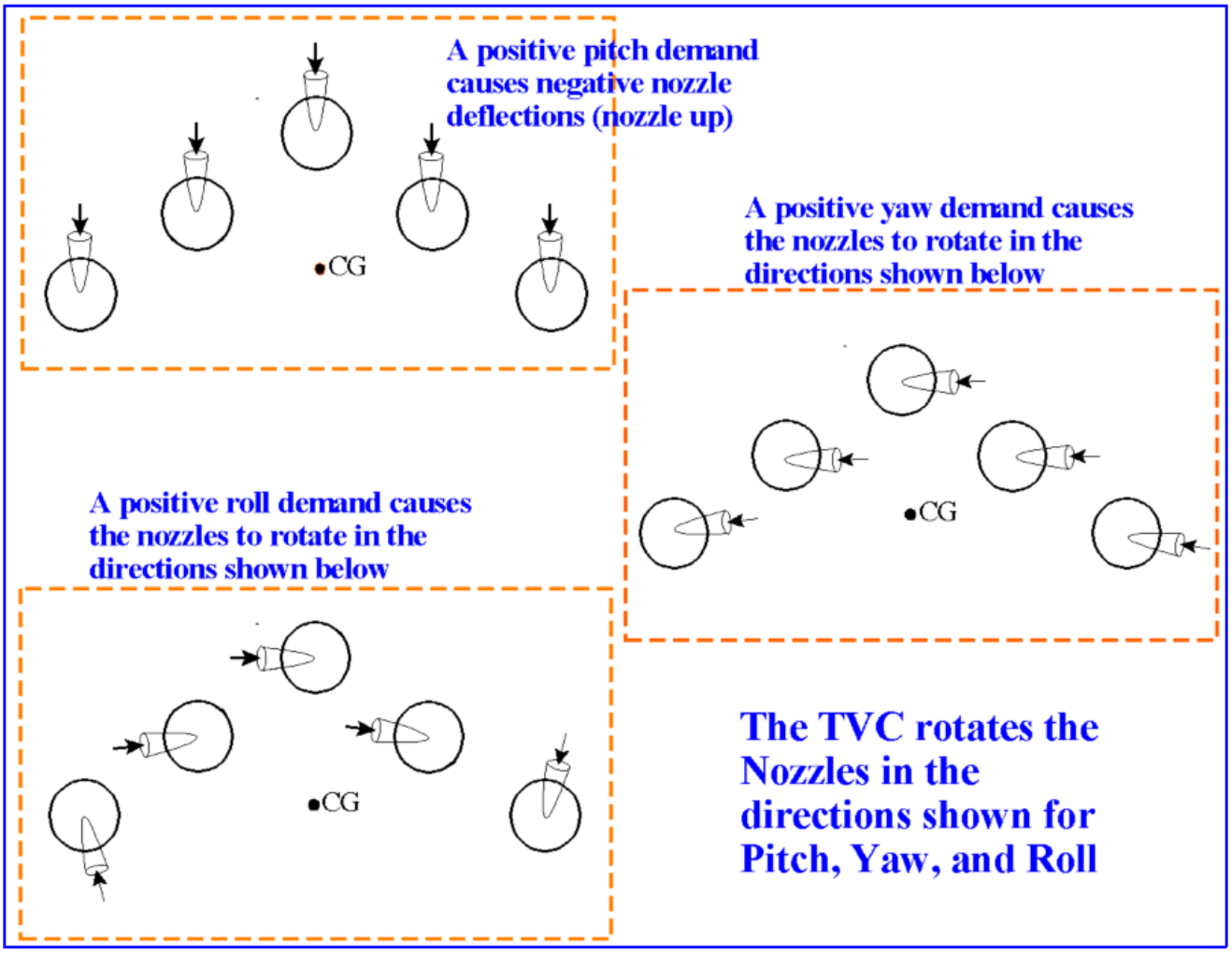
The following is the mixing-logic dataset in file "*ShAsc_Stg1.Inp*" that creates the Shuttle Ascent mixing logic matrix Kmix1 in the systems file "*ShAsc_Stg1.Qdr*", shown below the dataset. It uses the vehicle dataset "*Shuttle Ascent, Coupled Model, Max_Q T=55 sec*".

```
MIXING LOGIC MATRIX DATA ..........    (Matrix Title, Name, Vehicle Title, Control Directions)
Mixing Logic for Shuttle Ascent, Coupled Model, Max_Q T=55 sec (3-rotat.)
! Mixing Logic Matrix for the Shuttle Ascent Vehicle at Max-Q
! The vehicle has 5 TVC engines, and the FCS controls 3 rotations.
!
Kmix1
Shuttle Ascent, Coupled Model, Max_Q T=55 sec
P-dot Roll  Acceleration About X Axis
Q-dot Pitch Acceleration About Y Axis
R-dot Yaw   Acceleration About Z Axis
-----------------------------------------------------------------------------------------------


Gain Matrix for ...
Mixing Logic for Shuttle Ascent, Coupled Model, Max_Q T=55 sec (3-rotat.)
! Mixing Logic Matrix for the Shuttle Ascent Vehicle at Max-Q. The vehicle has 5 TVC engines,
! and the FCS controls 3 rotations.
Matrix Kmix1                Size = 10 X  3
           1-Roll              2-Pitch             3-Yaw
   1-Row  0.000000000000E+00 -0.332385989120E+00  0.000000000000E+00
   2-Row  0.419778734424E-01 -0.343653931130E+00 -0.800718091209E-02
   3-Row -0.419778734424E-01 -0.343653931130E+00  0.800718091209E-02
   4-Row  0.250744834033E+00 -0.556397259061E+00 -0.451939299076E-01
   5-Row -0.250744834033E+00 -0.556397259061E+00  0.451939299076E-01
   6-Row  0.285424016251E+00  0.000000000000E+00 -0.410883120272E+00
   7-Row  0.204455664212E+00  0.000000000000E+00 -0.407521945585E+00
   8-Row  0.204455664212E+00  0.000000000000E+00 -0.407521945585E+00
   9-Row -0.145540006956E-01  0.000000000000E+00 -0.594135868134E+00
  10-Row -0.145540006956E-01  0.000000000000E+00 -0.594135868134E+00
-----------------------------------------------------------------------------------------------
Definitions of Matrix Inputs (Columns):    3
P-dot Roll  Accel Demand About X Axis
Q-dot Pitch Accel Demand About Y Axis
R-dot Yaw   Accel Demand About Z Axis

Definitions of Matrix Outputs (Rows):     10
Output:  1 Dy(engine):  1 Pitch Deflection
Output:  2 Dy(engine):  2 Pitch Deflection
Output:  3 Dy(engine):  3 Pitch Deflection
Output:  4 Dy(engine):  4 Pitch Deflection
Output:  5 Dy(engine):  5 Pitch Deflection
Output:  6 Dz(engine):  1 Yaw  Deflection
Output:  7 Dz(engine):  2 Yaw  Deflection
Output:  8 Dz(engine):  3 Yaw  Deflection
Output:  9 Dz(engine):  4 Yaw  Deflection
Output: 10 Dz(engine):  5 Yaw  Deflection
-----------------------------------------------------------------------------------------------
```

The three columns in matrix Kmix1 correspond to the roll, pitch, and yaw vehicle acceleration demands. The first 5 rows generate the pitch engine deflection commands that drive the actuators, and the last 5 rows generate the deflection commands for the yaw actuators. A positive pitch acceleration demand (middle column) causes all engines to deflect in the negative pitch direction that causes the vehicle nose up. The yaw TVC deflections are zero. A yaw acceleration demand (right column) causes the engines to deflect mostly in the negative yaw direction. It causes, however, a small amount of differential pitch deflections. This is for counteracting the roll accelerations induced by the yaw gimbaling. A roll acceleration demand causes the engines on the left side of the vehicle to deflect in the positive pitch direction, and the engines on the right side of the vehicle to deflect in the negative pitch direction to create a positive roll. It also causes the top engine to deflect in yaw.



The second TVC matrix in file "ShAsc_Stg1.Qdr"is a (10x2) matrix "*Mixing Logic for Shuttle Ascent, Coupled Model, Max_Q T=55 sec (Pitch/Nz)*". The two input directions are pitch acceleration and Nz acceleration demands. The outputs are the same. The third TVC matrix is a (10x2) that corresponds only to roll and yaw acceleration demands. Its title is "*Mixing Logic for Shuttle Ascent, Coupled Model, Max_Q T=55 sec (roll/yaw)*".

Shuttle TVC Simulation in Matlab

We may now illustrate the operation of this TVC matrix and demonstrate its capability to decouple the 3 flight control loops and to achieve the FCS accelerations demanded by creating a simple open-loop simulation. This open-loop decoupling and acceleration control is based on the rigid-body dynamics, geometry and effector parameters. It simplifies the control design because the system becomes diagonally dominant. It does not take into consideration, however, the aerodynamic forces produced due to the variations in the aerodynamic angles, but even in the presence of aero this method still achieves significant amount of open-loop decoupling and acceleration control for short periods before it diverges, as we shall see in this example. In any case we do not intend to stabilize the vehicle open-loop, but by providing a significant amount of control decoupling, open-loop, then the control design of the individual loops becomes more efficient.

In this example we will simulate the open-loop acceleration response in roll, pitch and yaw of the Shuttle vehicle with the TVC matrix connected in series, as shown in Figure (5.6). The yellow block on the left contains the TVC matrix Kmix1, and the vehicle is the green block on the right. The vehicle outputs are accelerations. The inputs to the TVC matrix are roll, pitch and yaw acceleration demands. The TVC matrix is designed to control the acceleration by properly deflecting the 5 engine nozzles, and counteracting the inherent coupling due to the cross-products of inertia and the CG offset. It does not prevent the coupling due to α and β. A Max-Q is probably the worst case to choose for demonstrating the mixing logic effect because the aerodynamic moments dominate. A Low-Q would have been a better choice. However, we are still able to demonstrate the decoupling function of the TVC by running this open-loop simulation for a short period of time before it begins to diverge due to the fact that the vehicle is open-loop unstable.

This model "*Open_Loop.Mdl*" is located in "*Flixan\Mixing Logic\ Examples\Mat*". The green block on the right contains the modified Shuttle vehicle state-space system "*Shuttle Ascent, Coupled Model, Max_Q T=55 sec (acceler out)*" which has rotational acceleration outputs $(\dot{p},\dot{q},\dot{r})$. It is loaded from file "Shuttle_accel.m". The systems and matrices were created from file "ShAsc_Stg1.Inp", saved in the systems file "ShAsc_Stg1.Qdr", and are loaded into Matlab workspace by using the script file "run.m".

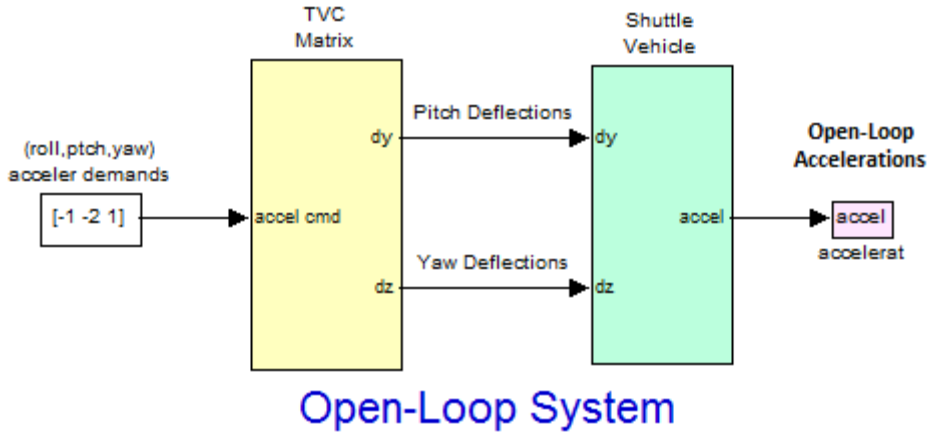


Figure (5.6) Simulink Model "Open_Loop.mdl" used for Open-Loop Analysis

Figure (5.6b) shows the open-loop system acceleration response to an arbitrary acceleration demand (-1, -2, +1) in roll, pitch, and yaw respectively in (deg/sec^2). The output accelerations are equal to the commanded accelerations. We run the simulation for a short time to avoid diverging because it is unstable.

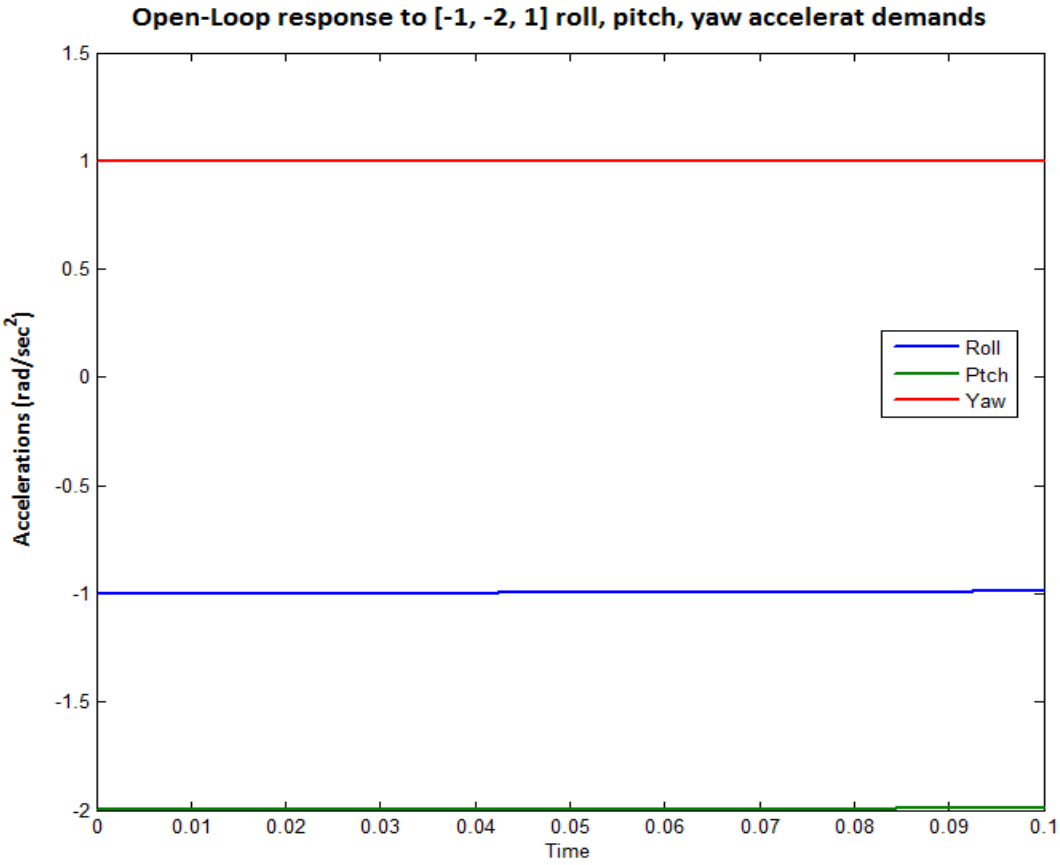


Figure 5.6b Response of the Open-Loop System to Roll, Pitch, and yaw Acceleration Demands

The next step is to use the decoupled plant in order to design the feedback control system by shaping the open-loop frequency response as needed to achieve the desired stability and performance characteristics. By including the TVC matrix pre-multiplying the plant dynamics, as shown, it makes a good synthesis model for LQR or H-infinity control design. This approach creates more efficient designs because the plant is more rounded with all loops having approximately the same bandwidth.

Figure 5.7 shows the Shuttle vehicle with the loop closed via a PD feedback control system using attitude and rate measurements. It is commanded to perform an attitude change maneuver by simultaneously commanding attitude steps in roll, pitch, and yaw. Figure (5.8) shows the closed-loop system response to the attitude commands in (deg). We run the simulation for 12 seconds to show that the closed-loop system is stable and that it responds to the commanded attitude. The file "Pl.m" is used to plot the data.

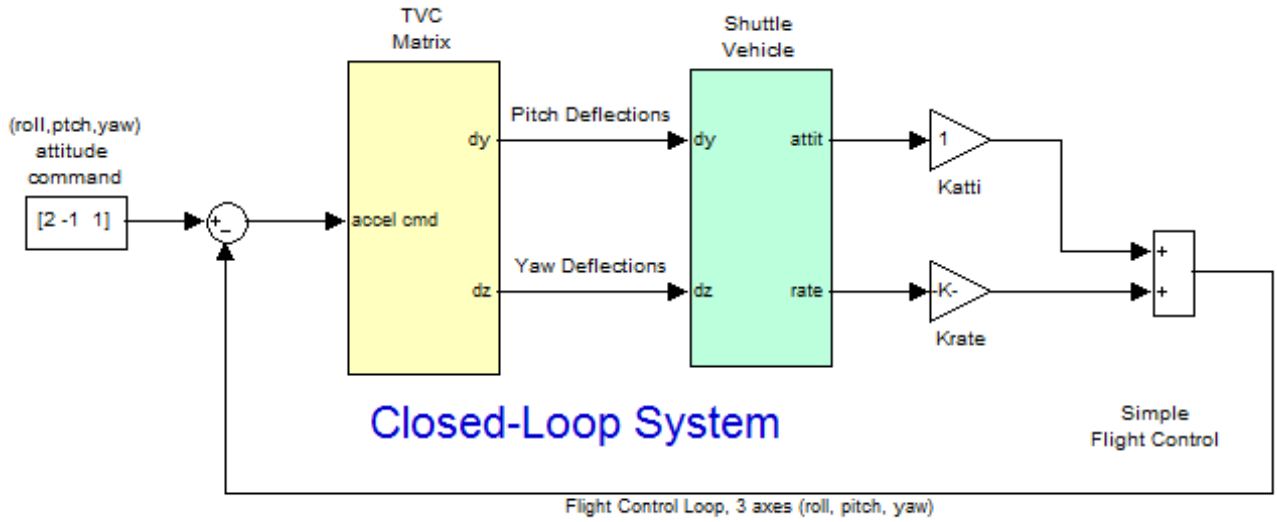


Figure 5.7 Closed-Loop Simulation Model "TVC_Sim.Mdl"

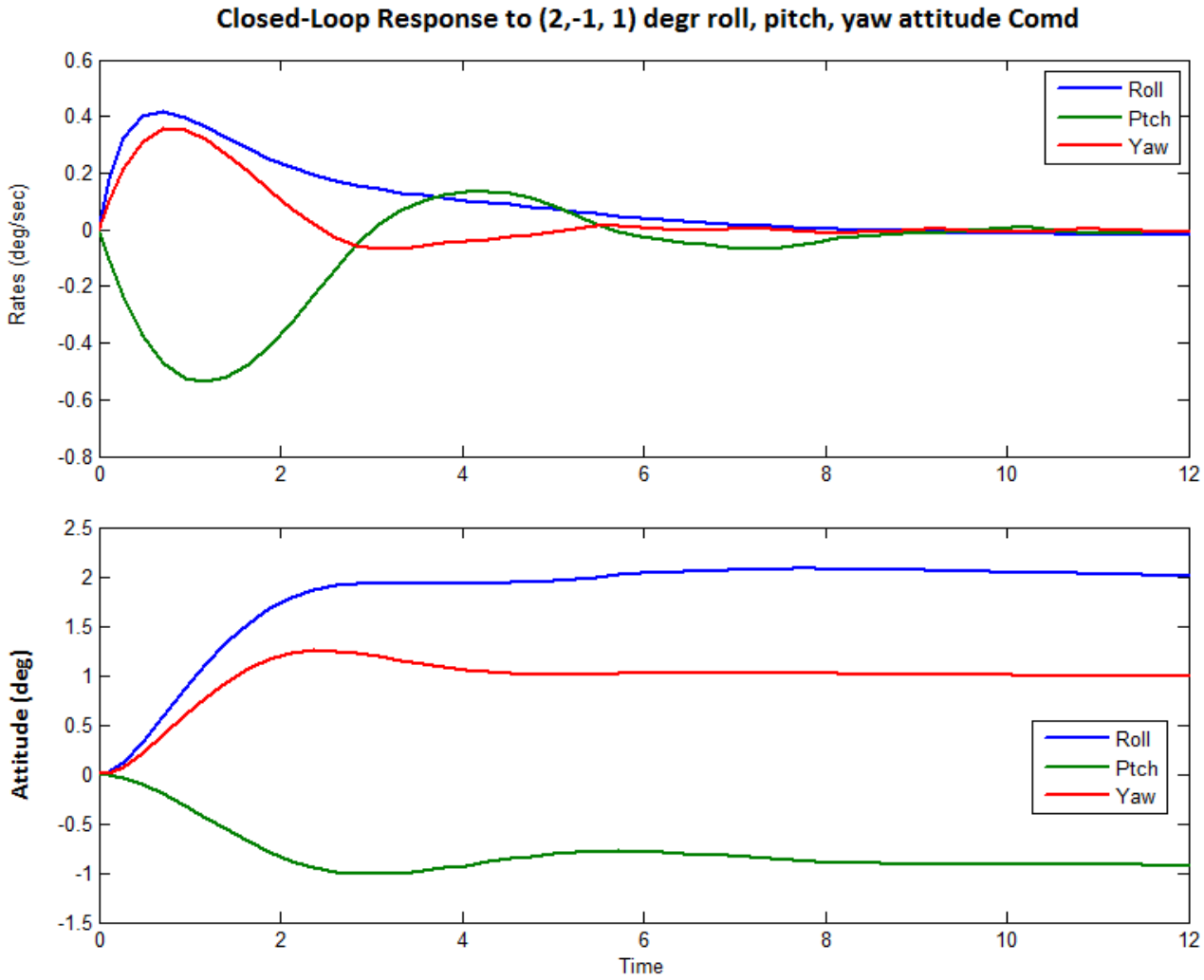


Figure (5.8) Closed-Loop System response with the TVC Matrix in the loop, System achieves a coordinated step response in all 3 directions

5.8.5 Vehicle with Multiple types of Effectors

We will now demonstrate a vehicle example that uses all types of effectors: TVC, throttling, reaction control jets, and aerosurfaces. It is a rocket-plane shown in Figure 5.10. It has two main engines of 60,000 (lb) of thrust each, that gimbal in pitch and yaw and they can also throttle ±30% from nominal thrust. It has 5 aero-surfaces: two inboard and two outboard elevons, and a vertical rudder. It also has two sets of analogue RCS thrusters, one pair thrusting in the ±Z direction and the second pair is thrusting in the ±Y direction. The RCS thrusts are proportional to the throttle commands that vary between 0 and ±1 producing a maximum thrust of ±3,000 (lb). This vehicle obviously has enough effectors to be controlled in all 6 directions, 3 rotational and 3 translational. To design, however, the effector combination logic by inspection is not easy. We shall, therefore, use the Mixing Logic calculation program.

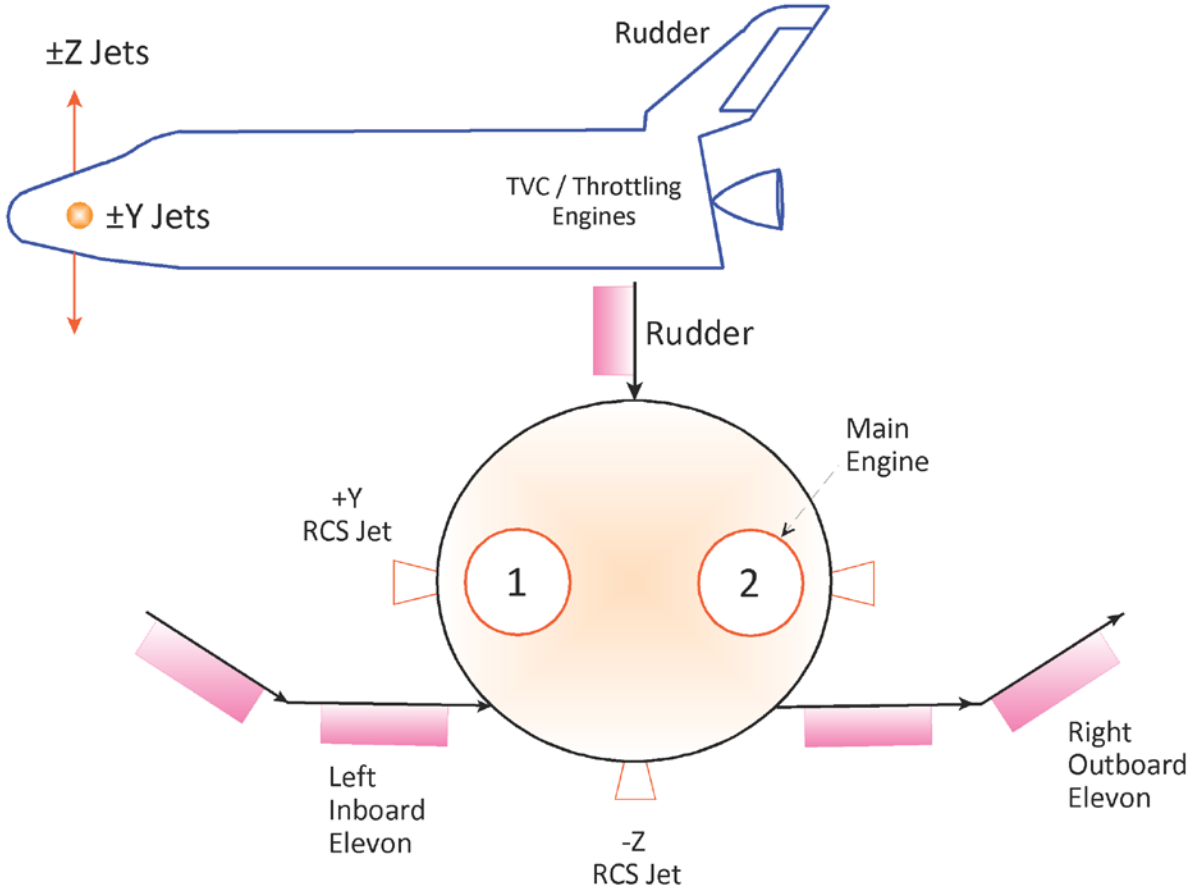


Figure 5.10 A Flight Vehicle controlled with multiple types of effectors

The input data for this example is in file "*Shuttle_MixLogic.Inp*" located in folder "*Flixan/ Mixing Logic/ Examples/Shuttle Mixing Logic Example*". The title of the vehicle model described is "*Shuttle Early Hypersonic Re-Entry, Rigid Body Axes Model*". There is also a mixing logic dataset in this file that uses the vehicle model to generate a (13x6) mixing-logic matrix that will convert the 3 rotational and 3 translational acceleration demands to 13 effector commands. The name of the matrix is Kmix and its title is "*Mixing Logic for the 5 Aero-Surfaces, TVC, and RCS*".

```
-------------------------------------------------------------------------------------------------
MIXING LOGIC MATRIX DATA .........    (Matrix Title, Name, Vehicle Title, Control Directions)
Mixing Logic for the 5 Aero-Surfaces, TVC, and RCS
! Mixing Logic Matrix for combining the five Space Shuttle control surfaces (inboard and
! outboard elevons and rudder) excluding the RCS jets to achieve 3 rotational accelerations
!
Kmix
Shuttle Early Hypersonic Re-Entry, Rigid Body Axes Model
P-dot Roll  Acceleration About X Axis
Q-dot Pitch Acceleration About Y Axis
R-dot Yaw   Acceleration About Z Axis
Ax  Axial   Acceleration Along X Axis
Ay  Lateral Acceleration Along Y Axis
Az  Normal  Acceleration Along Z Axis
-------------------------------------------------------------------------------------------------
```

Figure 5.11 shows the mixing-logic matrix Kmix that combines the 13 effectors to produce the accelerations demanded in all six directions. The inputs to the mixing matrix are the six demands, 3 rotational and 3 translational that would normally come from the flight control system. The 13 matrix outputs go to the vehicle effectors. The vehicle has 4 engine TVC deflections (2 pitch and 2 yaw), 2 engine throttle commands, 2 RCS throttle commands, and 5 aerosurface deflection commands in (radians). The throttle inputs in the vehicle model are scaled and they must vary between 0 and ±1. Note, the throttle input is not thrust but it represents the fraction of thrust variation above or below nominal and its magnitude should not exceed 1. The value of the actual thrust is already included in the dynamic model.

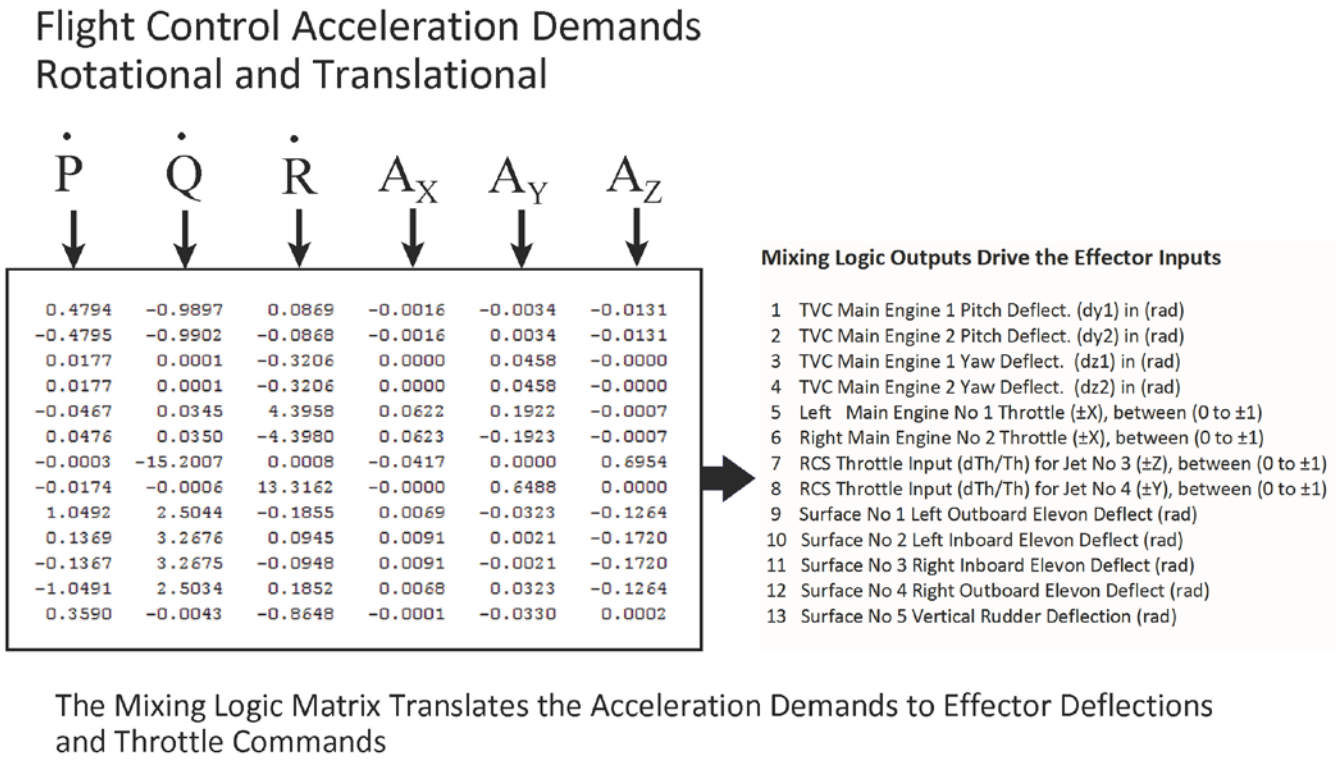


Figure 5.11 Mixing Logic Matrix Inputs and Outputs

In order to test the effectiveness of the mixing logic matrix capability to achieve the demanded input accelerations we must create an open-loop simulation model that combines the vehicle state-space model in series with the mixing matrix, similar to Figure 5.2. This is implemented in the Simulink model "Mix_Logic_Sim.Mdl" shown in Figure 5.12 for testing the mixing matrix. The input to Kmix is a

vector of 6 acceleration demands, 3-rotational and 3-translational as already described, hypothetically coming from the FCS, and the mixing logic matrix transforms the demands to 13 effector commands that become inputs to the vehicle dynamic model. The outputs of the vehicle model are modified to produce vehicle accelerations, 3-rotational and 3-translational. If the mixing-logic matrix is properly designed, the accelerations generated by the dynamic model will be equal to the step accelerations demanded by the FCS. We run the simulation for a short period of time because the vehicle is open-loop unstable and it will eventually diverge. A short time is sufficient to measure the accelerations produced. The simulation shows that the plant in series with the mixing-logic matrix Kmix is perfectly diagonalized. From the accelerations view point it behaves like a 6x6 identity matrix. Any combination of acceleration demands produces identical vehicle accelerations, as shown in the simulation results.

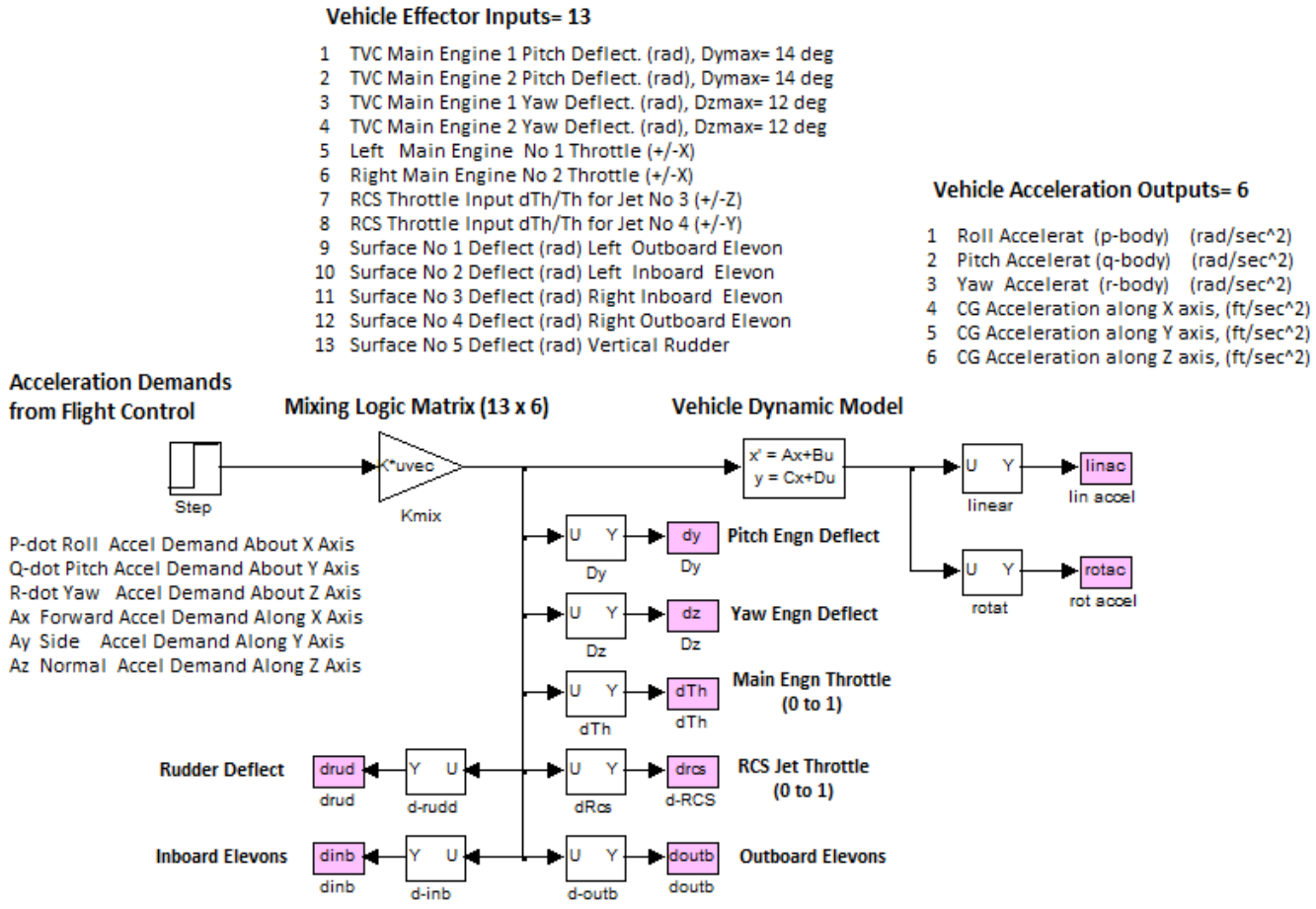
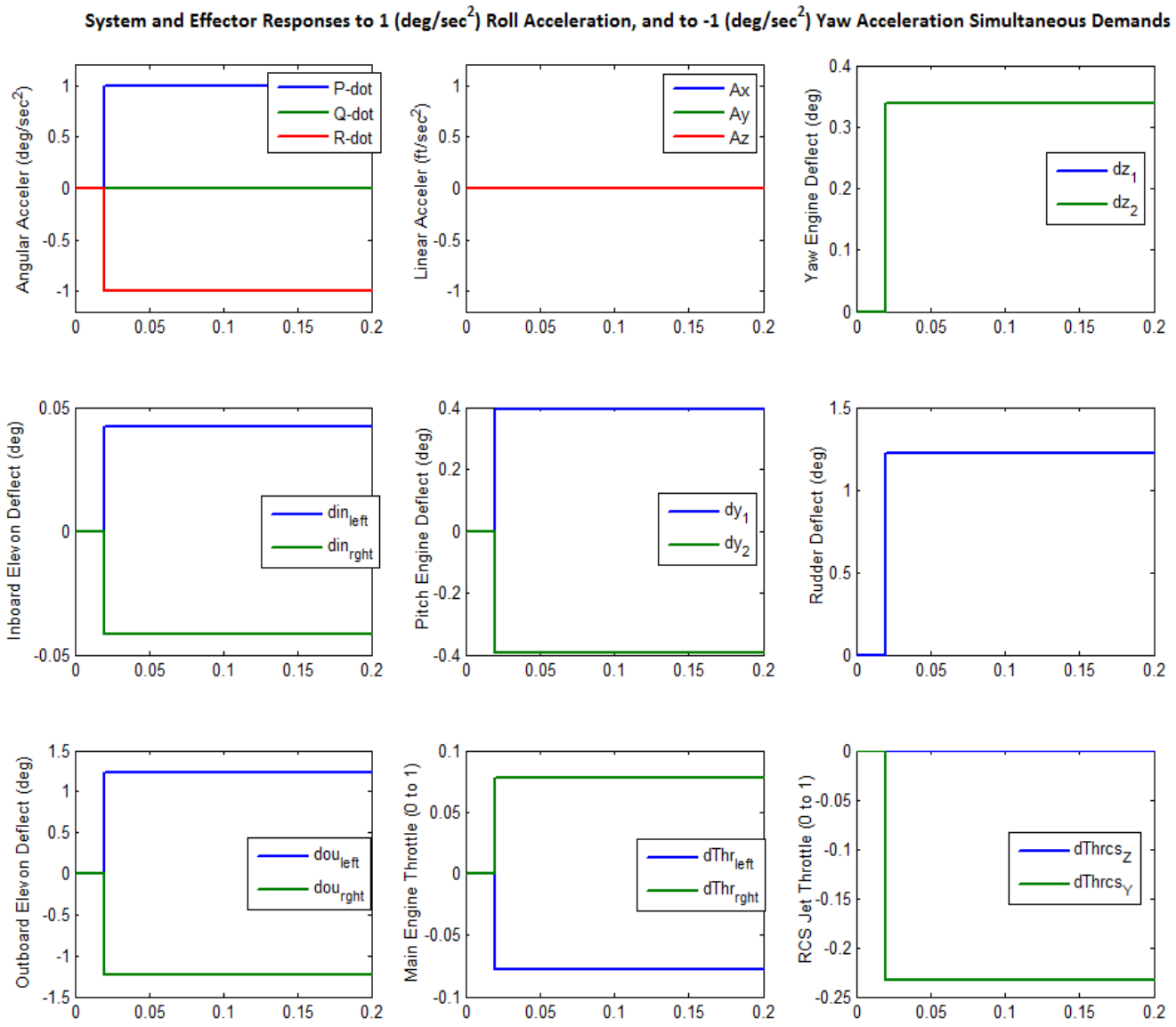


Figure 5.12 Open-Loop Simulation model "Mix_Logic_Sim.mdl" used to test the effectiveness of the Mixing Logic Matrix.



In Figure 5.13 the vehicle model is commanded open-loop to accelerate in +roll and -yaw simultaneously. The translational acceleration demands are zero. The results show that all open-loop accelerations produced are equal to the demanded accelerations. The two yaw TVC gimbals (δz) rotate in the +yaw direction, and also the rudder rotates positive to generate the required negative yaw acceleration. The negative yaw acceleration is also assisted by the throttling yaw RCS jets that produce a force in the –Y direction. The –yaw acceleration is also slightly assisted by the differential throttling of the two main engines (left engine throttles down, right engine throttles up). The +roll acceleration is produced by differentially deflecting the main engine pitch gimbals (δy), the inboard elevons, and the outboard elevons (left side down, right side up). The outboard elevons deflect more than the inboard elevons, because they are obviously more effective in roll.

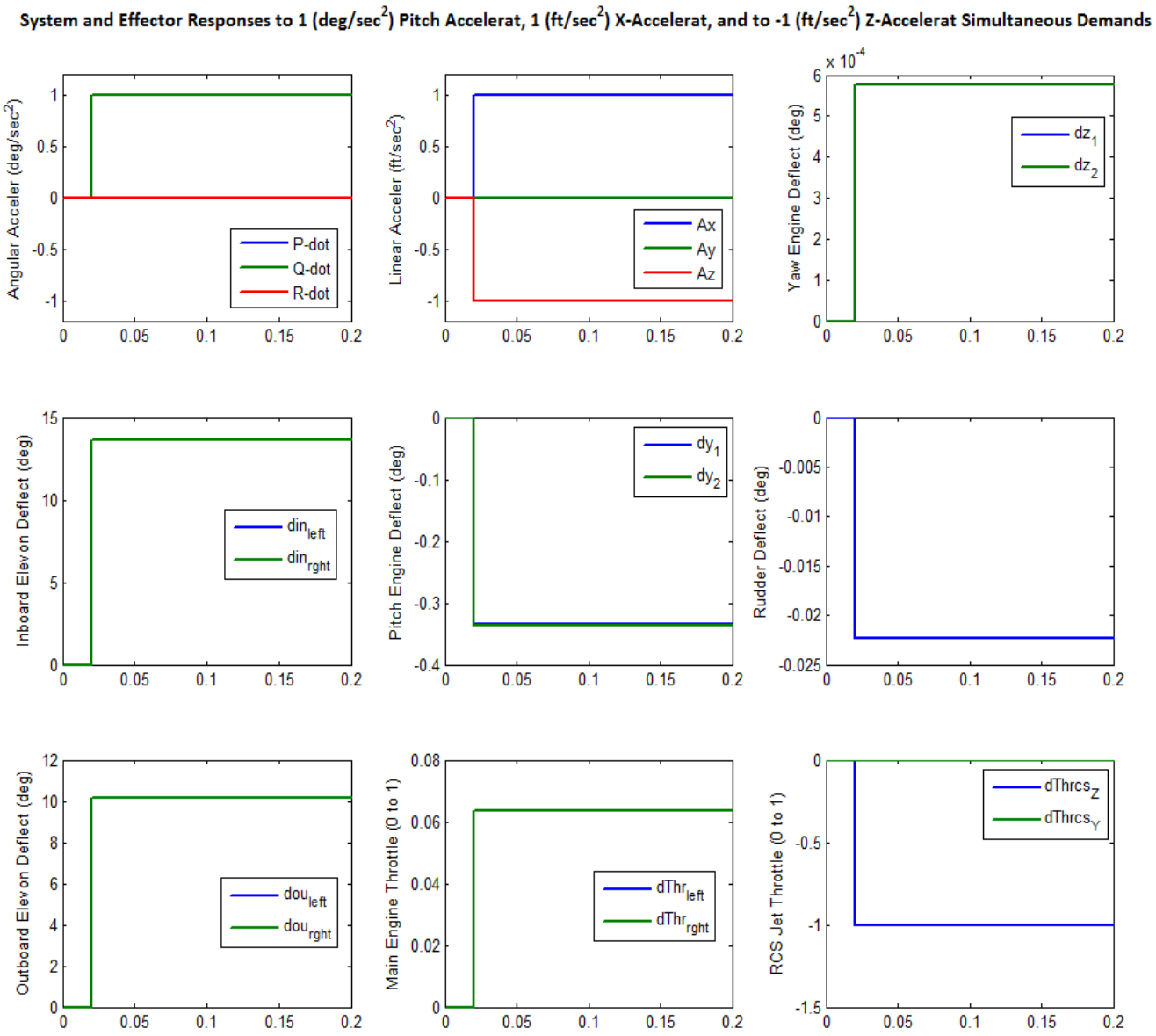


Figure 5.14 shows the open-loop system and effector response to one rotational and two translational step acceleration demands which are applied simultaneously. That is: 1 (deg/sec^2) pitch acceleration demand, 1 (ft/sec^2) axial acceleration demand, and -1 (ft/sec^2) normal acceleration demand.

- All accelerations produced are equal to the demanded accelerations, the pitch acceleration and the translational accelerations, as shown above.
- The negative normal acceleration demand -Az causes all four elevons to deflect symmetrically in the positive direction (down). The inboard elevons are more effective and they are deflecting further than the outboard.
- The RCS jet is throttling heavily in the –Z direction assisting in the –Az acceleration.
- The increase in the axial acceleration +Ax is produced by the positive throttling of the two main engines.
- The +pitch acceleration is produced by symmetrically deflecting the two TVC engines in the negative pitch direction (-δy).
- The forward firing RCS thruster in the -Z direction is also helping in pitch acceleration.