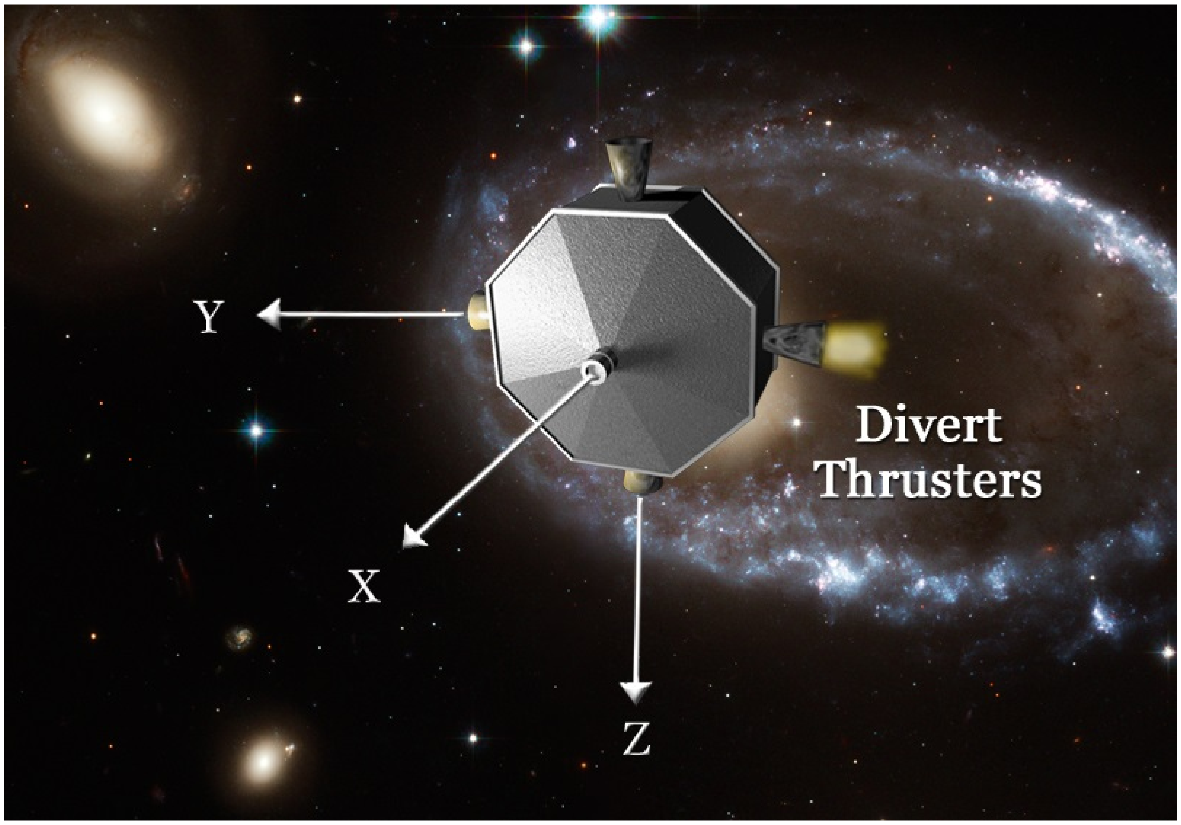
Design of a Kinetic Space Interceptor



Introduction

In this example we will analyze a guided intercept between two space vehicles: an interceptor which is a kinetic vehicle (KV) and a target that may be a meteorite heading towards the earth, space debris, or an enemy missile. A typical space intercept is divided into three phases:

a) The boost phase where the KV is attached to an accelerating booster that points its direction towards the target and acquires sufficient kinetic energy to destroy the target by impact.
b) The Mid-Course guidance phase that attempts to align the direction of the KV in a collision course with the target by predicting the target's future trajectory.
c) In this example will analyze the final phase guidance, the End-Game which begins when the interceptor is sufficiently close to acquire the target position in its field of view and it ends when the KV impacts and destroys the target.

The end-game is a dynamic engagement using closed-loop guidance to improve impact precision and probability, especially when the target is randomly accelerating in order to avoid getting hit. The KV is already in a collision course with the target and it is no longer accelerating. The target, however, may be accelerating towards and perpendicular to the KV direction. The KV uses an optical seeker to track the target and its line-of-sight (LOS) is always pointing towards the target. It has an Attitude Control System (ACS) to maneuver its attitude, tracking the target at the center of the seeker's field of view and aligning the x-axis with the target, as shown in Figure 1. If the target is not maneuvering and if the mid-course boost was executed perfectly, the target would remain in the center of the seeker's field of view all the way to impact. The KV has four 5 (lb) thrusters for attitude control located in the back. It also has four 20 (lb) divert thrusters for translation control which are firing through the CG to prevent rotations. Since the KV is already in a collision course with the target and it is no longer accelerating towards the target, its translational motion is controlled only in two directions perpendicular to the LOS, as it attempts to take out small translational errors that have accumulated due to target unexpected acceleration and noise. If the seeker detects an error or the target moves perpendicular to the LOS, the guidance will fire divert thrusters to produce acceleration that will zero the error in the corresponding direction. It is assumed that the approximate relative position, velocity, and acceleration of the target are calculated from the seeker azimuth and elevation measurements, and from the target distance which is estimated from navigation and used in the End-Game algorithm. Figure 1 shows the locations of the jets, the input numbers in the dynamic model and the corresponding node numbers in the structural model.

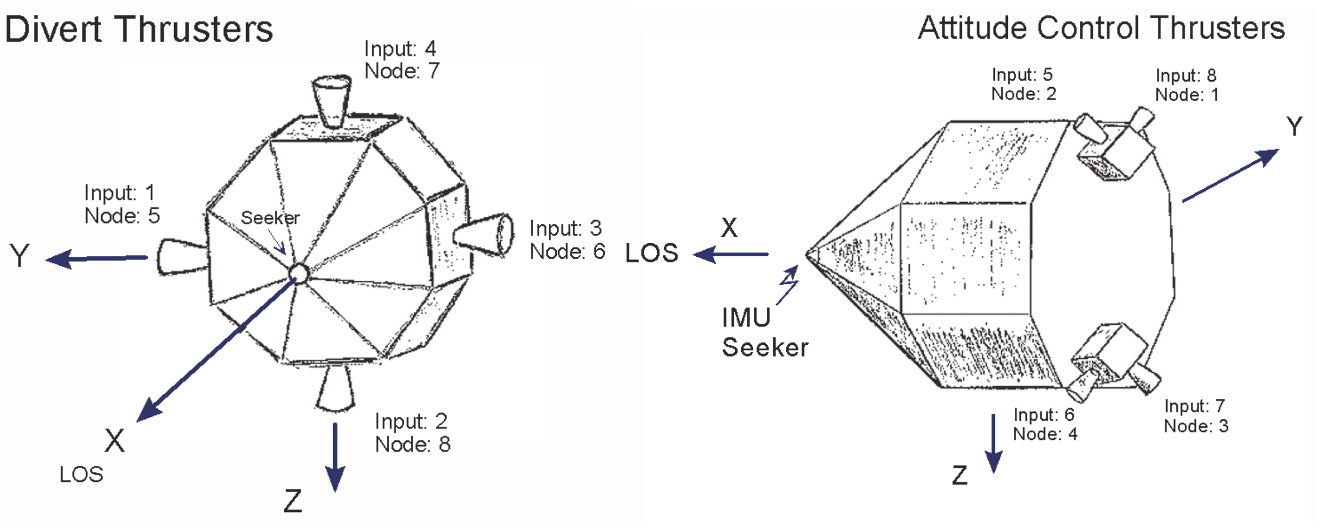


Figure 1 KV Spacecraft and the Location of RCS Thrusters

1. Dynamic Models

In this Section we will develop several dynamic models that describe the rotational and translational motion of the KV spacecraft in response to divert and attitude control thrusters. The spacecraft models include structural flexibility derived from a finite elements model. They are implemented using Flixan programs and will be used to design the control laws, simulate, and analyze the system stability. We will use the spacecraft models to design an attitude control system that uses the ACS thrusters to point the spacecraft LOS towards the target and tracks it by rotating the KV. We will also design a linear acceleration control system that controls the spacecraft acceleration by pulsing the divert thrusters.

An additional dynamic model will be presented in Section 4 for designing an optimal control law that will guide the KV to the target. It describes the relative translational motion of the two spacecraft in a direction (y or z) perpendicular to the LOS. It is initialized from the state of the target relative to the KV when the seeker acquires the target. It is used by Flixan to derive the state-feedback gains and the Kalman-Filter, and also in Matlab simulations. The azimuth and elevation LOS errors of the target relative to the KV are measured by the seeker from its initial local inertial attitude, when the target is first acquired. They are translated into relative target position (S_{ry}, S_{rz}) perpendicular to the LOS, which command the translational control system to fire the corresponding divert thrusters to zero the relative position at the estimated impact time.

Two guidance laws will be analyzed using this model: Proportional Navigation (PN), and Optimal Control (OC). Estimates of relative position, velocity and acceleration are used to calculate the KV acceleration commands in order to take out the position errors perpendicular to the LOS at impact. The continuous acceleration commands calculated from guidance are converted into jet pulses that produce the demanded average acceleration. The two algorithms will finally be combined together for the purpose of taking advantage of their properties in different flight conditions and designing a system that is fuel efficient and robust to range and to uncertainties of target motion.

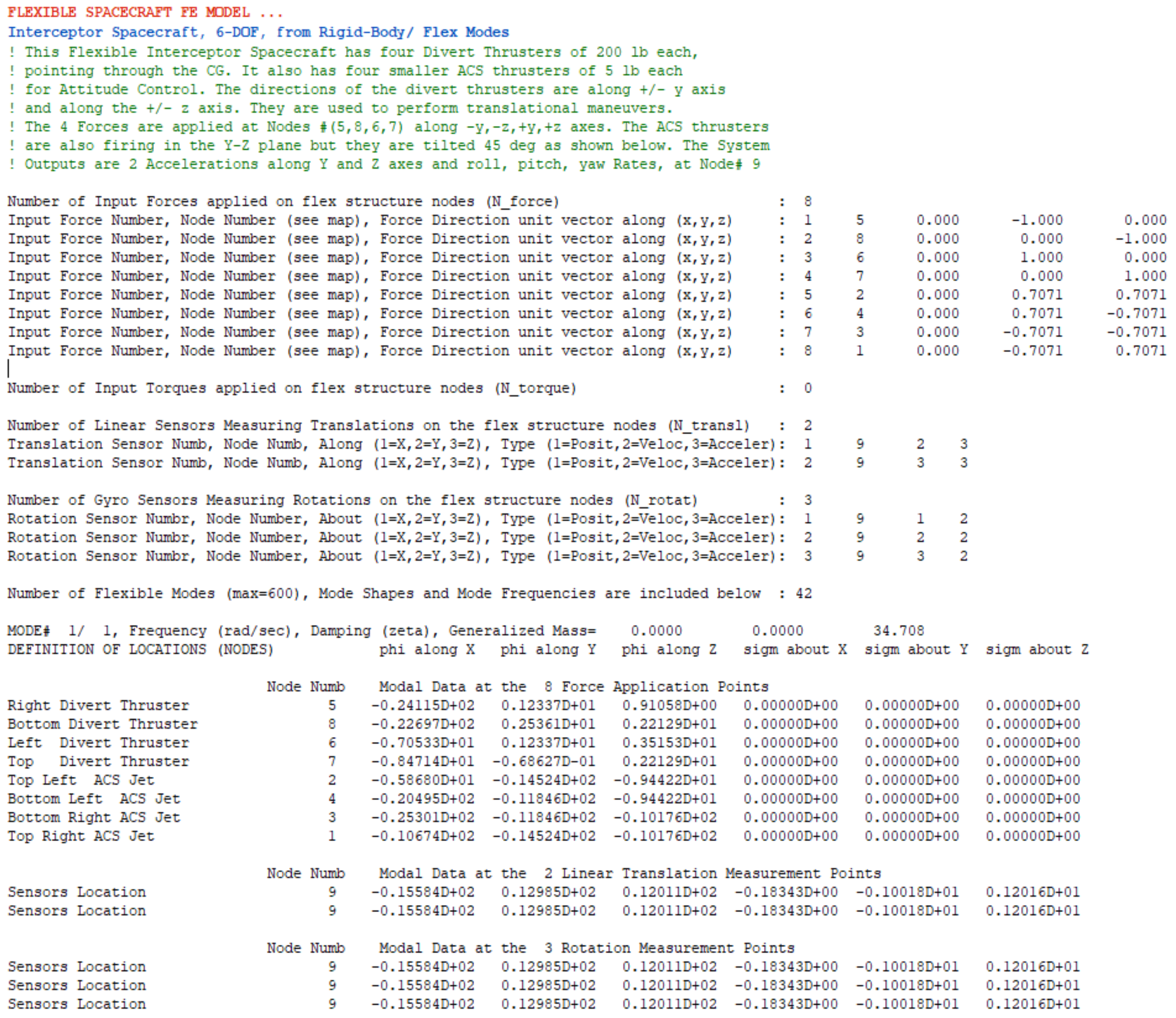
The data files for this example are in the directory *"Flixan\Examples\Interceptor Spacecraft"*. The spacecraft model was derived from a Nastran finite elements model. The modal data were extracted and saved in file *"Interceptor.Mod"* that contains a total of 140 structural modes at 19 vehicle locations. The first six are rigid body modes used for modeling the 3 rigid translations and 3 rotations of the spacecraft. The nodes file: *"Interceptor.Nod"* includes a list of the nodes and it is used as a look-up table by the mode selection program to identify structural nodes in the modal data file that correspond to spacecraft locations. The maneuvering spacecraft does not have any rotating solar arrays or gimbaling payloads. Therefore, it does not require a coupling coefficients data file *(*.hpr)*.

We used Flixan to create two dynamic models for this spacecraft that include structural flexibility. The system “*rigbod_mod.m*” was created using the flexible spacecraft modeling program, and the system “*kkv_acs.m*” that was generated from the flight vehicle modeling program. They both use 8 thrust varying inputs: 4 divert thrusters to perform translation maneuvers along the y and z axes respectively and 4 ACS thrusters for 3 axes attitude control. The outputs consist of: two accelerometers along the y and z axes, and 3 rate gyros which measure body rates in roll, pitch, and yaw axes.

1.1 Flexible Spacecraft Model

The input dataset for generating the flexible spacecraft system is already prepared and it is saved in file *“Interceptor.inp”*. It is processed by the Flixan *“Flexible Spacecraft Modeling”* program to create the spacecraft state-space system from selected Nastran modes. Its title is *“Interceptor Spacecraft, 6-DOF, from Rigid-Body/ Flex Modes”*. This dataset consists of three parts. The first part defines the input forces, node numbers, and force directions. The second part describes the types and location of the sensors which are: 2 accelerometers along y and z axes, and 3 rate gyros measuring roll, pitch and yaw. The third part of the dataset consists of selected structural modes to be included in the model. 42 modes were selected in this example, including the first 6 rigid-body modes. The modes were selected from the modal data file by a mode selection process that will be described in Section 1.2.

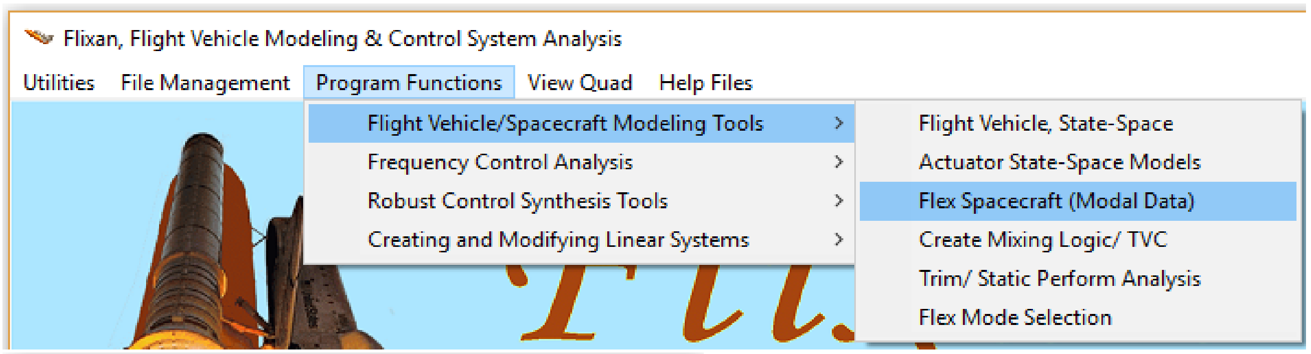
For translation control in the ±y and ±z KV directions the spacecraft uses four divert thrusters of 20 (lb) each that fire through the GC, inputs: 1 to 4, see Figure 1a. For attitude control the system uses four 5 (lb) reaction control jets (inputs: 5 to 8). They are tilted at 45 degrees with respect to the y and z axes to provide attitude control in all 3 directions, see Figure 1b. The Nastran model is already scaled to include the thruster forces, so the jet forces must not exceed ±1. There are no torque actuators in this spacecraft. The spacecraft state-space model is produced by processing the input dataset using the flexible spacecraft modeling program. Its title is also “*Interceptor Spacecraft, 6-DOF, from Rigid-Body/ Flex Modes*”. Its outputs include the two accelerometers in the y and z directions, and the 3 rate-gyros measuring roll, pitch, and yaw rates. It is saved in file “*Interceptor.Qdr*” and it will be used in the upcoming simulations.

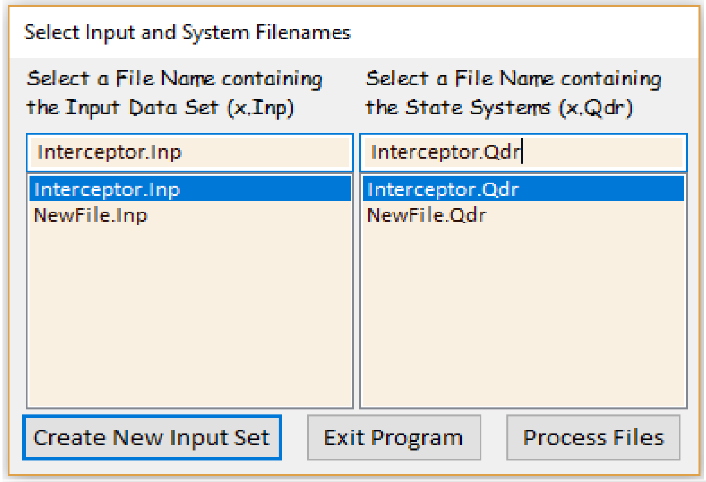


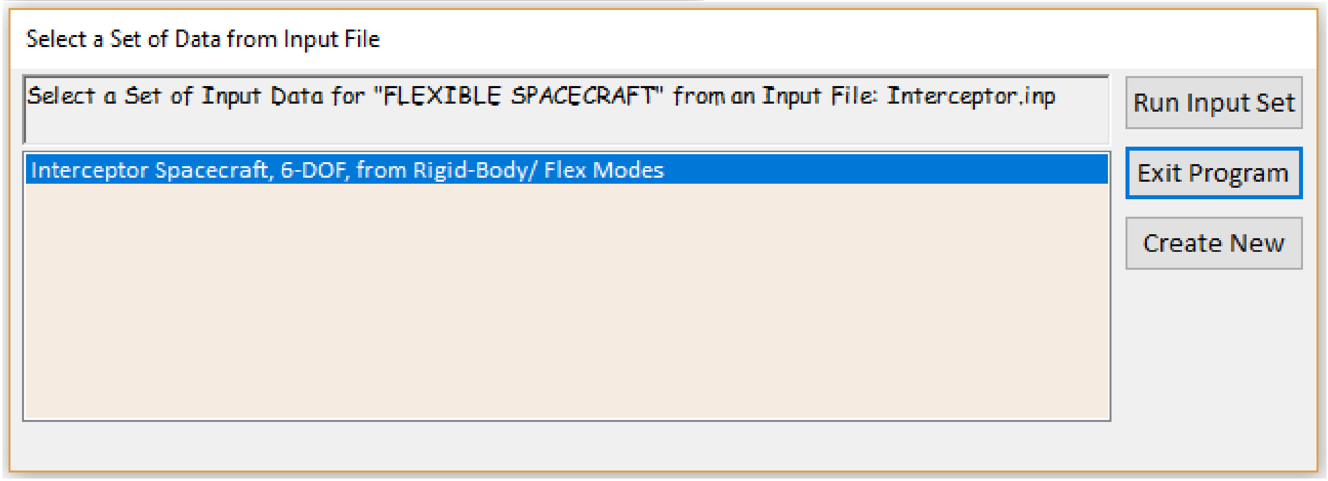
```
FLEXIBLE SPACECRAFT FE MODEL ...
Interceptor Spacecraft, 6-DOF, from Rigid-Body/ Flex Modes
! This Flexible Interceptor Spacecraft has four Divert Thrusters of 200 lb each,
! pointing through the CG. It also has four smaller ACS thrusters of 5 lb each
! for Attitude Control. The directions of the divert thrusters are along +/- y axis
! and along the +/- z axis. They are used to perform translational maneuvers.
! The 4 Forces are applied at Nodes #(5,8,6,7) along -y,-z,+y,+z axes. The ACS thrusters
! are also firing in the Y-Z plane but they are tilted 45 deg as shown below. The System
! Outputs are 2 Accelerations along Y and Z axes and roll, pitch, yaw Rates, at Node# 9

Number of Input Forces applied on flex structure nodes (N_force)                 :  8
Input Force Number, Node Number (see map), Force Direction unit vector along (x,y,z)     :  1     5      0.000      -1.000       0.000
Input Force Number, Node Number (see map), Force Direction unit vector along (x,y,z)     :  2     8      0.000       0.000      -1.000
Input Force Number, Node Number (see map), Force Direction unit vector along (x,y,z)     :  3     6      0.000       1.000       0.000
Input Force Number, Node Number (see map), Force Direction unit vector along (x,y,z)     :  4     7      0.000       0.000       1.000
Input Force Number, Node Number (see map), Force Direction unit vector along (x,y,z)     :  5     2      0.000       0.7071      0.7071
Input Force Number, Node Number (see map), Force Direction unit vector along (x,y,z)     :  6     4      0.000       0.7071     -0.7071
Input Force Number, Node Number (see map), Force Direction unit vector along (x,y,z)     :  7     3      0.000      -0.7071     -0.7071
Input Force Number, Node Number (see map), Force Direction unit vector along (x,y,z)     :  8     1      0.000      -0.7071      0.7071
|
Number of Input Torques applied on flex structure nodes (N_torque)               :  0

Number of Linear Sensors Measuring Translations on the flex structure nodes (N_transl)   :  2
Translation Sensor Numb, Node Numb, Along (1=X,2=Y,3=Z), Type (1=Posit,2=Veloc,3=Acceler):  1     9      2    3
Translation Sensor Numb, Node Numb, Along (1=X,2=Y,3=Z), Type (1=Posit,2=Veloc,3=Acceler):  2     9      3    3

Number of Gyro Sensors Measuring Rotations on the flex structure nodes (N_rotat)         :  3
Rotation Sensor Numbr, Node Number, About (1=X,2=Y,3=Z), Type (1=Posit,2=Veloc,3=Acceler):  1     9      1    2
Rotation Sensor Numbr, Node Number, About (1=X,2=Y,3=Z), Type (1=Posit,2=Veloc,3=Acceler):  2     9      2    2
Rotation Sensor Numbr, Node Number, About (1=X,2=Y,3=Z), Type (1=Posit,2=Veloc,3=Acceler):  3     9      3    2

Number of Flexible Modes (max=600), Mode Shapes and Mode Frequencies are included below  : 42

MODE#  1/  1, Frequency (rad/sec), Damping (zeta), Generalized Mass=    0.0000        0.0000        34.708
DEFINITION OF LOCATIONS (NODES)           phi along X   phi along Y   phi along Z   sigm about X  sigm about Y  sigm about Z

                              Node Numb    Modal Data at the  8 Force Application Points
Right Divert Thruster                5    -0.24115D+02   0.12337D+01   0.91058D+00   0.00000D+00   0.00000D+00   0.00000D+00
Bottom Divert Thruster               8    -0.22697D+02   0.25361D+01   0.22129D+01   0.00000D+00   0.00000D+00   0.00000D+00
Left  Divert Thruster                6    -0.70533D+01   0.12337D+01   0.35153D+01   0.00000D+00   0.00000D+00   0.00000D+00
Top   Divert Thruster                7    -0.84714D+01  -0.68627D-01   0.22129D+01   0.00000D+00   0.00000D+00   0.00000D+00
Top Left  ACS Jet                    2    -0.58680D+01  -0.14524D+02  -0.94422D+01   0.00000D+00   0.00000D+00   0.00000D+00
Bottom Left  ACS Jet                 4    -0.20495D+02  -0.11846D+02  -0.94422D+01   0.00000D+00   0.00000D+00   0.00000D+00
Bottom Right ACS Jet                 3    -0.25301D+02  -0.11846D+02  -0.10176D+02   0.00000D+00   0.00000D+00   0.00000D+00
Top Right ACS Jet                    1    -0.10674D+02  -0.14524D+02  -0.10176D+02   0.00000D+00   0.00000D+00   0.00000D+00

                              Node Numb    Modal Data at the  2 Linear Translation Measurement Points
Sensors Location                     9    -0.15584D+02   0.12985D+02   0.12011D+02  -0.18343D+00  -0.10018D+01   0.12016D+01
Sensors Location                     9    -0.15584D+02   0.12985D+02   0.12011D+02  -0.18343D+00  -0.10018D+01   0.12016D+01

                              Node Numb    Modal Data at the  3 Rotation Measurement Points
Sensors Location                     9    -0.15584D+02   0.12985D+02   0.12011D+02  -0.18343D+00  -0.10018D+01   0.12016D+01
Sensors Location                     9    -0.15584D+02   0.12985D+02   0.12011D+02  -0.18343D+00  -0.10018D+01   0.12016D+01
Sensors Location                     9    -0.15584D+02   0.12985D+02   0.12011D+02  -0.18343D+00  -0.10018D+01   0.12016D+01
```

Input dataset in file "*Interceptor.Inp*" that implements the interceptor spacecraft dynamics including flexibility. Only the first rigid-body mode is shown. It is processed by the Flixan *"Flexible Spacecraft Modeling"* program to produce the system "*Interceptor Spacecraft, 6-DOF, from Rigid-Body/ Flex Modes*" in file "*Interceptor.Qdr*".

To process the already created flex spacecraft dataset, start Flixan, select the project directory, and from the Flixan main menu go to "*Program Functions*", "*Flight Vehicle/ Spacecraft Modeling Tools*", and select the "*Flex Spacecraft (Modal Data)*" program. From the next filename selection menu select the input file name "*interceptor.Inp*", and the systems file name "*interceptor.Qdr*" where the spacecraft system will be saved, and click on "*Process Files*". The next menu shows the Flexible Spacecraft datasets which are already saved in the input file. There is only one. Select it and click on "*Run Input Set*". Flixan will process the dataset and save the spacecraft system in file "*interceptor.Qdr*".


Flixan, Flight Vehicle Modeling & Control System Analysis
Utilities
File Management
Program Functions
View Quad
Help Files
Flight Vehicle/Spacecraft Modeling Tools
Frequency Control Analysis
Robust Control Synthesis Tools
Creating and Modifying Linear Systems
Flight Vehicle, State-Space
Actuator State-Space Models
Flex Spacecraft (Modal Data)
Create Mixing Logic/ TVC
Trim/ Static Perform Analysis
Flex Mode Selection


Select Input and System Filenames
Select a File Name containing the Input Data Set (x.Inp)
Select a File Name containing the State Systems (x.Qdr)
Interceptor.Inp
Interceptor.Qdr
Interceptor.Inp
NewFile.Inp
Interceptor.Qdr
NewFile.Qdr
Create New Input Set
Exit Program
Process Files

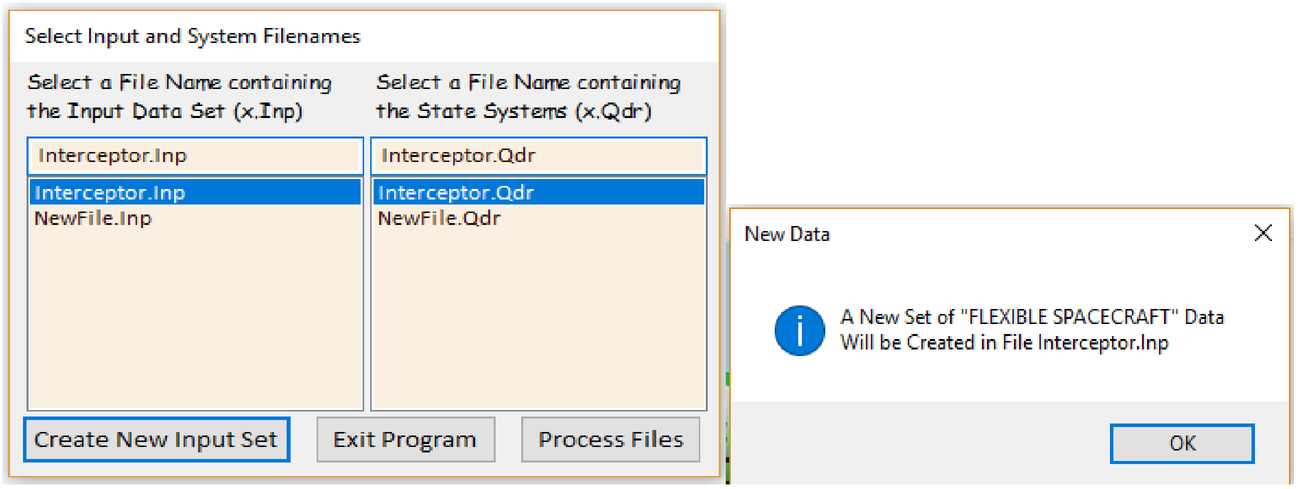


Select a Set of Input Data for "FLEXIBLE SPACECRAFT" from an Input File: Interceptor.inp

Interceptor Spacecraft, 6-DOF, from Rigid-Body/ Flex Modes

1.2 Creating the Flex Spacecraft Input Dataset

We will now show how to create a new spacecraft dataset from scratch, including flexibility. Start the Flixan program, select the project directory, and from the Flixan main menu go to "*Program Functions*", "*Flight Vehicle/ Spacecraft Modeling Tools*", and select the "*Flex Spacecraft (Modal Data)*" program, as before. From the next filename selection menu select the input file name "*interceptor.Inp*", and the systems file name "*interceptor.Qdr*" where the spacecraft data and system will be saved, and click on "*Create New Input Set*".


Select Input and System Filenames

Select a File Name containing the Input Data Set (x.Inp)

Interceptor.Inp

Interceptor.Inp
NewFile.Inp

Select a File Name containing the State Systems (x.Qdr)

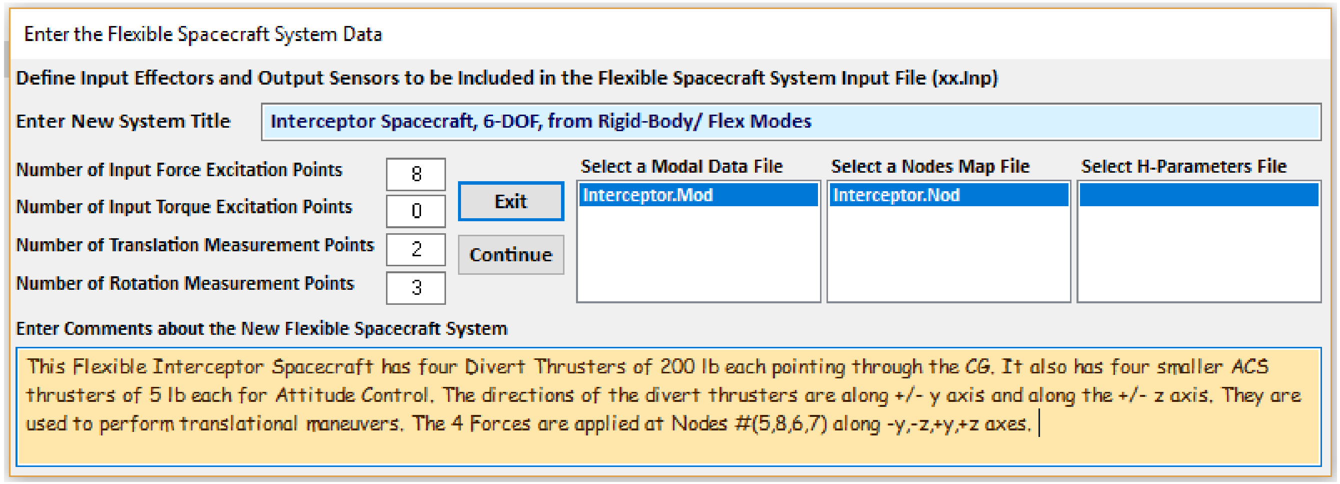
Interceptor.Qdr

Interceptor.Qdr
NewFile.Qdr

New Data

A New Set of "FLEXIBLE SPACECRAFT" Data Will be Created in File Interceptor.Inp

In the next dialog you must define the new spacecraft system by entering its title, some comments at the bottom that describe the spacecraft characteristics, you must define the number of excitation forces, torques, and the number of translations and rotational sensors to be included in the spacecraft model. You must also select the modal data file that includes the finite element modes and the nodes file. The mode shapes and frequencies at the selected node points will be read from the modal data file and be processed by the program.


Enter the Flexible Spacecraft System Data

Define Input Effectors and Output Sensors to be Included in the Flexible Spacecraft System Input File (xx.Inp)

Enter New System Title: Interceptor Spacecraft, 6-DOF, from Rigid-Body/ Flex Modes

Number of Input Force Excitation Points	8
Number of Input Torque Excitation Points	0
Number of Translation Measurement Points	2
Number of Rotation Measurement Points	3

Select a Modal Data File: Interceptor.Mod

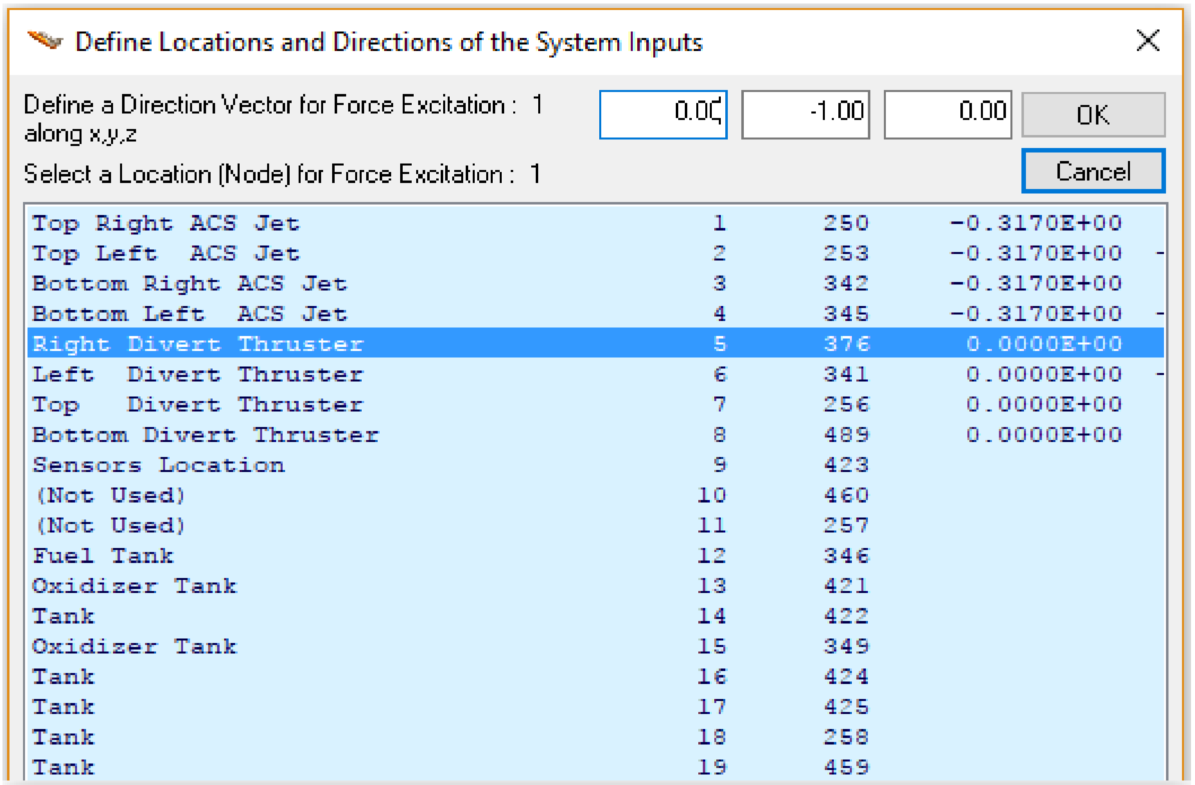
Select a Nodes Map File: Interceptor.Nod

Select H-Parameters File

Enter Comments about the New Flexible Spacecraft System

This Flexible Interceptor Spacecraft has four Divert Thrusters of 200 lb each pointing through the CG. It also has four smaller ACS thrusters of 5 lb each for Attitude Control. The directions of the divert thrusters are along +/- y axis and along the +/- z axis. They are used to perform translational maneuvers. The 4 Forces are applied at Nodes #(5,8,6,7) along -y,-z,+y,+z axes.

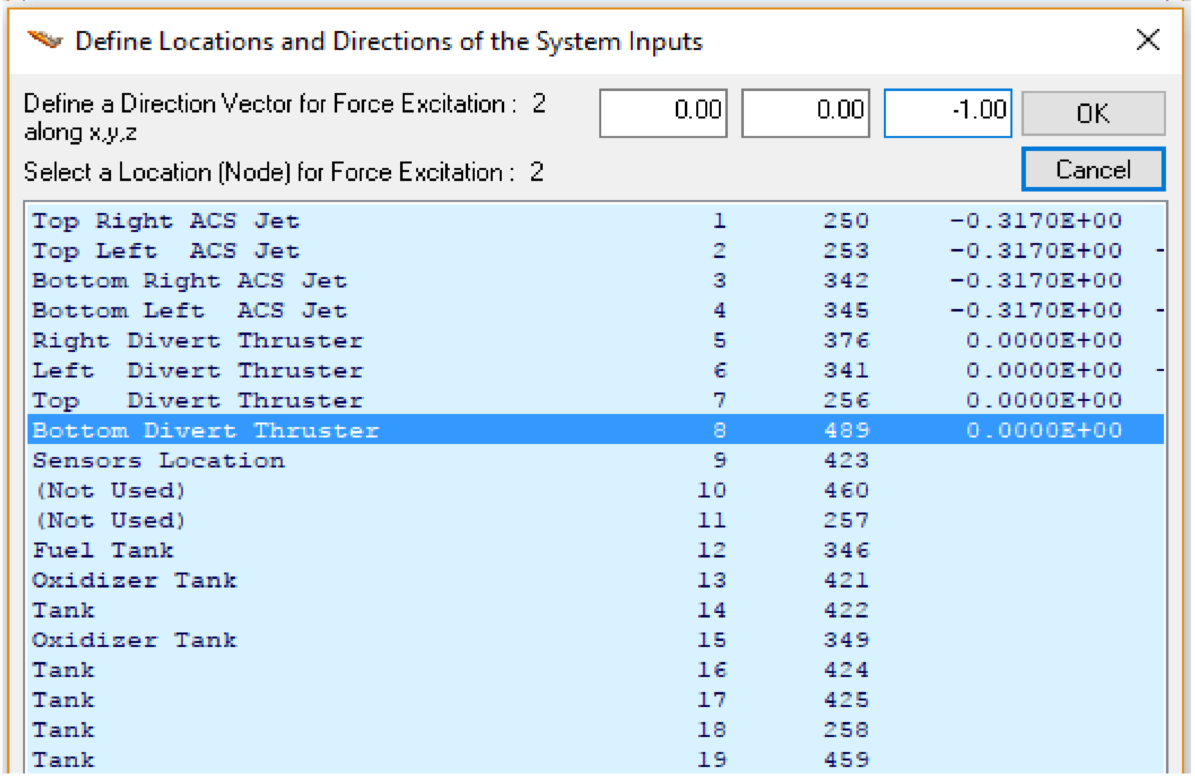
The next step is to define the 8 force excitation points specified with their corresponding force directions using the node selection menus. The node menu displays a list of nodes from the nodes file "*Interceptor.Nod*" and the user selects a node for each excitation force and enters its direction vector. In this case our first input is a force in the –Y direction which is applied at the right divert thruster. Similarly our second input is a force in the –Z direction which is applied at the bottom divert thruster. Click on "OK" to continue.



Define a Direction Vector for Force Excitation : 1 along x,y,z — 0.00 | -1.00 | 0.00

Select a Location (Node) for Force Excitation : 1

```
Top Right ACS Jet              1     250    -0.3170E+00
Top Left  ACS Jet              2     253    -0.3170E+00  -
Bottom Right ACS Jet           3     342    -0.3170E+00
Bottom Left  ACS Jet           4     345    -0.3170E+00  -
Right Divert Thruster          5     376     0.0000E+00
Left  Divert Thruster          6     341     0.0000E+00  -
Top   Divert Thruster          7     256     0.0000E+00
Bottom Divert Thruster         8     489     0.0000E+00
Sensors Location               9     423
(Not Used)                    10     460
(Not Used)                    11     257
Fuel Tank                     12     346
Oxidizer Tank                 13     421
Tank                          14     422
Oxidizer Tank                 15     349
Tank                          16     424
Tank                          17     425
Tank                          18     258
Tank                          19     459
```

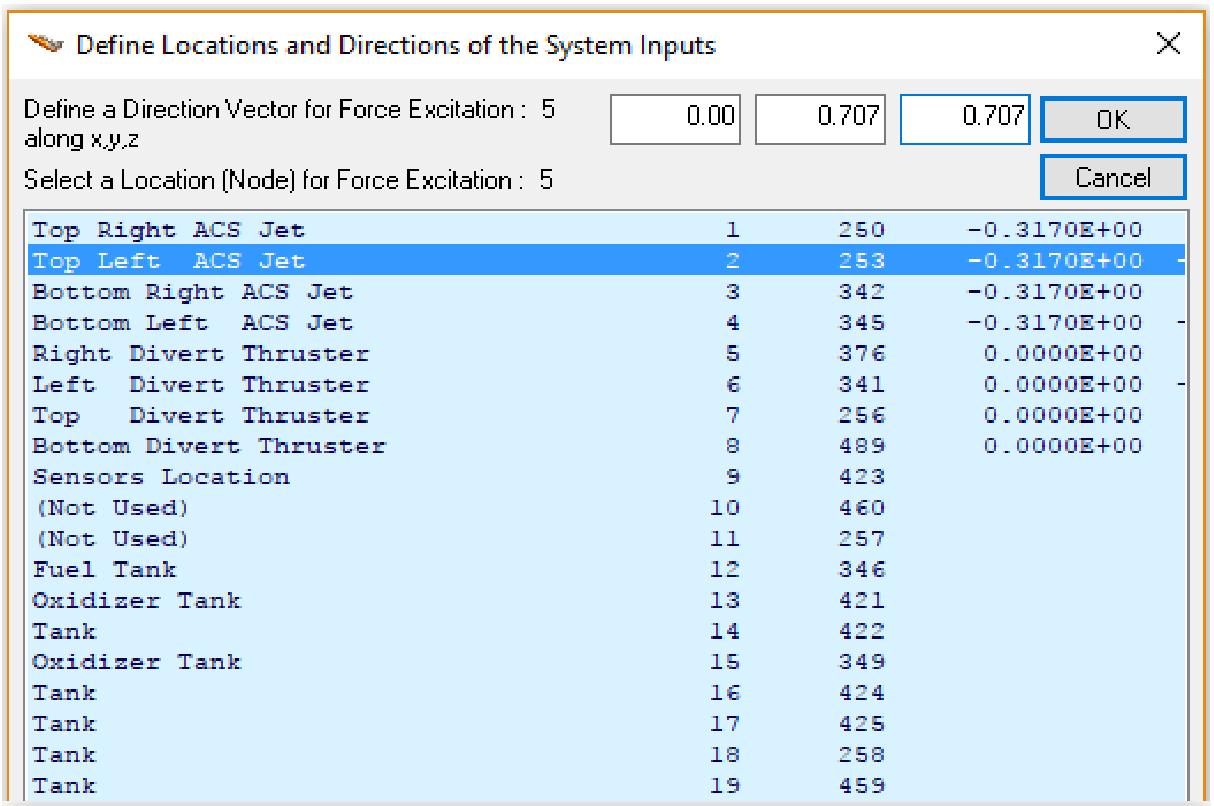


Define a Direction Vector for Force Excitation : 2 along x,y,z — 0.00 | 0.00 | -1.00

Select a Location (Node) for Force Excitation : 2

```
Top Right ACS Jet              1     250    -0.3170E+00
Top Left  ACS Jet              2     253    -0.3170E+00  -
Bottom Right ACS Jet           3     342    -0.3170E+00
Bottom Left  ACS Jet           4     345    -0.3170E+00  -
Right Divert Thruster          5     376     0.0000E+00
Left  Divert Thruster          6     341     0.0000E+00  -
Top   Divert Thruster          7     256     0.0000E+00
Bottom Divert Thruster         8     489     0.0000E+00
Sensors Location               9     423
(Not Used)                    10     460
(Not Used)                    11     257
Fuel Tank                     12     346
Oxidizer Tank                 13     421
Tank                          14     422
Oxidizer Tank                 15     349
Tank                          16     424
Tank                          17     425
Tank                          18     258
Tank                          19     459
```

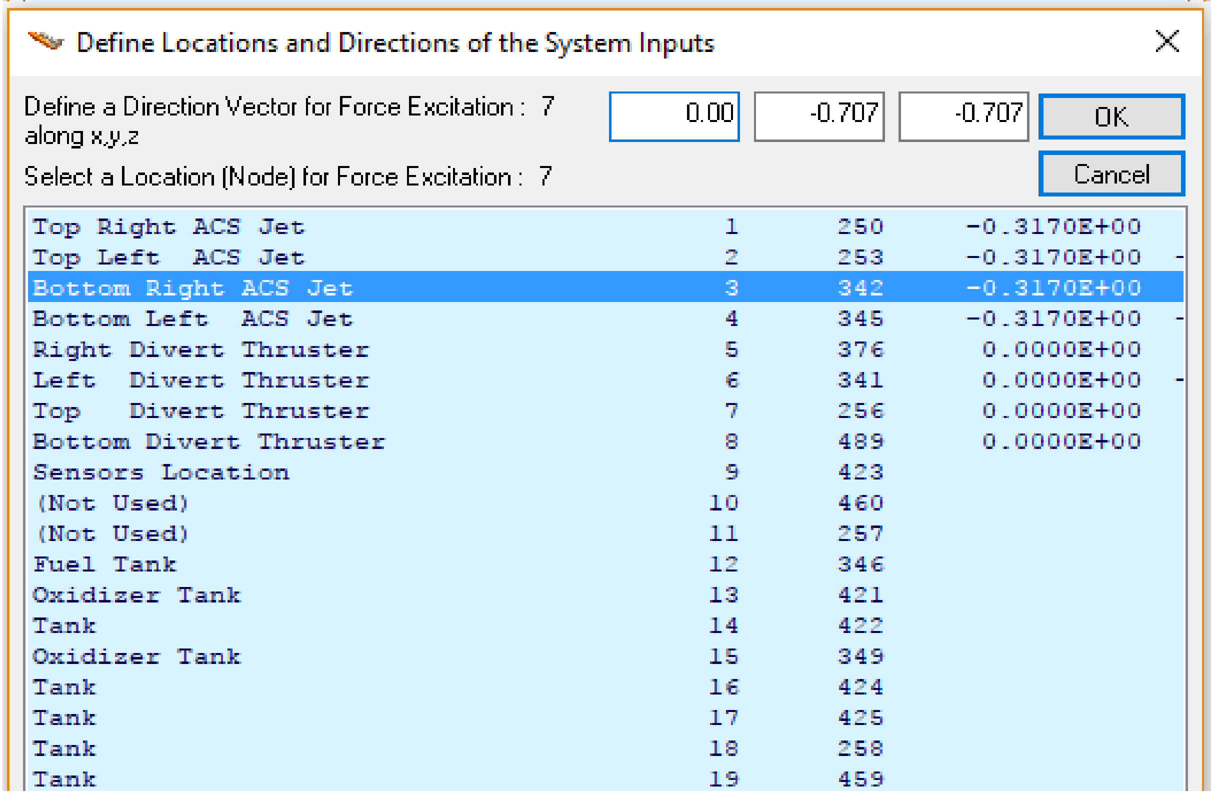
We also use the nodes menu to define the four ACS throttle inputs that apply forces at the four ACS thrusters in skewed directions, as shown in Figure 1b. Click on “OK” to continue.



Define a Direction Vector for Force Excitation : 5 along x,y,z — 0.00 | 0.707 | 0.707

Select a Location (Node) for Force Excitation : 5

```
Top Right ACS Jet            1    250   -0.3170E+00 -
Top Left  ACS Jet            2    253   -0.3170E+00 -
Bottom Right ACS Jet         3    342   -0.3170E+00
Bottom Left  ACS Jet         4    345   -0.3170E+00 -
Right  Divert Thruster       5    376    0.0000E+00
Left   Divert Thruster       6    341    0.0000E+00 -
Top    Divert Thruster       7    256    0.0000E+00
Bottom Divert Thruster       8    489    0.0000E+00
Sensors Location             9    423
(Not Used)                  10    460
(Not Used)                  11    257
Fuel Tank                   12    346
Oxidizer Tank               13    421
Tank                        14    422
Oxidizer Tank               15    349
Tank                        16    424
Tank                        17    425
Tank                        18    258
Tank                        19    459
```

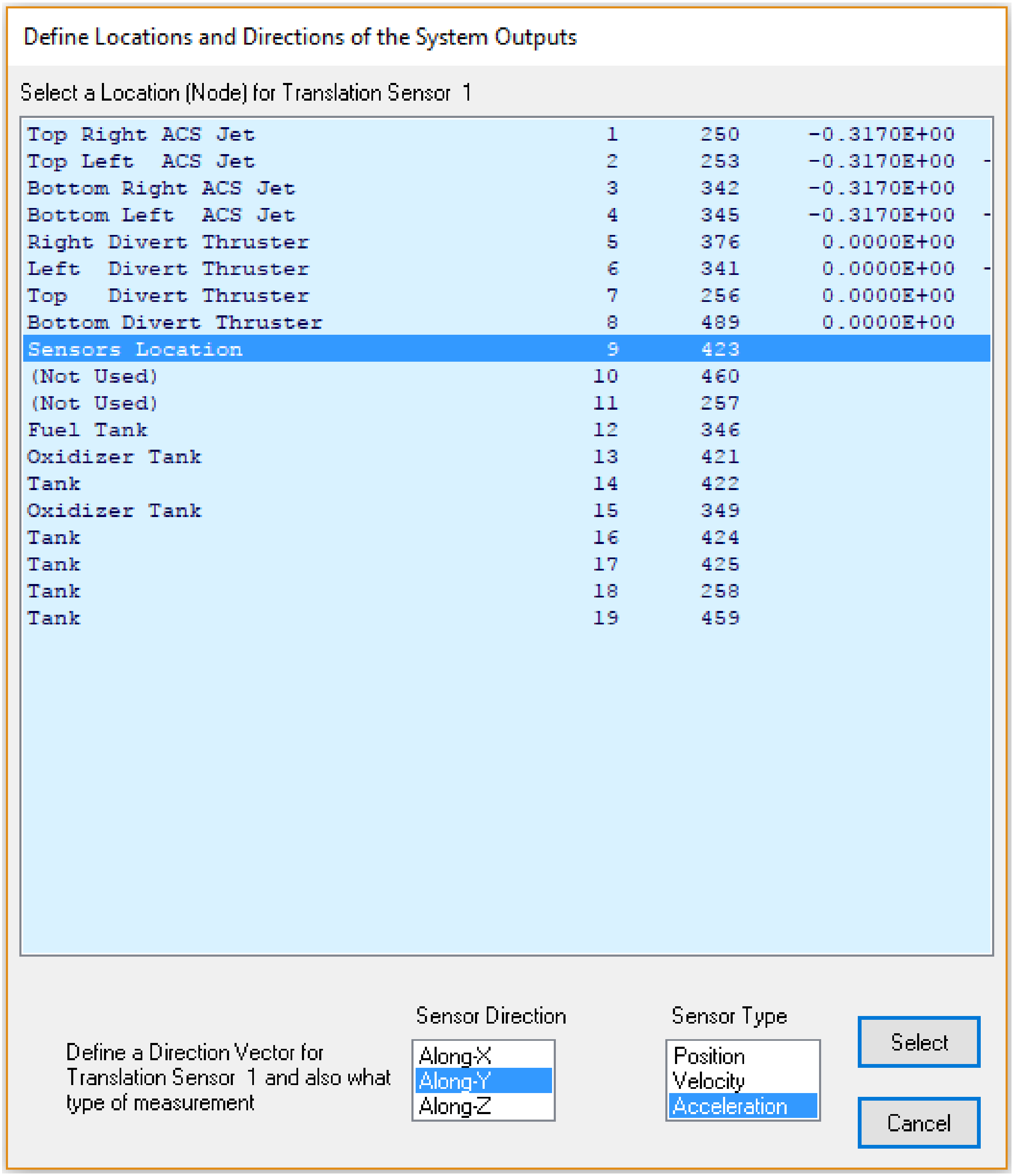


Define a Direction Vector for Force Excitation : 7 along x,y,z — 0.00 | -0.707 | -0.707

Select a Location (Node) for Force Excitation : 7

```
Top Right ACS Jet            1    250   -0.3170E+00
Top Left  ACS Jet            2    253   -0.3170E+00 -
Bottom Right ACS Jet         3    342   -0.3170E+00
Bottom Left  ACS Jet         4    345   -0.3170E+00 -
Right  Divert Thruster       5    376    0.0000E+00
Left   Divert Thruster       6    341    0.0000E+00 -
Top    Divert Thruster       7    256    0.0000E+00
Bottom Divert Thruster       8    489    0.0000E+00
Sensors Location             9    423
(Not Used)                  10    460
(Not Used)                  11    257
Fuel Tank                   12    346
Oxidizer Tank               13    421
Tank                        14    422
Oxidizer Tank               15    349
Tank                        16    424
Tank                        17    425
Tank                        18    258
Tank                        19    459
```

The next step is to define the two translational accelerometer outputs. We select node #9 which is the sensors location, define the type of sensor (accelerometer), and the direction of measurement along Y and Z respectively, and click on “Select”.



Define Locations and Directions of the System Outputs

Select a Location (Node) for Translation Sensor 1

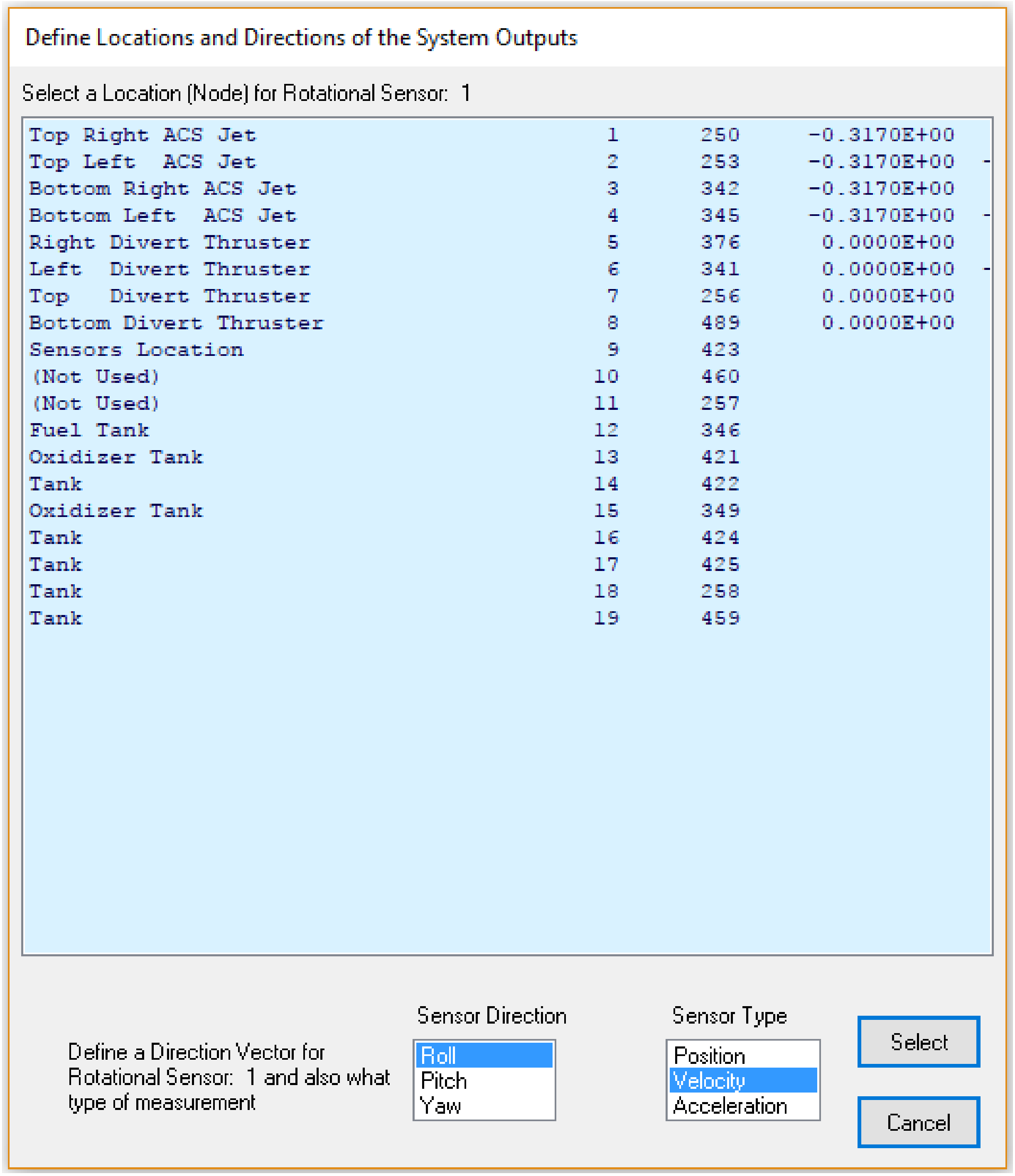
```
Top Right ACS Jet                       1      250     -0.3170E+00
Top Left  ACS Jet                       2      253     -0.3170E+00  -
Bottom Right ACS Jet                    3      342     -0.3170E+00
Bottom Left  ACS Jet                    4      345     -0.3170E+00  -
Right Divert Thruster                   5      376      0.0000E+00
Left  Divert Thruster                   6      341      0.0000E+00  -
Top   Divert Thruster                   7      256      0.0000E+00
Bottom Divert Thruster                  8      489      0.0000E+00
Sensors Location                        9      423
 (Not Used)                            10      460
 (Not Used)                            11      257
Fuel Tank                              12      346
Oxidizer Tank                          13      421
Tank                                   14      422
Oxidizer Tank                          15      349
Tank                                   16      424
Tank                                   17      425
Tank                                   18      258
Tank                                   19      459
```

Define a Direction Vector for Translation Sensor 1 and also what type of measurement

Sensor Direction: Along-X [✓ Along-Y] Along-Z

Sensor Type: Position Velocity [✓ Acceleration]

We must also define the 3 rotational outputs that correspond to the rate gyros. We select node #9 which is the sensors location, define the type of sensor (rotational velocity), and the sensor direction: roll, pitch and yaw respectively, and click on “Select”.



Define Locations and Directions of the System Outputs

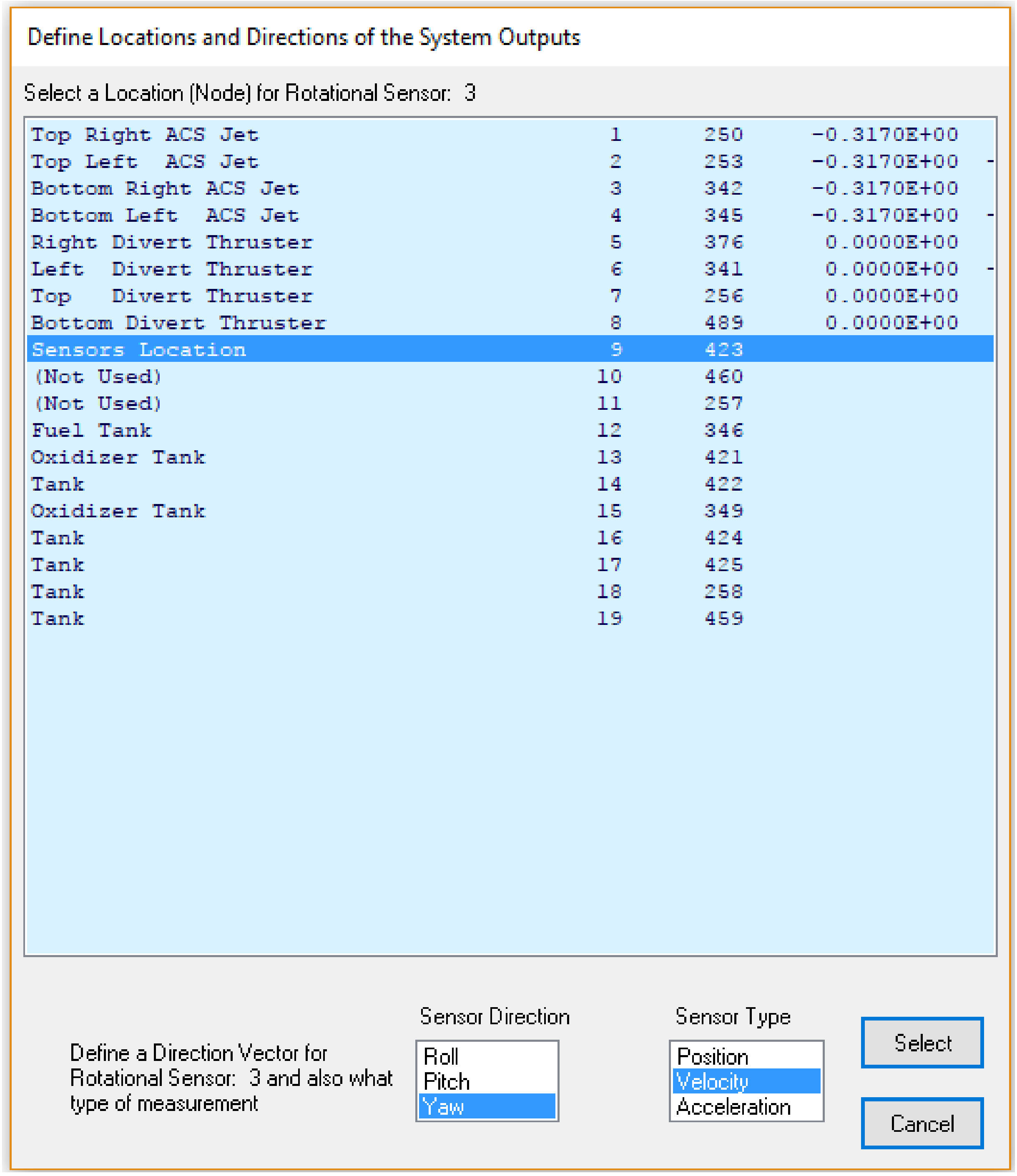
Select a Location (Node) for Rotational Sensor: 1

```
Top Right ACS Jet                          1       250      -0.3170E+00
Top Left  ACS Jet                          2       253      -0.3170E+00  -
Bottom Right ACS Jet                       3       342      -0.3170E+00
Bottom Left  ACS Jet                       4       345      -0.3170E+00  -
Right  Divert Thruster                     5       376       0.0000E+00
Left   Divert Thruster                     6       341       0.0000E+00  -
Top    Divert Thruster                     7       256       0.0000E+00
Bottom Divert Thruster                     8       489       0.0000E+00
Sensors Location                           9       423
(Not Used)                                10       460
(Not Used)                                11       257
Fuel Tank                                 12       346
Oxidizer Tank                             13       421
Tank                                      14       422
Oxidizer Tank                             15       349
Tank                                      16       424
Tank                                      17       425
Tank                                      18       258
Tank                                      19       459
```

Define a Direction Vector for Rotational Sensor: 1 and also what type of measurement

Sensor Direction: [✓ Roll] Pitch Yaw

Sensor Type: Position [✓ Velocity] Acceleration



Define Locations and Directions of the System Outputs

Select a Location (Node) for Rotational Sensor: 3

```
Top Right ACS Jet                  1      250     -0.3170E+00
Top Left  ACS Jet                  2      253     -0.3170E+00  -
Bottom Right ACS Jet               3      342     -0.3170E+00
Bottom Left  ACS Jet               4      345     -0.3170E+00  -
Right  Divert Thruster             5      376      0.0000E+00
Left   Divert Thruster             6      341      0.0000E+00  -
Top    Divert Thruster             7      256      0.0000E+00
Bottom Divert Thruster             8      489      0.0000E+00
Sensors Location                   9      423
(Not Used)                        10      460
(Not Used)                        11      257
Fuel Tank                         12      346
Oxidizer Tank                     13      421
Tank                              14      422
Oxidizer Tank                     15      349
Tank                              16      424
Tank                              17      425
Tank                              18      258
Tank                              19      459
```

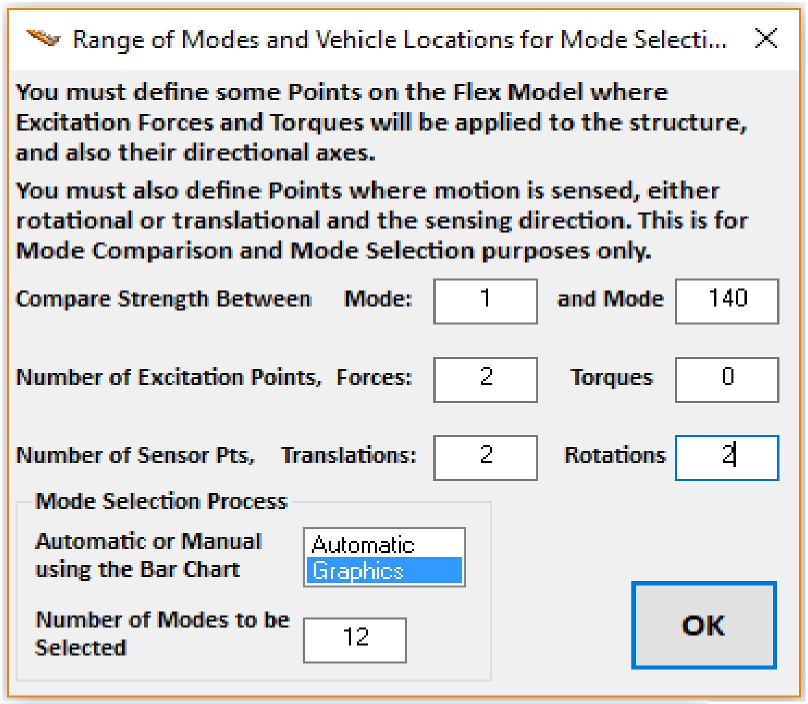
Define a Direction Vector for Rotational Sensor: 3 and also what type of measurement

Sensor Direction: Roll, Pitch, [✓ Yaw]

Sensor Type: Position, [✓ Velocity], Acceleration

Now the spacecraft configuration has been defined in terms of inputs and outputs, the next step is to compare and select the modes. The mode selection program allows the user to use any structural points to calculate the modal strength in between, points which are not necessarily actuators and sensors. This is because the user may wish to analyze modal excitation sensitivity between different locations and force directions.

From the following menu we must define some mode selection parameters. The range of modes to be compared (1 to 140). We begin from mode 1 because we want to include the first 6 modes which are rigid-body modes. Remember, in the Flex Spacecraft modeling program the rigid-body dynamics are not calculated from mass properties but they are implemented like the flex modes, imported from the finite elements model. In this case we define 2 force excitation points, 2 translational sensor points, and 2 rotational sensor points. They will be used only for mode comparison purposes. The locations of the dynamic model actuators and sensors have already been defined. Choose the graphical approach to manually select the modes using the bar chart and click "OK". We will not rescale the modal data this time because the FEM units and directions are acceptable.


Range of Modes and Vehicle Locations for Mode Selecti...

You must define some Points on the Flex Model where Excitation Forces and Torques will be applied to the structure, and also their directional axes.

You must also define Points where motion is sensed, either rotational or translational and the sensing direction. This is for Mode Comparison and Mode Selection purposes only.

Compare Strength Between Mode: 1 and Mode 140

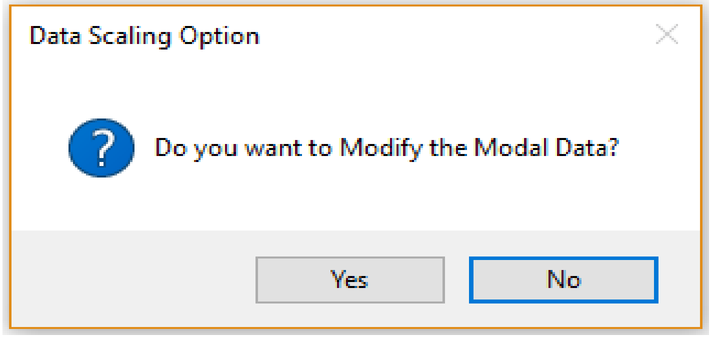
Number of Excitation Points, Forces: 2 Torques 0

Number of Sensor Pts, Translations: 2 Rotations 2

Mode Selection Process

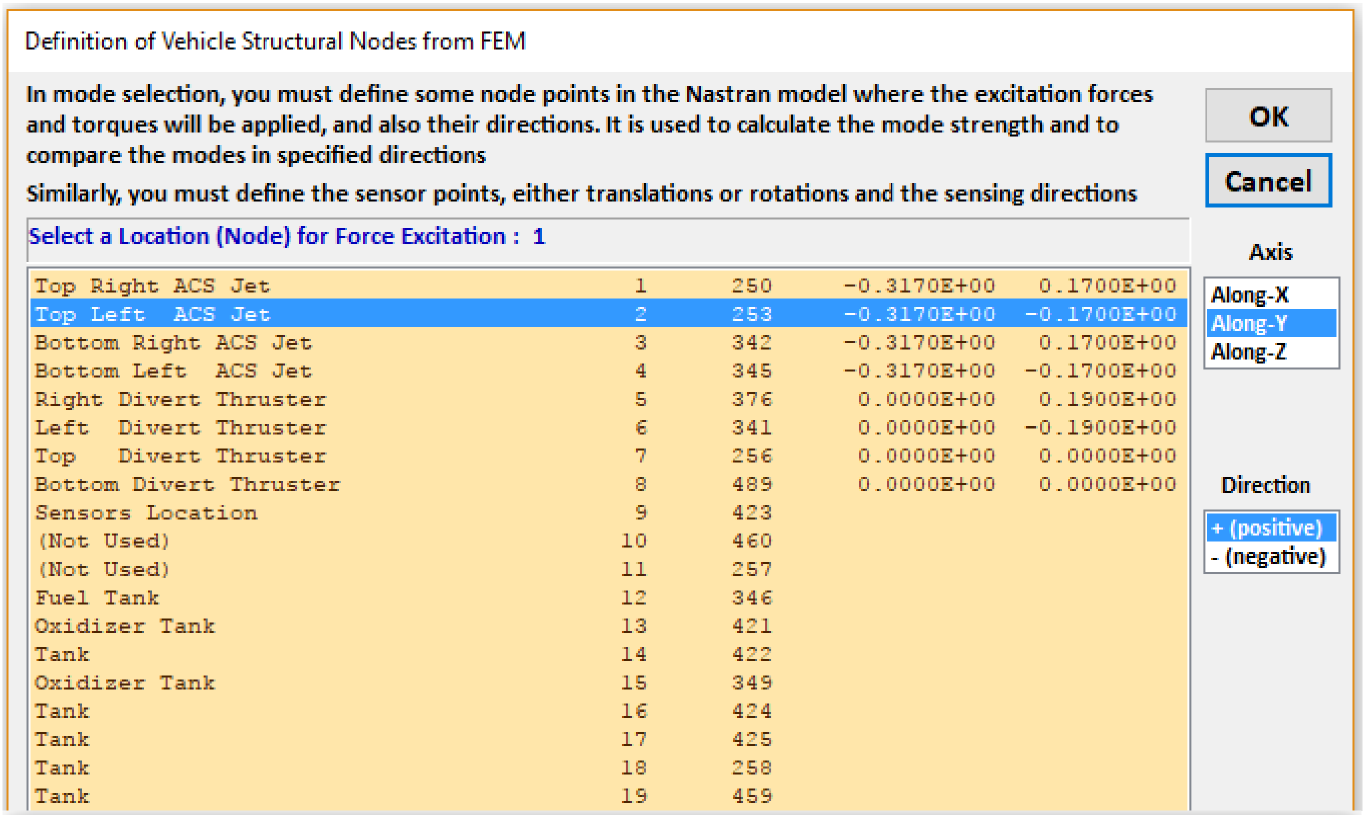
Automatic or Manual using the Bar Chart: Automatic [✓ Graphics]

Number of Modes to be Selected: 12


Data Scaling Option

Do you want to Modify the Modal Data?

From the following menu select nodes for the two force excitation points and the corresponding force directions along +Y and +Z respectively



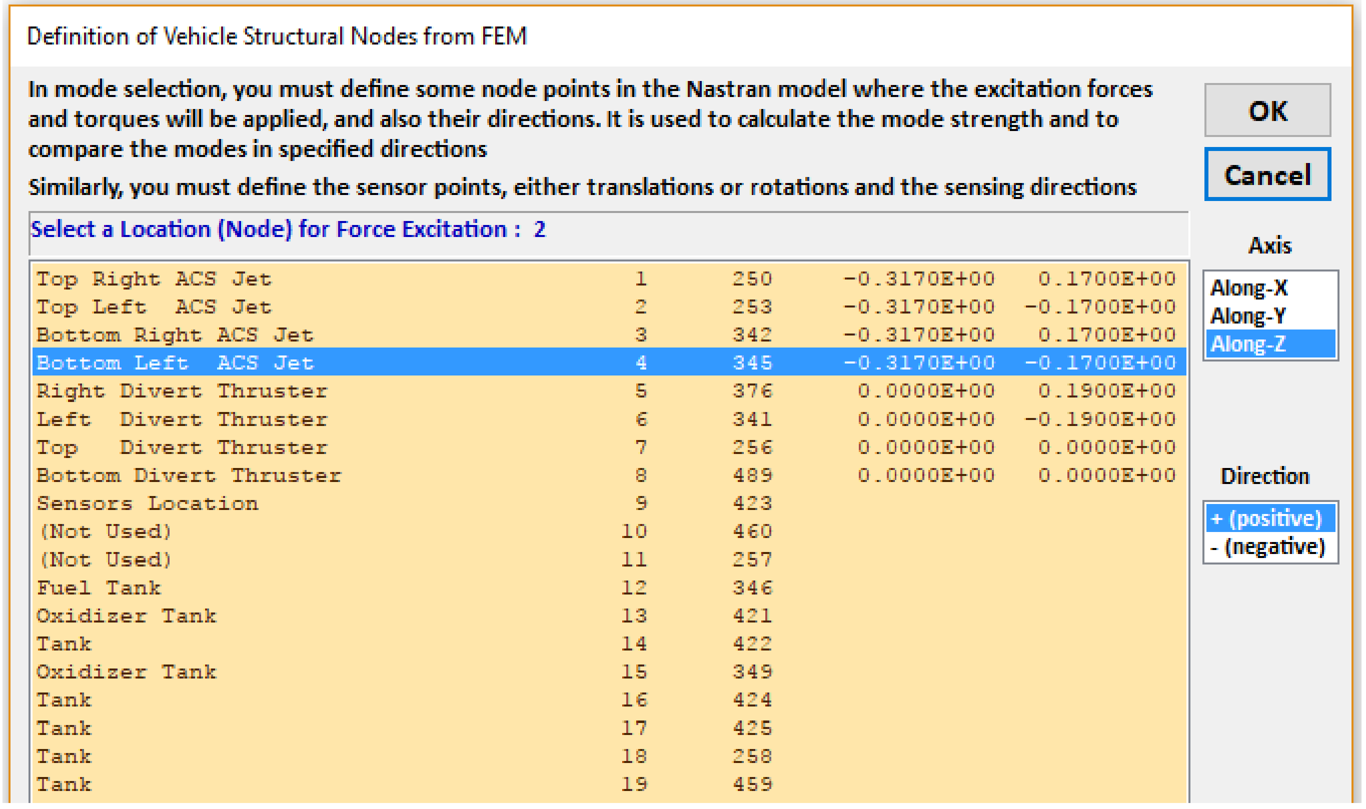
Definition of Vehicle Structural Nodes from FEM

In mode selection, you must define some node points in the Nastran model where the excitation forces and torques will be applied, and also their directions. It is used to calculate the mode strength and to compare the modes in specified directions

Similarly, you must define the sensor points, either translations or rotations and the sensing directions

Select a Location (Node) for Force Excitation : 1

```
Top Right ACS Jet           1     250   -0.3170E+00   0.1700E+00
Top Left  ACS Jet           2     253   -0.3170E+00  -0.1700E+00
Bottom Right ACS Jet        3     342   -0.3170E+00   0.1700E+00
Bottom Left  ACS Jet        4     345   -0.3170E+00  -0.1700E+00
Right  Divert Thruster      5     376    0.0000E+00   0.1900E+00
Left   Divert Thruster      6     341    0.0000E+00  -0.1900E+00
Top    Divert Thruster      7     256    0.0000E+00   0.0000E+00
Bottom Divert Thruster      8     489    0.0000E+00   0.0000E+00
Sensors Location            9     423
(Not Used)                 10     460
(Not Used)                 11     257
Fuel Tank                  12     346
Oxidizer Tank              13     421
Tank                       14     422
Oxidizer Tank              15     349
Tank                       16     424
Tank                       17     425
Tank                       18     258
Tank                       19     459
```



Definition of Vehicle Structural Nodes from FEM

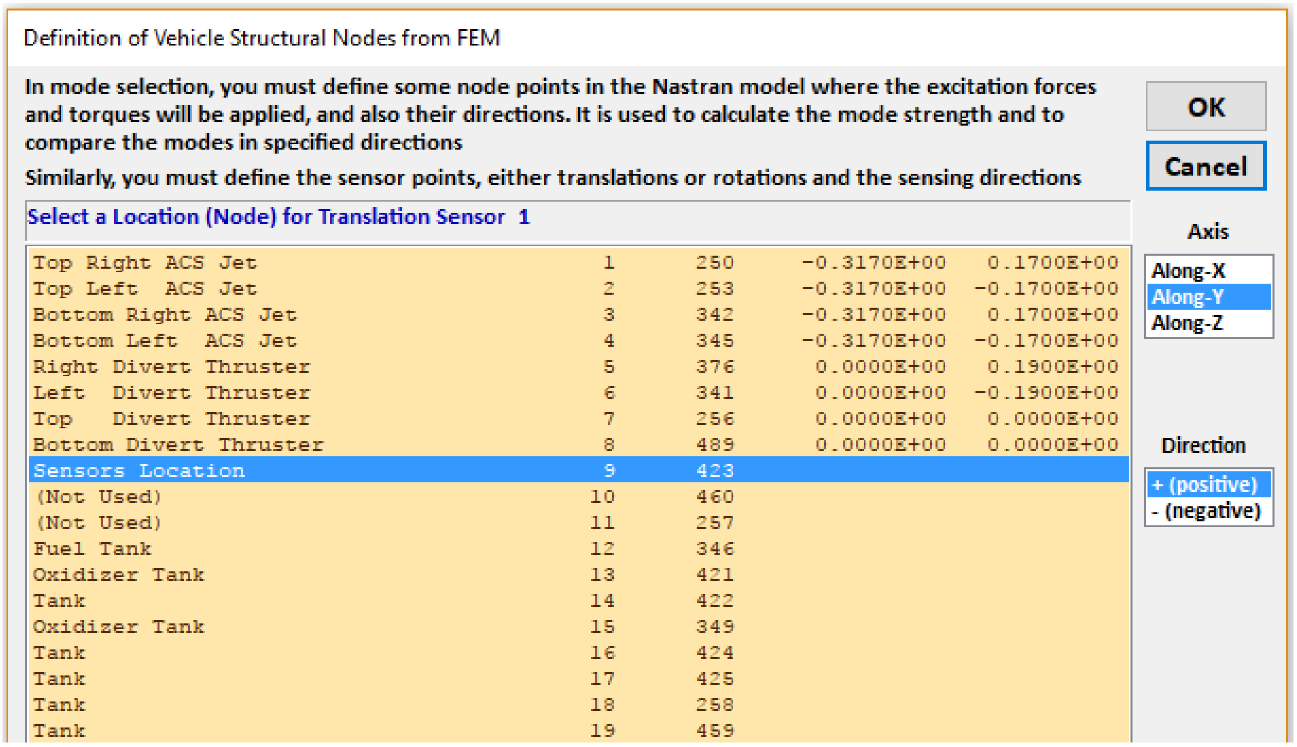
In mode selection, you must define some node points in the Nastran model where the excitation forces and torques will be applied, and also their directions. It is used to calculate the mode strength and to compare the modes in specified directions

Similarly, you must define the sensor points, either translations or rotations and the sensing directions

Select a Location (Node) for Force Excitation : 2

```
Top Right ACS Jet           1     250   -0.3170E+00   0.1700E+00
Top Left  ACS Jet           2     253   -0.3170E+00  -0.1700E+00
Bottom Right ACS Jet        3     342   -0.3170E+00   0.1700E+00
Bottom Left  ACS Jet        4     345   -0.3170E+00  -0.1700E+00
Right  Divert Thruster      5     376    0.0000E+00   0.1900E+00
Left   Divert Thruster      6     341    0.0000E+00  -0.1900E+00
Top    Divert Thruster      7     256    0.0000E+00   0.0000E+00
Bottom Divert Thruster      8     489    0.0000E+00   0.0000E+00
Sensors Location            9     423
(Not Used)                 10     460
(Not Used)                 11     257
Fuel Tank                  12     346
Oxidizer Tank              13     421
Tank                       14     422
Oxidizer Tank              15     349
Tank                       16     424
Tank                       17     425
Tank                       18     258
Tank                       19     459
```

From the following menu select node #9 for the two translational sensor points and the corresponding directions along +Y and +Z respectively.



Definition of Vehicle Structural Nodes from FEM

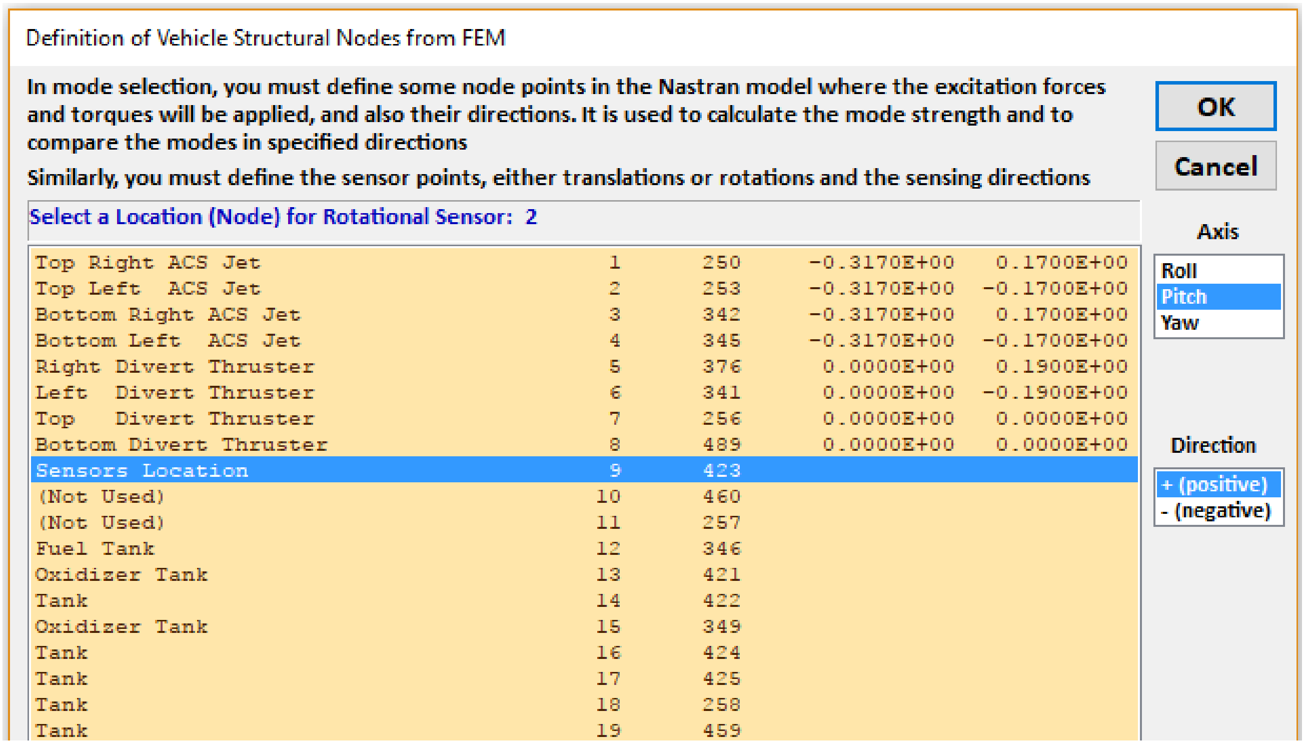
In mode selection, you must define some node points in the Nastran model where the excitation forces and torques will be applied, and also their directions. It is used to calculate the mode strength and to compare the modes in specified directions

Similarly, you must define the sensor points, either translations or rotations and the sensing directions

Select a Location (Node) for Translation Sensor 1

```
Top Right ACS Jet               1     250    -0.3170E+00   0.1700E+00
Top Left  ACS Jet               2     253    -0.3170E+00  -0.1700E+00
Bottom Right ACS Jet            3     342    -0.3170E+00   0.1700E+00
Bottom Left  ACS Jet            4     345    -0.3170E+00  -0.1700E+00
Right Divert Thruster           5     376     0.0000E+00   0.1900E+00
Left  Divert Thruster           6     341     0.0000E+00  -0.1900E+00
Top   Divert Thruster           7     256     0.0000E+00   0.0000E+00
Bottom Divert Thruster          8     489     0.0000E+00   0.0000E+00
Sensors Location                9     423
(Not Used)                     10     460
(Not Used)                     11     257
Fuel Tank                      12     346
Oxidizer Tank                  13     421
Tank                           14     422
Oxidizer Tank                  15     349
Tank                           16     424
Tank                           17     425
Tank                           18     258
Tank                           19     459
```

From the following menu select node #9 for the two rotational sensor points measuring in the pitch and yaw directions respectively.



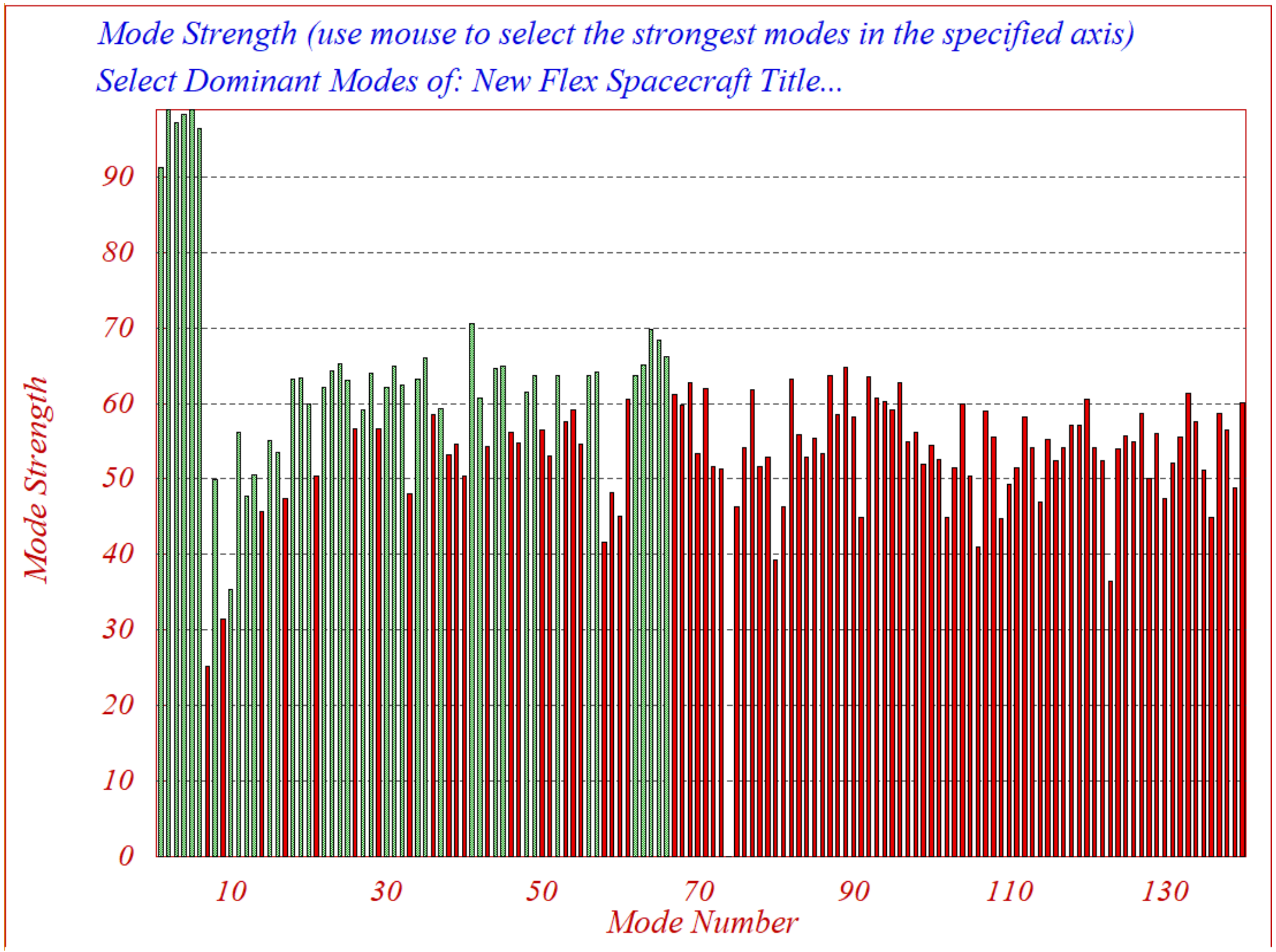
Definition of Vehicle Structural Nodes from FEM

In mode selection, you must define some node points in the Nastran model where the excitation forces and torques will be applied, and also their directions. It is used to calculate the mode strength and to compare the modes in specified directions

Similarly, you must define the sensor points, either translations or rotations and the sensing directions

Select a Location (Node) for Rotational Sensor: 2

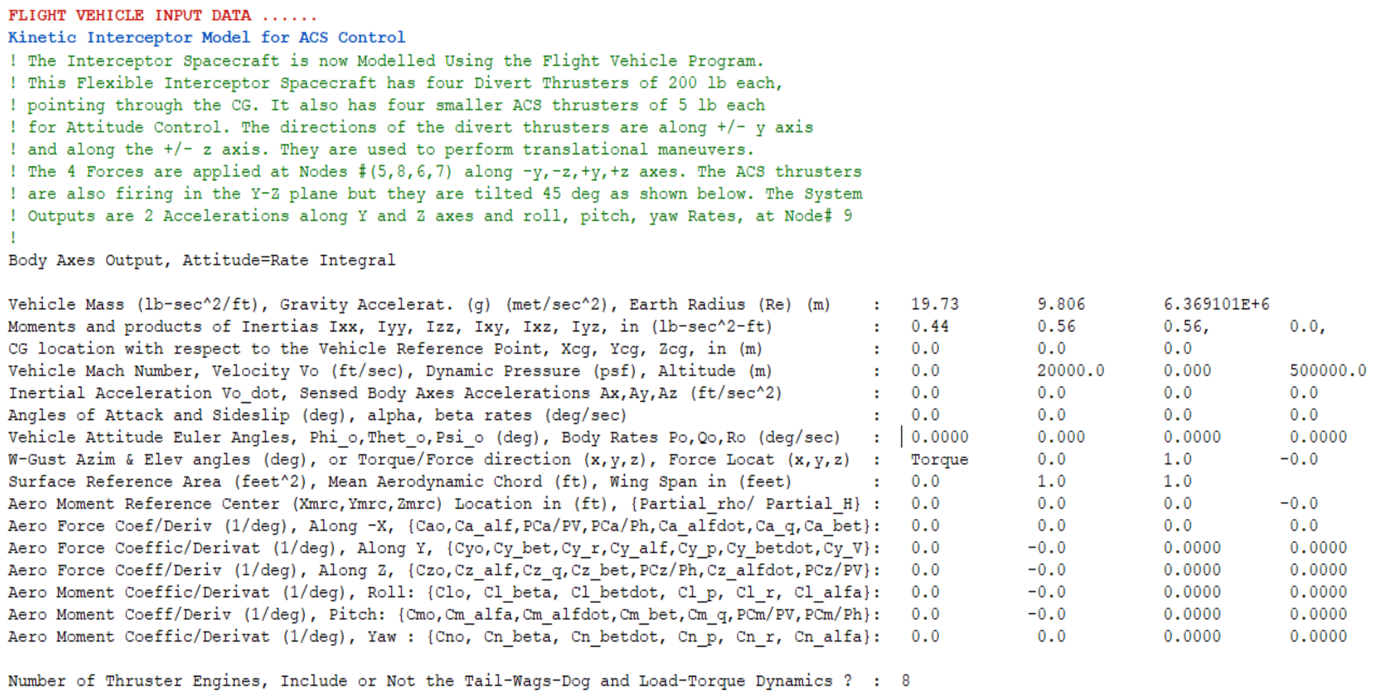
```
Top Right ACS Jet               1     250    -0.3170E+00   0.1700E+00
Top Left  ACS Jet               2     253    -0.3170E+00  -0.1700E+00
Bottom Right ACS Jet            3     342    -0.3170E+00   0.1700E+00
Bottom Left  ACS Jet            4     345    -0.3170E+00  -0.1700E+00
Right Divert Thruster           5     376     0.0000E+00   0.1900E+00
Left  Divert Thruster           6     341     0.0000E+00  -0.1900E+00
Top   Divert Thruster           7     256     0.0000E+00   0.0000E+00
Bottom Divert Thruster          8     489     0.0000E+00   0.0000E+00
Sensors Location                9     423
(Not Used)                     10     460
(Not Used)                     11     257
Fuel Tank                      12     346
Oxidizer Tank                  13     421
Tank                           14     422
Oxidizer Tank                  15     349
Tank                           16     424
Tank                           17     425
Tank                           18     258
Tank                           19     459
```



From the above chart the user can select a large number of dominant modes by using the mouse. A mode changes color from red to green when it is selected. Modes: 1, 2, 3, 4, 5, 6, 8, 10, 11, 12, 13, 15, 16, 18, 19, 20, 22, 23, 24, 25, 27, 28, 30, 31, 32, 34, etc. were included in the flexible spacecraft model. The first 6 modes are rigid-body modes. The flex mode selection was based on the modal strength that is calculated between the thrusters and the sensors specified. Click on the "Enter" key, and the flexible spacecraft description dataset with the selected modal data will be saved in file *"Interceptor.inp"*. Its title is "*Interceptor Spacecraft, 6-DOF, from Rigid-Body/ Flex Modes*" and the spacecraft description is inserted below the title.

1.3 Spacecraft Model Created by the Flight Vehicle Modeling Program

The input file “*Interceptor.Inp”* also contains a flight vehicle dataset, shown below, that generates the same flex spacecraft model using the flight vehicle modeling program. Its title is “*Kinetic Interceptor Model for ACS Control*”. The rigid-body dynamics are implemented from the spacecraft mass properties and it is combined with a set of selected structural modes that obviously do not include the rigid-body modes. The previously selected modes are saved as a separate dataset in the same input file under the title: “*Kinetic Interceptor Vehicle, Divert Flex Modes*”. This system also includes the four divert thrusters and the four ACS thrusters implemented as throttling engines where the throttle varies between zero and ±1, since the maximum thrust is included in the dynamics. The system outputs are also 3 rate-gyros (roll, pitch, yaw), and two accelerometers along y and z. This state-space system is also saved in file “*Interceptor.Qdr*” to be used for control analysis. The title of the selected modes is included at the bottom of the vehicle dataset, below the number of modes, in order to be processed with the vehicle data.

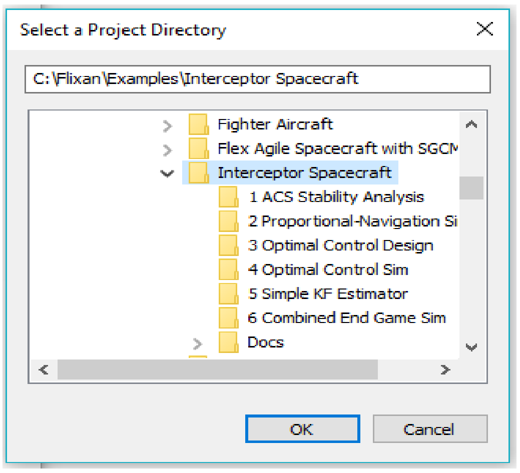


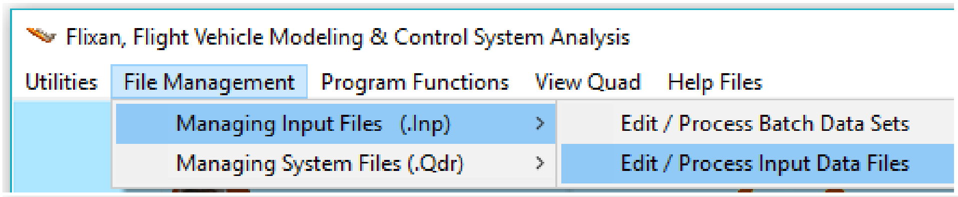
```
FLIGHT VEHICLE INPUT DATA ......
Kinetic Interceptor Model for ACS Control
! The Interceptor Spacecraft is now Modelled Using the Flight Vehicle Program.
! This Flexible Interceptor Spacecraft has four Divert Thrusters of 200 lb each,
! pointing through the CG. It also has four smaller ACS thrusters of 5 lb each
! for Attitude Control. The directions of the divert thrusters are along +/- y axis
! and along the +/- z axis. They are used to perform translational maneuvers.
! The 4 Forces are applied at Nodes #(5,8,6,7) along -y,-z,+y,+z axes. The ACS thrusters
! are also firing in the Y-Z plane but they are tilted 45 deg as shown below. The System
! Outputs are 2 Accelerations along Y and Z axes and roll, pitch, yaw Rates, at Node# 9
!
Body Axes Output, Attitude=Rate Integral

Vehicle Mass (lb-sec^2/ft), Gravity Accelerat. (g) (met/sec^2), Earth Radius (Re) (m)    :   19.73       9.806       6.369101E+6
Moments and products of Inertias Ixx, Iyy, Izz, Ixy, Ixz, Iyz, in (lb-sec^2-ft)         :   0.44        0.56        0.56,       0.0,
CG location with respect to the Vehicle Reference Point, Xcg, Ycg, Zcg, in (m)           :   0.0         0.0         0.0
Vehicle Mach Number, Velocity Vo (ft/sec), Dynamic Pressure (psf), Altitude (m)          :   0.0         20000.0     0.000       500000.0
Inertial Acceleration Vo_dot, Sensed Body Axes Accelerations Ax,Ay,Az (ft/sec^2)         :   0.0         0.0         0.0         0.0
Angles of Attack and Sideslip (deg), alpha, beta rates (deg/sec)                         :   0.0         0.0         0.0         0.0
Vehicle Attitude Euler Angles, Phi_o,Thet_o,Psi_o (deg), Body Rates Po,Qo,Ro (deg/sec)   :  |0.0000      0.000       0.0000      0.0000
W-Gust Azim & Elev angles (deg), or Torque/Force direction (x,y,z), Force Locat (x,y,z)  :   Torque      0.0         1.0        -0.0
Surface Reference Area (feet^2), Mean Aerodynamic Chord (ft), Wing Span in (feet)        :   0.0         1.0         1.0
Aero Moment Reference Center (Xmrc,Ymrc,Zmrc) Location in (ft), {Partial_rho/ Partial_H} :   0.0         0.0         0.0        -0.0
Aero Force Coef/Deriv (1/deg), Along -X, {Cao,Ca_alf,PCa/PV,PCa/Ph,Ca_alfdot,Ca_q,Ca_bet}:   0.0         0.0         0.0         0.0
Aero Force Coeffic/Derivat (1/deg), Along Y, {Cyo,Cy_bet,Cy_r,Cy_alf,Cy_p,Cy_betdot,Cy_V}:   0.0        -0.0         0.0000      0.0000
Aero Force Coeff/Deriv (1/deg), Along Z, {Czo,Cz_alf,Cz_q,Cz_bet,PCz/Ph,Cz_alfdot,PCz/PV}:   0.0        -0.0         0.0000      0.0000
Aero Moment Coeffic/Derivat (1/deg), Roll: {Clo, Cl_beta, Cl_betdot, Cl_p, Cl_r, Cl_alfa}:   0.0        -0.0         0.0000      0.0000
Aero Moment Coeff/Deriv (1/deg), Pitch: {Cmo,Cm_alfa,Cm_alfdot,Cm_bet,Cm_q,PCm/PV,PCm/Ph}:   0.0        -0.0         0.0000      0.0000
Aero Moment Coeffic/Derivat (1/deg), Yaw : {Cno, Cn_beta, Cn_betdot, Cn_p, Cn_r, Cn_alfa}:   0.0         0.0         0.0000      0.0000

Number of Thruster Engines, Include or Not the Tail-Wags-Dog and Load-Torque Dynamics ?  :  8
```

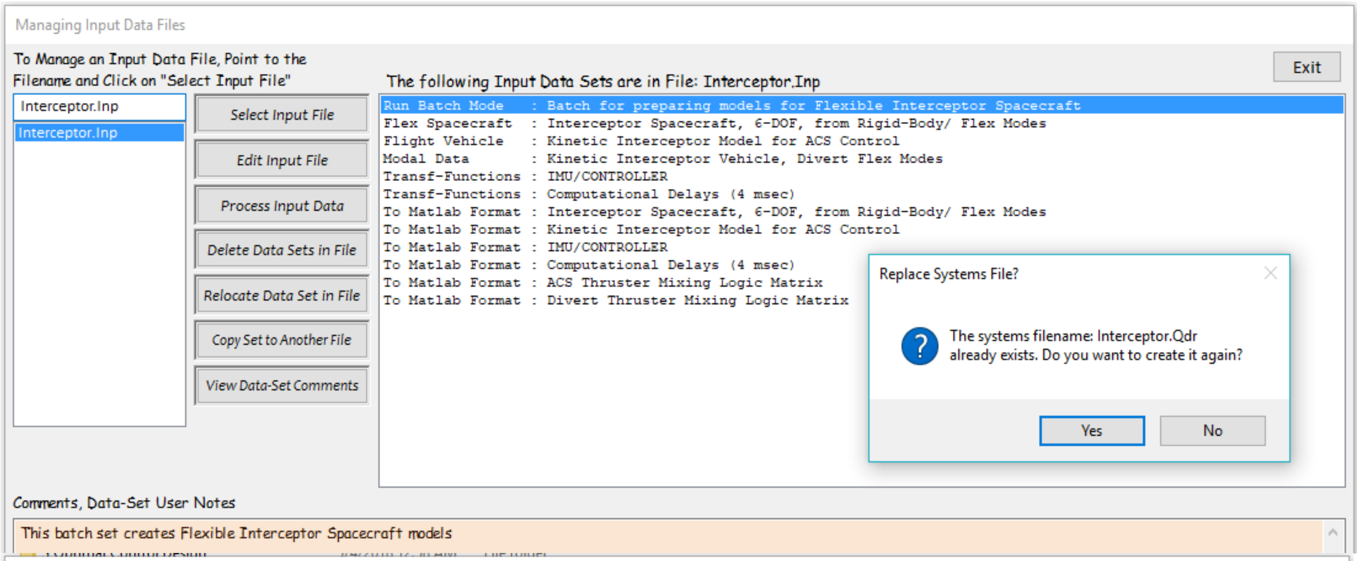
```
RCS Jet No:  1     Right Divert Thruster -Y  (200 lb jet)                          : Divert-Y      Throttling                Node 5
Engine Nominal Thrust, and Maximum Thrust in (lb)    (for throttling)              :  0.0          15.0
Mounting Angles wrt Vehicle (Dyn,Dzn), Maximum Deflections from Mount (Dymax,Dzmax) (deg):  0.0     -90.0         0.0          0.0
Eng Mass (slug), Inertia about Gimbal (lb-sec^2-ft), Moment Arm, engine CG to gimbal (ft):  0.0      0.0          0.0
Thruster location with respect to the Vehicle Reference Axes, Xjet, Yjet, Zjet,    (ft)   :  0.0,      0.19,        0.0
RCS Jet No:  2     Bottom Divert Thruster -Z  (200 lb jet)                         : Divert-Z      Throttling                Node 8
Engine Nominal Thrust, and Maximum Thrust in (lb)    (for throttling)              :  0.0          15.0
Mounting Angles wrt Vehicle (Dyn,Dzn), Maximum Deflections from Mount (Dymax,Dzmax) (deg): +90.0    0.0          0.0          0.0
Eng Mass (slug), Inertia about Gimbal (lb-sec^2-ft), Moment Arm, engine CG to gimbal (ft):  0.0      0.0          0.0
Thruster location with respect to the Vehicle Reference Axes, Xjet, Yjet, Zjet,    (ft)   :  0.0,      0.0,        -0.19
RCS Jet No:  3     Left  Divert Thruster +Y  (200 lb jet)                          : Divert+Y      Throttling                Node 6
Engine Nominal Thrust, and Maximum Thrust in (lb)    (for throttling)              :  0.0          15.0
Mounting Angles wrt Vehicle (Dyn,Dzn), Maximum Deflections from Mount (Dymax,Dzmax) (deg):  0.0     +90.0         0.0          0.0
Eng Mass (slug), Inertia about Gimbal (lb-sec^2-ft), Moment Arm, engine CG to gimbal (ft):  0.0      0.0          0.0
Thruster location with respect to the Vehicle Reference Axes, Xjet, Yjet, Zjet,    (ft)   :  0.0,     -0.19,        0.0
RCS Jet No:  4     Top Divert Thruster +Z  (200 lb jet)                            : Divert+Z      Throttling                Node 7
Engine Nominal Thrust, and Maximum Thrust in (lb)    (for throttling)              :  0.0          15.0
Mounting Angles wrt Vehicle (Dyn,Dzn), Maximum Deflections from Mount (Dymax,Dzmax) (deg): -90.0    0.0          0.0          0.0
Eng Mass (slug), Inertia about Gimbal (lb-sec^2-ft), Moment Arm, engine CG to gimbal (ft):  0.0      0.0          0.0
Thruster location with respect to the Vehicle Reference Axes, Xjet, Yjet, Zjet,    (ft)   :  0.0,      0.0,         0.19
RCS Jet No:  5     Top Left ACS Jet                                                : ACS-TL        Throttling                Node 2
Engine Nominal Thrust, and Maximum Thrust in (lb)    (for throttling)              :  0.0          5.0
Mounting Angles wrt Vehicle (Dyn,Dzn), Maximum Deflections from Mount (Dymax,Dzmax) (deg): -45.0   +90.0         0.0          0.0
Eng Mass (slug), Inertia about Gimbal (lb-sec^2-ft), Moment Arm, engine CG to gimbal (ft):  0.0      0.0          0.0
Thruster location with respect to the Vehicle Reference Axes, Xjet, Yjet, Zjet,    (ft)   : -0.317,   -0.05,       -0.19
RCS Jet No:  6     Bottom Left ACS Jet                                             : ACS-BL        Throttling                Node 4
Engine Nominal Thrust, and Maximum Thrust in (lb)    (for throttling)              :  0.0          5.0
Mounting Angles wrt Vehicle (Dyn,Dzn), Maximum Deflections from Mount (Dymax,Dzmax) (deg): +45.0   +90.0         0.0          0.0
Eng Mass (slug), Inertia about Gimbal (lb-sec^2-ft), Moment Arm, engine CG to gimbal (ft):  0.0      0.0          0.0
Thruster location with respect to the Vehicle Reference Axes, Xjet, Yjet, Zjet,    (ft)   : -0.317,   -0.05,        0.19
RCS Jet No:  7     Bottom Right ACS Jet                                            : ACS-BR        Throttling                Node 3
Engine Nominal Thrust, and Maximum Thrust in (lb)    (for throttling)              :  0.0          5.0
Mounting Angles wrt Vehicle (Dyn,Dzn), Maximum Deflections from Mount (Dymax,Dzmax) (deg): +45.0   -90.0         0.0          0.0
Eng Mass (slug), Inertia about Gimbal (lb-sec^2-ft), Moment Arm, engine CG to gimbal (ft):  0.0      0.0          0.0
Thruster location with respect to the Vehicle Reference Axes, Xjet, Yjet, Zjet,    (ft)   : -0.317,    0.05,        0.19
RCS Jet No:  8     Top Right ACS Jet                                               : ACS-TR        Throttling                Node 1
Engine Nominal Thrust, and Maximum Thrust in (lb)    (for throttling)              :  0.0          5.0
Mounting Angles wrt Vehicle (Dyn,Dzn), Maximum Deflections from Mount (Dymax,Dzmax) (deg): -45.0   -90.0         0.0          0.0
Eng Mass (slug), Inertia about Gimbal (lb-sec^2-ft), Moment Arm, engine CG to gimbal (ft):  0.0      0.0          0.0
Thruster location with respect to the Vehicle Reference Axes, Xjet, Yjet, Zjet,    (ft)   : -0.317,    0.05,       -0.19

Number of External Torques on the Vehicle                                          :  0

Number of Gyros, (Attitude and Rate)                                               :  3
Gyro No  1 Axis:(Pitch,Yaw,Roll), (Attitude, Rate, Accelerat), Sensor Locat,   Node 2   :  Roll   Rate          0.0        0.0        0.0
Gyro No  2 Axis:(Pitch,Yaw,Roll), (Attitude, Rate, Accelerat), Sensor Locat,   Node 2   :  Pitch  Rate          0.0        0.0        0.0
Gyro No  3 Axis:(Pitch,Yaw,Roll), (Attitude, Rate, Accelerat), Sensor Locat,   Node 2   :  Yaw    Rate          0.0        0.0        0.0

Number of Accelerometers, Along Axes: (x,y,z)                                      :  2
Acceleromet No  1 Axis:(X,Y,Z), (Position, Velocity, Acceleration), Sensor Loc, Node 9  :  Y-axis Accelerat.    0.5        0.0        0.0
Acceleromet No  2 Axis:(X,Y,Z), (Position, Velocity, Acceleration), Sensor Loc, Node 9  :  Z-axis Accelerat.    0.5        0.0        0.0

Number of Bending Modes                                                            : 40
Kinetic Interceptor Vehicle, Divert Flex Modes

SELECTED MODAL DATA AND LOCATIONS FOR : Flex Modes
Kinetic Interceptor Vehicle, Divert Flex Modes
! This set of modes was selected mainly between the Divert Thrusters and the
! Y and Z Accelerometers in Node-9

MODE#  1/  8, Frequency (rad/sec), Damping (zeta), Generalized Mass=    998.88      0.10000E-01    4000.0
DEFINITION OF LOCATIONS (NODES)              phi along X   phi along Y   phi along Z   sigm about X  sigm about Y  sigm about Z

                              Node ID#     Modal Data at the  8 Engines, (x,y,z)...
Right Divert Thruster            376      0.14165D+01  -0.60545D-01  -0.18606D+00   0.00000D+00   0.00000D+00   0.00000D+00
Bottom Divert Thruster           489     -0.36855D+01  -0.23043D+00  -0.40226D+00   0.00000D+00   0.00000D+00   0.00000D+00
Left  Divert Thruster            341     -0.11363D+01  -0.10079D+00  -0.38963D+00   0.00000D+00   0.00000D+00   0.00000D+00
Top   Divert Thruster            256      0.34396D+01  -0.77800D-01  -0.32951D+00   0.00000D+00   0.00000D+00   0.00000D+00
Top Left  ACS Jet                253     -0.11059D+03   0.42843D+01  -0.23713D+01   0.00000D+00   0.00000D+00   0.00000D+00
Bottom Left  ACS Jet             345      0.13590D+03   0.41910D+01  -0.20673D+01   0.00000D+00   0.00000D+00   0.00000D+00
Bottom Right ACS Jet             342      0.14468D+03   0.41713D+01  -0.22824D+01   0.00000D+00   0.00000D+00   0.00000D+00
Top Right ACS Jet                250     -0.95398D+02   0.42728D+01  -0.21199D+01   0.00000D+00   0.00000D+00   0.00000D+00

                              Node ID#     Modal Data at the  3 Gyros ...
Sensors Location                 423      0.11237D-01  -0.23650D+01   0.52652D+01   0.5077       -6.6404       -2.9456
Sensors Location                 423      0.11237D-01  -0.23650D+01   0.52652D+01   0.5077       -6.6404       -2.9456
Sensors Location                 423      0.11237D-01  -0.23650D+01   0.52652D+01   0.5077       -6.6404       -2.9456

                              Node ID#     Modal Data at the  2 Accelerometers, along (x,y,z)...
Sensors Location                 423      0.11237D-01  -0.23650D+01   0.52652D+01
Sensors Location                 423      0.11237D-01  -0.23650D+01   0.52652D+01

                              Node ID#     Modal Data at the Disturbance Point
Oxidizer Tank                    349      0.16335D+01  -0.84384D+00   0.17223D+01   0.00000D+00   0.00000D+00   0.00000D+00
```

1.4 Batch Processing

There is a batch dataset located at the top of the input data file “*Interceptor.Inp*” that can be used to process the data faster and to create the spacecraft systems and matrices for Matlab analysis. Its title is: “*Batch for preparing models for the Flexible Interceptor Spacecraft*”. To run it, select the project directory, go to “*File Management*”, “*Managing Input Files*”, and select “*Edit/ Process Input Files*”, as shown below. From the input file manager utility dialog select the input file and click on “*Process Input Data*”.




Flixan, Flight Vehicle Modeling & Control System Analysis
Utilities
File Management
Program Functions
View Quad
Help Files
Managing Input Files (.Inp)
Managing System Files (.Qdr)
Edit / Process Batch Data Sets
Edit / Process Input Data Files

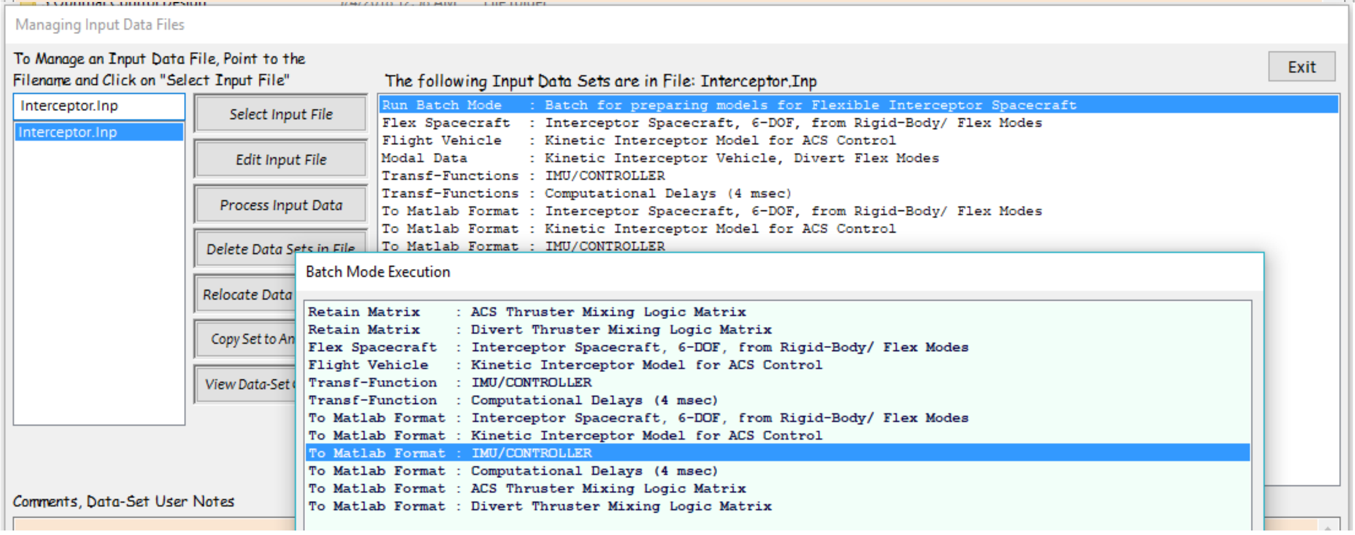

Managing Input Data Files
To Manage an Input Data File, Point to the Filename and Click on "Select Input File"
Interceptor.Inp
Interceptor.Inp
Select Input File
Edit Input File
Process Input Data
Delete Data Sets in File
Relocate Data Set in File
Copy Set to Another File
View Data-Set Comments
Exit
The following Input Data Sets are in File: Interceptor.Inp
Comments, Data-Set User Notes

```
Run Batch Mode     : Batch for preparing models for Flexible Interceptor Spacecraft
Flex Spacecraft    : Interceptor Spacecraft, 6-DOF, from Rigid-Body/ Flex Modes
Flight Vehicle     : Kinetic Interceptor Model for ACS Control
Modal Data         : Kinetic Interceptor Vehicle, Divert Flex Modes
Transf-Functions   : IMU/CONTROLLER
Transf-Functions   : Computational Delays (4 msec)
To Matlab Format   : Interceptor Spacecraft, 6-DOF, from Rigid-Body/ Flex Modes
To Matlab Format   : Kinetic Interceptor Model for ACS Control
To Matlab Format   : IMU/CONTROLLER
To Matlab Format   : Computational Delays (4 msec)
To Matlab Format   : ACS Thruster Mixing Logic Matrix
To Matlab Format   : Divert Thruster Mixing Logic Matrix
```

Replace Systems File?

The systems filename: Interceptor.Qdr already exists. Do you want to create it again?

This batch set creates Flexible Interceptor Spacecraft models


Managing Input Data Files
To Manage an Input Data File, Point to the Filename and Click on "Select Input File"
Interceptor.Inp
Interceptor.Inp
Select Input File
Edit Input File
Process Input Data
Delete Data Sets in File
Relocate Data
Copy Set to An
View Data-Set
Exit
The following Input Data Sets are in File: Interceptor.Inp
Comments, Data-Set User Notes
Batch Mode Execution

```
Run Batch Mode     : Batch for preparing models for Flexible Interceptor Spacecraft
Flex Spacecraft    : Interceptor Spacecraft, 6-DOF, from Rigid-Body/ Flex Modes
Flight Vehicle     : Kinetic Interceptor Model for ACS Control
Modal Data         : Kinetic Interceptor Vehicle, Divert Flex Modes
Transf-Functions   : IMU/CONTROLLER
Transf-Functions   : Computational Delays (4 msec)
To Matlab Format   : Interceptor Spacecraft, 6-DOF, from Rigid-Body/ Flex Modes
To Matlab Format   : Kinetic Interceptor Model for ACS Control
To Matlab Format   : IMU/CONTROLLER
```

```
Retain Matrix     : ACS Thruster Mixing Logic Matrix
Retain Matrix     : Divert Thruster Mixing Logic Matrix
Flex Spacecraft   : Interceptor Spacecraft, 6-DOF, from Rigid-Body/ Flex Modes
Flight Vehicle    : Kinetic Interceptor Model for ACS Control
Transf-Function   : IMU/CONTROLLER
Transf-Function   : Computational Delays (4 msec)
To Matlab Format  : Interceptor Spacecraft, 6-DOF, from Rigid-Body/ Flex Modes
To Matlab Format  : Kinetic Interceptor Model for ACS Control
To Matlab Format  : IMU/CONTROLLER
To Matlab Format  : Computational Delays (4 msec)
To Matlab Format  : ACS Thruster Mixing Logic Matrix
To Matlab Format  : Divert Thruster Mixing Logic Matrix
```

1.4 Dynamic Model Comparison

Figure 1.4.1 shows the two Flixan spacecraft systems in parallel. They are implemented in Simulink file "*Model_Compare.Mdl*" in subdirectory "*Examples\Interceptor Spacecraft\2 Proportional-Navigation Sim*". The top one is generated from the flexible spacecraft modeling program and it includes the system "*Interceptor Spacecraft, 6-DOF, from Rigid-Body/ Flex Modes*" which is in file "*rigbod_mod.m*", as described in Section 1.1. It consists of both: rigid and flex modes that were created from a finite elements program. The system at the bottom is generated from the flight vehicle modeling program and includes the system "*Kinetic Interceptor Model for ACS Control*" that was saved in file "*kkv_acs.m*". The inputs to the two models are scaled so an input of ±1 corresponds to maximum thrust which is ±18 (lb) for divert and ±5 (lb) for the ACS thrusters. The acceleration output of the top model is converted from (in/sec^2) to (feet/sec^2). The two models are very similar although not exact because the top system includes more details in the finite-elements implementation. This example illustrates the multiplicity of methods that can be used to model flight vehicles or spacecraft using the Flixan program.

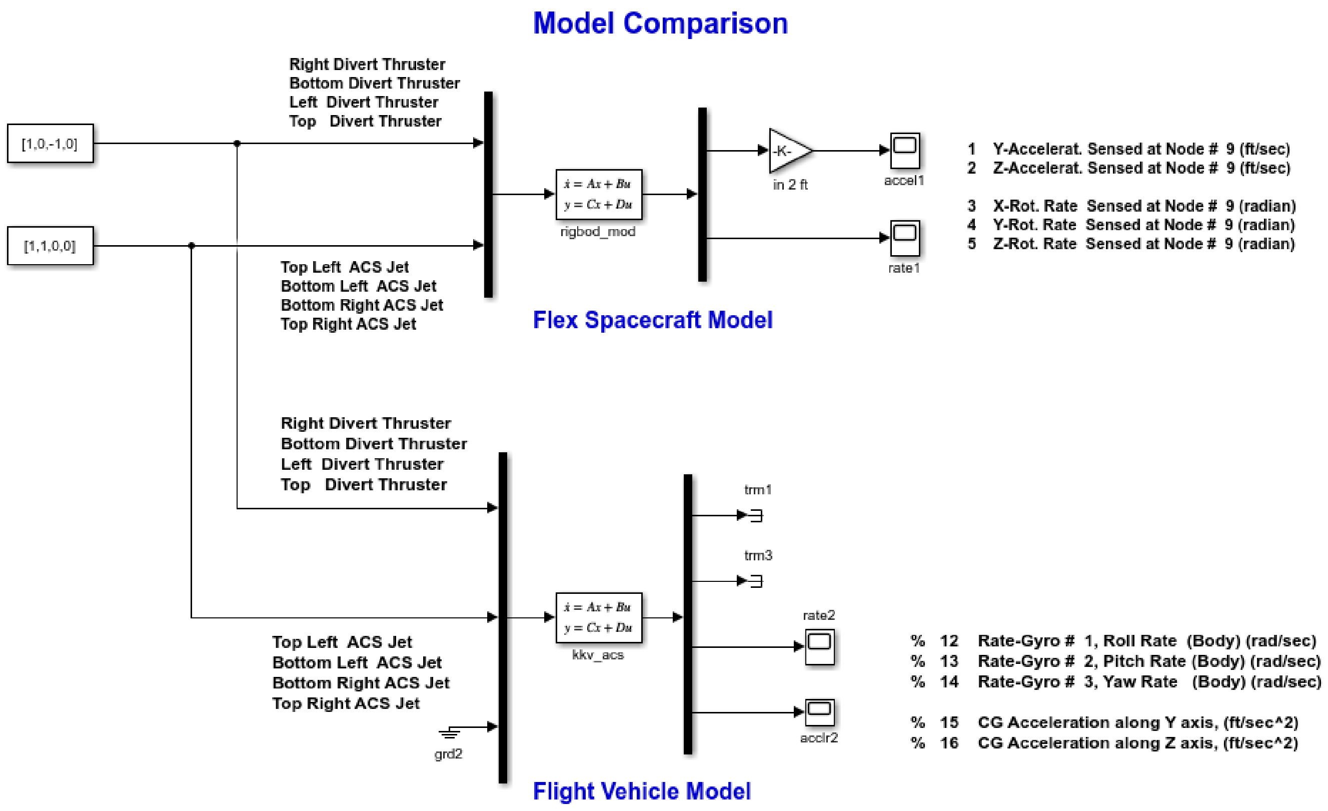


Figure 1.4.1 Comparing the Two Flixan Generated Dynamic Models of the Interceptor Spacecraft

2. Attitude Control System

Figure 2.1 shows the Attitude Control System which is a PD controller. The rate gyros receive body rates (inputs 1:3) from the spacecraft model. They are integrated by the IMU to calculate attitude outputs (4:6). Inputs (4:6) are the attitude commands in roll, pitch and yaw. Outputs (1:3) are the attitude errors. The ACS was implemented using the Flixan transfer function utility program. Its title is "*IMU/ CONTROLLER*", and it was saved in file "*control.m*" for Matlab analysis.

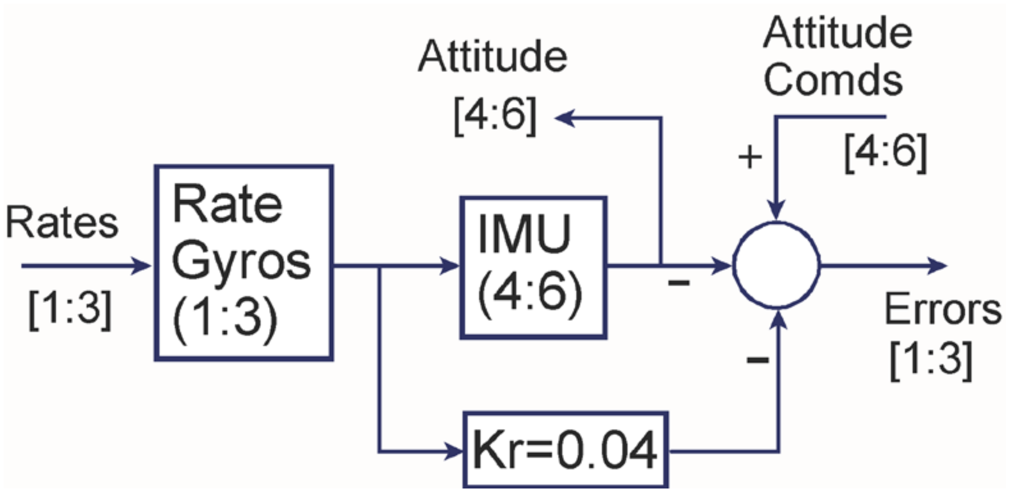


Figure 2.1 Spacecraft Attitude Control System

2.1 Mixing Logic Matrices

The systems file "*Interceptor.Qdr*" includes two mixing logic matrices for the four divert thrusters and the four ACS thrusters. The matrix T_{mix} converts the three ACS (roll, pitch, and yaw) demands to throttle variations for the four ACS thrusters. The matrix K_{div} converts the acceleration demands to throttle variations for the four divert thrusters. The matrices are saved in Matlab format and loaded into the workspace.

```
GAIN MATRIX FOR ...
ACS Thruster Mixing Logic Matrix
! Converts Roll, Pitch, Yaw Demands to RCS Throttling
!
Matrix Tmix                  Size =  4 X  3
           1-Roll               2-Pitch            3-Yaw
   1-Row  0.5                  0.5               -0.5
   2-Row -0.5                 -0.5               -0.5
   3-Row  0.5                 -0.5                0.5
   4-Row -0.5                  0.5                0.5
--------------------------------------------------------
Definitions of Matrix Inputs (Columns):    3
DP Roll  FCS Demand
DQ Pitch FCS Demand
DR Yaw   FCS Demand

Definitions of Matrix Outputs (Rows):      4
Top    Right RCS Jet
Top    Left  RCS Jet
Bottom Right RCS Jet
Bottom Left  RCS Jet
--------------------------------------------------------
```

2.2 ACS Stability Analysis

The attitude control system includes a dead-band non-linearity which controls the on-off switching of the ACS jets. Therefore, the Describing Function (DF) method will be used in the frequency domain to analyze the possibility of limit-cycling, measure the phase and gain stability margins and check if additional filter compensation may be needed to eliminate limit cycles and to improve the system performance. Limit cycling is obviously undesirable because it uses a lot of fuel. The DF of the Dead-Band non-linearity is shown in equation 2.1. It varies as a function of the input error amplitude e, and its maximum value is N_{max} is used to determine the margin

$$N(e) = \frac{4F}{\pi e}\sqrt{1-\frac{e_m^2}{e^2}}; \quad N_{\max} = \frac{2F}{\pi e_m} \tag{2.1}$$

Where: F is the jet thrust which is normalized to unity in this case because it is included in the dynamic model. e_m is the size of the dead-band (0.004 radians). The undesirable condition of limit cycling occurs when the linear part of the ACS loop $G(j\omega)$ intersects the inverse of the DF locus $\{1/N(e)\}$, that is when:

$$G(j\omega) = -\frac{1}{N(e)} \tag{2.2}$$

If we plot the inverse of the DF on a Nichol's chart 20 log{-1/ N(e)}, the intersection of the DF locus with the frequency response of the continuous system $G(j\omega)$ will indicate presence of a limit cycle. The lowest point of the -1/N(e) locus is the critical point for measuring stability along the -180° axis because it is closest to the $G(j\omega)$ locus. However, it is easier to scale the $G(j\omega)$ locus by multiplying it with N_{max} and plot it on a regular Nichols, where **+** in this case represents the minimum of the -1/ N(e) locus which is the critical point.

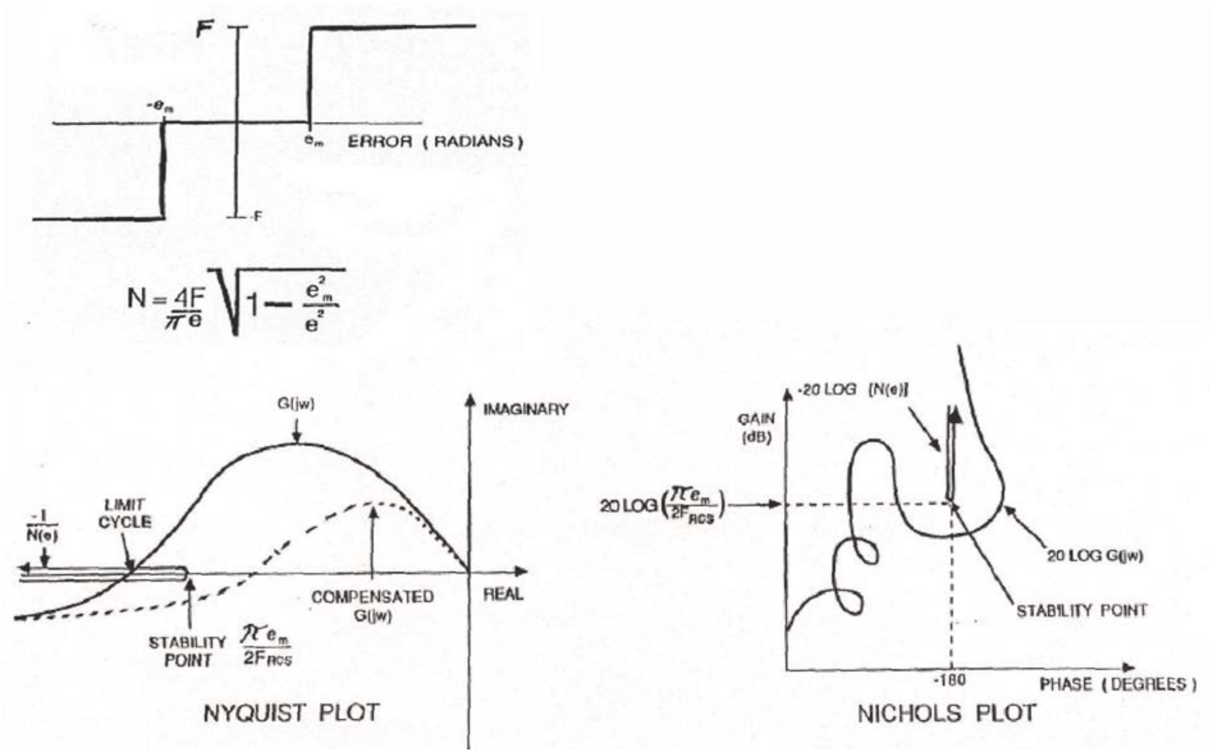


Figure 2.2 Limit-Cycle Analysis Using the Describing Function Method

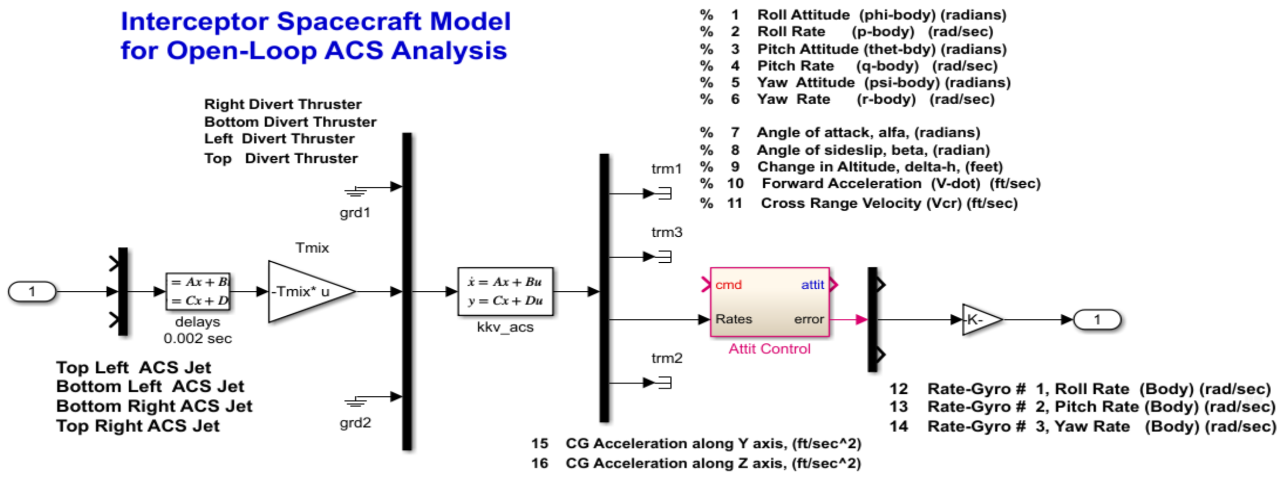
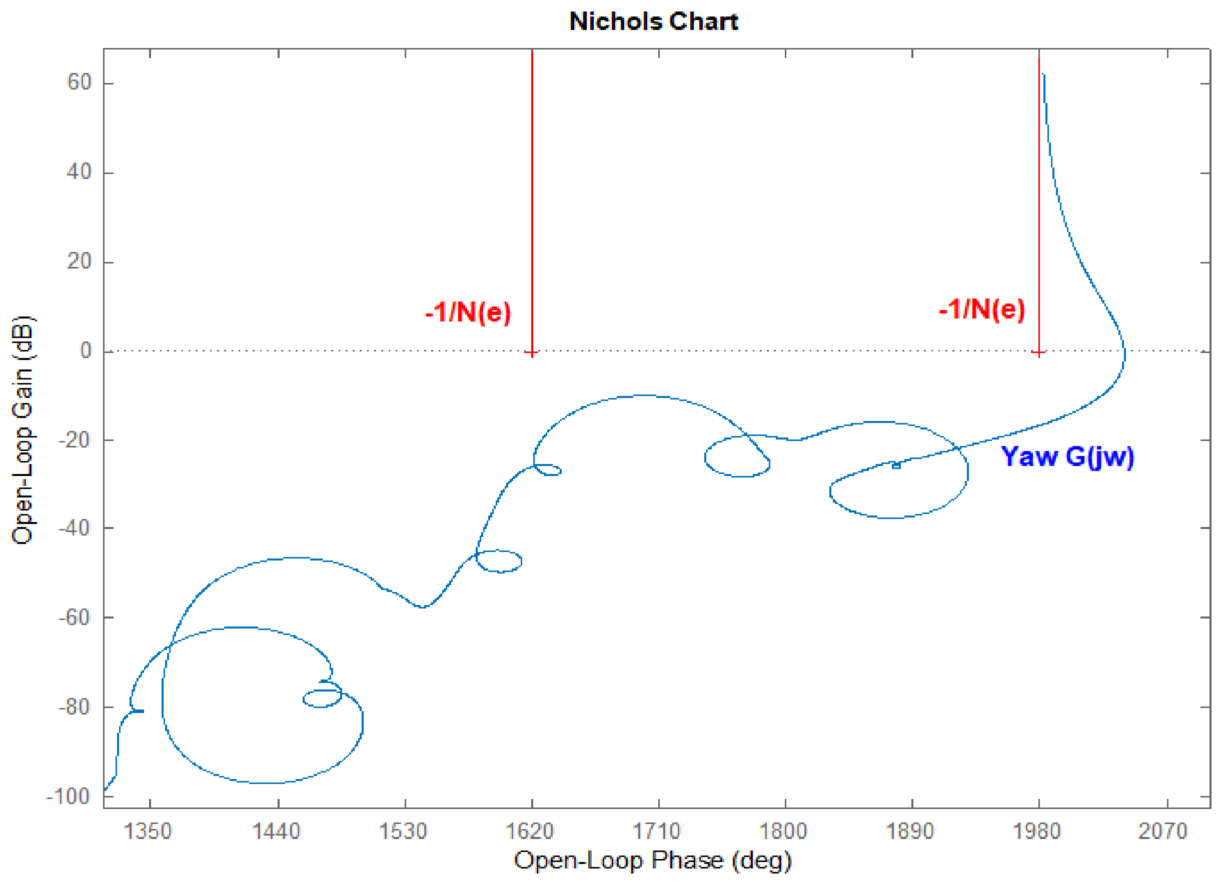

Interceptor Spacecraft Model
for Open-Loop ACS Analysis
% 1 Roll Attitude (phi-body) (radians)
% 2 Roll Rate (p-body) (rad/sec)
% 3 Pitch Attitude (thet-bdy) (radians)
% 4 Pitch Rate (q-body) (rad/sec)
% 5 Yaw Attitude (psi-body) (radians)
% 6 Yaw Rate (r-body) (rad/sec)
% 7 Angle of attack, alfa, (radians)
% 8 Angle of sideslip, beta, (radian)
% 9 Change in Altitude, delta-h, (feet)
% 10 Forward Acceleration (V-dot) (ft/sec)
% 11 Cross Range Velocity (Vcr) (ft/sec)
Right Divert Thruster
Bottom Divert Thruster
Left Divert Thruster
Top Divert Thruster
grd1
Tmix
-Tmix* u
delays
0.002 sec
kkv_acs
trm1
trm3
trm2
cmd
attit
Rates
error
Attit Control
-K-
1
Top Left ACS Jet
Bottom Left ACS Jet
Bottom Right ACS Jet
Top Right ACS Jet
grd2
12 Rate-Gyro # 1, Roll Rate (Body) (rad/sec)
13 Rate-Gyro # 2, Pitch Rate (Body) (rad/sec)
14 Rate-Gyro # 3, Yaw Rate (Body) (rad/sec)
15 CG Acceleration along Y axis, (ft/sec^2)
16 CG Acceleration along Z axis, (ft/sec^2)

Figure 2.3 Open-Loop Model "Open_Loop.mdl" for ACS Stability Analysis Using the Describing Function Method

The rate feedback gain Kr in the ACS block diagram in Figure 2.1 determines the amount of phase-lead and the attenuation of the G(jω) locus. It can be used to adjust the gain and phase margins relative to the critical point +. The open-loop model "*Open_Loop.mdl*" in subdirectory "*Interceptor Spacecraft\1 ACS Stability Analysis*" is used to analyze stability in the frequency domain. It includes the Flixan generated flex spacecraft model "*kkv_acs.m*". The file "*start.m*" loads the models and matrices, and the file "*freq.m*" calculates the Nichols plots, as shown below in Figure 2.4.



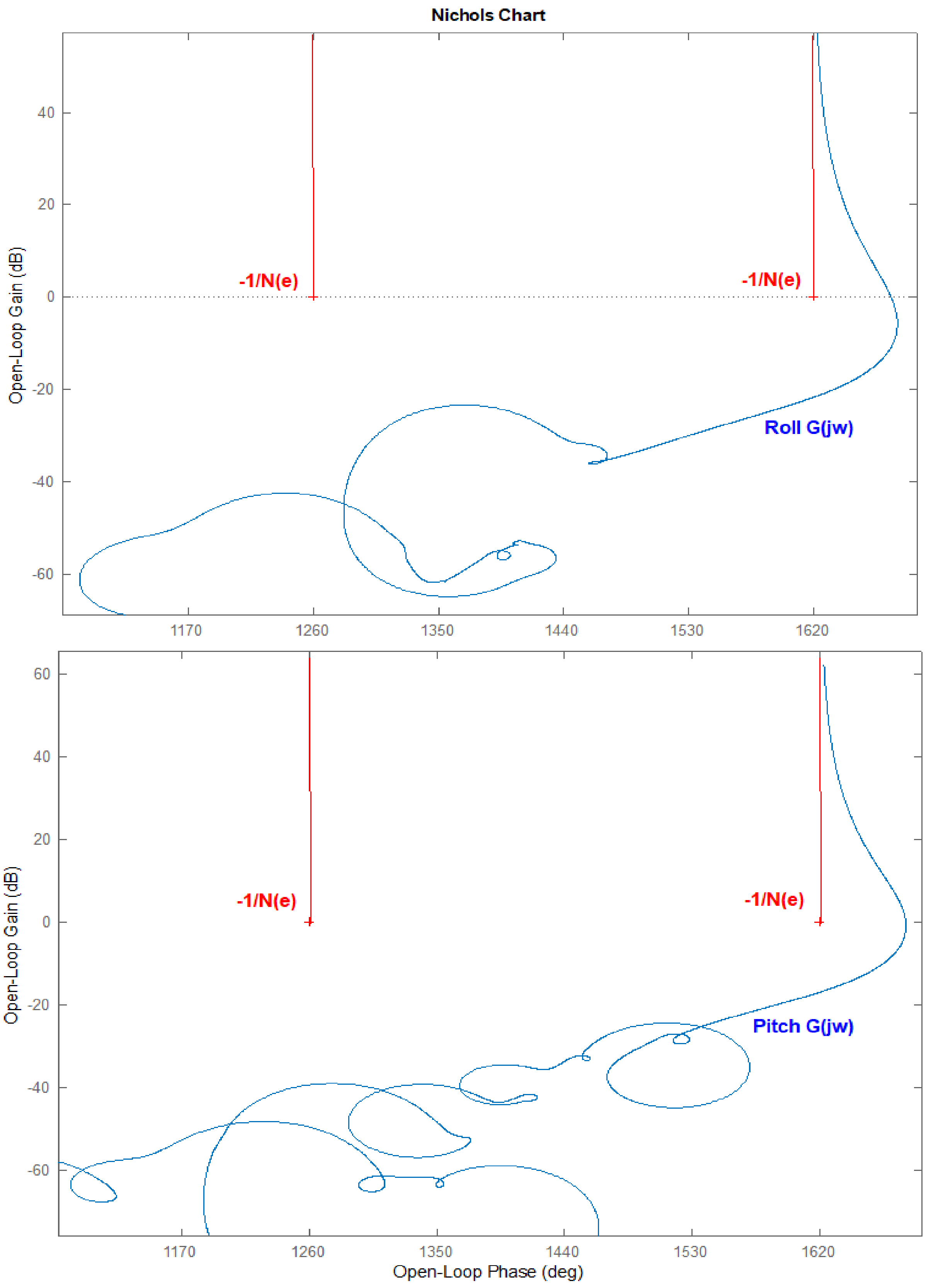


Figure 2.4 Nichols Plots Showing the ACS Stability Margins in Roll, Pitch and Yaw Axes

3. Linear Acceleration Control System

The End-Game control law obviously assumes that the interceptor acceleration response is analog and proportional to the continuous command from guidance. Continuous acceleration, however, cannot be obtained from the on/off thrusters and for the algorithm to be successful in intercepting the target it is necessary that the KV is able to respond efficiently to the acceleration commands. A control system is included, therefore, that regulates the average acceleration of the interceptor and makes it equal to the commanded acceleration.

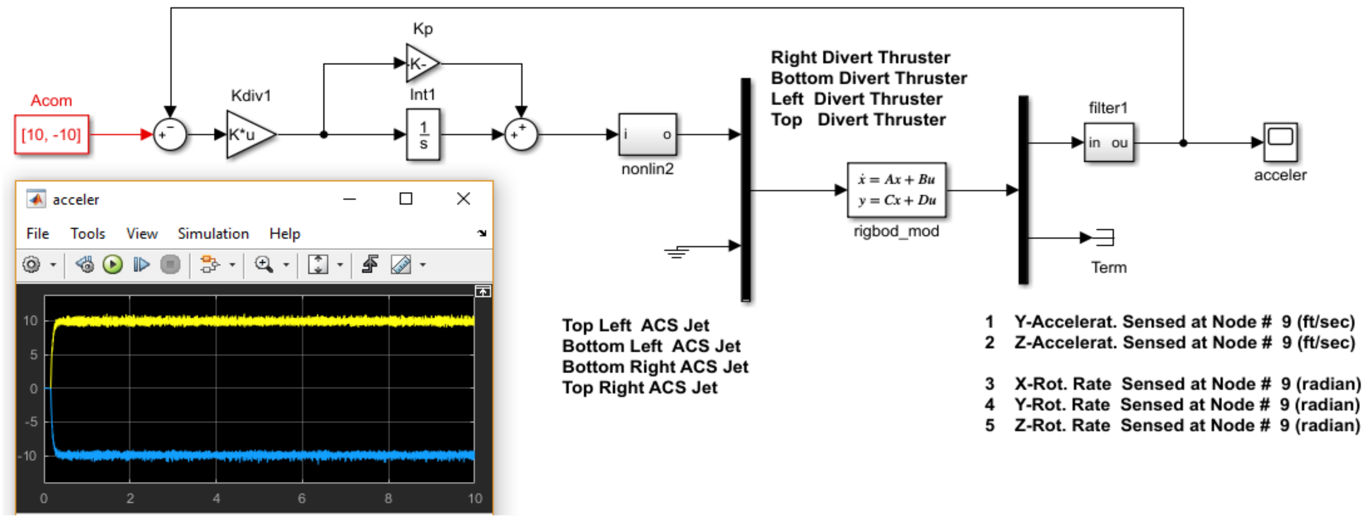


Figure 3.1 Linear Acceleration Control System

The linear acceleration control system is shown in Figure 3.1. It consists of a PI integrator that integrates the acceleration error signal (A_{com} - A_k) and feeds it to a dead-band non-linearity. The matrix K_{div} converts the two acceleration demands (along y and z) to throttle demands. The PD output regulates the switching of divert thrusters that accelerate the KV in the proper direction. It acts like a pulse width modulator and controls the spacecraft acceleration in the y and z directions by regulating the thruster firing. The accelerometer output is filtered to reduce flex mode excitation.

```
GAIN MATRIX FOR ...
Divert Thruster Mixing Logic Matrix
! Converts Ay and Az Demands to Divert Throttling
!
Matrix Kdiv                    Size =  4 X  2
           1-Ay                   2-Az
   1-Row -0.5                    0.0
   2-Row  0.0                   -0.5
   3-Row  0.5                    0.0
   4-Row  0.0                    0.5
--------------------------------------------------
Definitions of Matrix Inputs (Columns):     2
Y-Acceleration Demand
Z-Acceleration Demand

Definitions of Matrix Outputs (Rows):       4
Right  Divert Thruster -Y
Bottom Divert Thruster -Z
Left   Divert Thruster +Y
Top    Divert Thruster +Z
--------------------------------------------------
```

The Simulink model shown in Figure 3.1, in file "*Accel_Control.mdl*", is located in subdirectory "*Interceptor Spacecraft\2 Proportional-Navigation Sim*" and it is used to analyze the acceleration control system. In this example the spacecraft is commanded to accelerate ±10 (feet/sec^2) in the y and z directions and it responds with a small delay. The noise is due to flexibility which is also excited by the thrusters firing. The accelerometer signal is filtered to minimize flex mode sensing and excitation. The size of the dead-band determines the pulsing frequency. A narrow band will generate more frequent pulsing and it will regulate the average acceleration better than a wider one. It is important for the system to respond fast to the acceleration commands especially during the final phase of the End-Game in order to drive the relative state vector to zero. However, when the dead-band is too narrow it causes undesirable limit cycles which is fuel inefficient.

4. End-Game Dynamic Model

The dynamic model in this Section describes the relative motion between the interceptor spacecraft and the target. The relative motion of the two spacecraft can be described by two sets of equations: one that describes the relative motion along the x-direction which is along the main velocity direction and the LOS, and another set of equations that describe the motion perpendicular to the LOS. The motion along the LOS which is along the line joining the two spacecraft is uncontrollable because the interceptor has no thrust in the x-direction and the relative motion equation is only used to calculate the time to impact (t_{go}). The relative motion perpendicular to the LOS along the y and z directions is controlled by the interceptor's divert thrusters and it is identical in both perpendicular directions.

The time-to-go calculation t_{go} for an accelerating target in equation 4.1 is calculated from the estimated acceleration A_x, relative velocity V_x, and the distance to target R. If the target is not accelerating the time-to-go simplifies to: $t_{go} = R/V_x$

$$t_{go} = \frac{-V_x + \sqrt{V_x^2 - 2RA_x}}{A_x} \qquad (4.1)$$

Equation 4.2 describes the relative spacecraft motion perpendicular to the LOS (either y or z directions). It is controlled by the divert thrusters that provide the KV acceleration A_I. It describes the motion in the local inertial frame which is defined by the position of the interceptor at the initialization time, when it first locates the target.

$$\begin{pmatrix} \dot{S}_r \\ \dot{V}_r \\ \dot{A}_T \\ \dot{A}_K \end{pmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & -W_T & 0 \\ 0 & 0 & 0 & -W_I \end{bmatrix} \begin{pmatrix} S_r \\ V_r \\ A_T \\ A_I \end{pmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & W_T \\ W_I & 0 \end{bmatrix} \begin{pmatrix} A_{I\,com} \\ A_{T\,com} \end{pmatrix} \qquad (4.2)$$

The state vector consists of four states:

S_r	is the relative vehicle position (target – interceptor)
V_r	is the relative vehicle velocity (target – interceptor)
A_T	is the target acceleration normal to the LOS
A_I	is the interceptor acceleration perpendicular to the LOS

The two inputs are:

A_{Icom}	Interceptor Acceleration Command perpendicular to the LOS
A_{Tcom}	Target Acceleration Command perpendicular to the LOS

The dynamic model in equation 4.2 includes the maneuverability of the two spacecraft perpendicular to the LOS and introduces it in the control design. In addition to relative position and velocity it includes the target and interceptor bandwidths described by first order lags of frequencies W_T and W_I respectively, where: $W_T=5$ (rad/sec) and $W_I=100$ (rad/sec). Naturally the interceptor must have a broader bandwidth than the target because it is smaller in size. S_r and V_r are the target's position and velocity relative to the kill vehicle perpendicular to the LOS.

The control guidance of the interceptor must sense the relative motion of the target along the ±Y and ±Z KV axes using the optical sensor and apply the proper acceleration command to the thrusters that will null-out the relative position and will intercept the incoming target at the estimated impact time. The state-feedback gains are not constant but they increase as the time-to-go before impact decreases, which make the interceptor respond faster to errors as the distance gets shorter. The relative target to interceptor position is measured by the on-board seeker. The seeker makes azimuth and elevation measurements relative to the line-of-sight (LOS). The measurements are:

$$\begin{pmatrix} az \\ el \end{pmatrix} = \begin{bmatrix} \tan^{-1}\left(S_{ry} \middle/ S_{rx} \right) \\ \tan^{-1}\left(-S_{rz} \middle/ \sqrt{S_{rx}^2 + S_{ry}^2} \right) \end{bmatrix} + \begin{pmatrix} v_{az} \\ v_{el} \end{pmatrix} \tag{4.3}$$

Where: (S_{rx}, S_{ry}, S_{rz}) are the relative target to interceptor positions in the KV body coordinate frame and (v_{az}, v_{el}) are zero mean white measurement noise. The target's inertial acceleration is estimated and it is transformed in the KV body frame using the transformation matrix C_I^B. The Euler angles ϕ(t), θ(t), and ψ(t) are obtained from the KV rotations relative to the local inertial frame. The local inertial frame is the initial attitude of the KV when the End-Game begins at t=0. That is, when $\phi(t)=\theta(t)=\psi(t)=0$. The inertia to body transformation matrix is

$$C_I^B = \begin{bmatrix} 1 & \psi(t) & -\theta(t) \\ -\psi(t) & 1 & \phi(t) \\ \theta(t) & -\phi(t) & 1 \end{bmatrix} \tag{4.4}$$

The Kalman-Filter estimates the state vector from the position and accelerometer seeker measurements. The estimated state consists of relative position and velocity, target and interceptor accelerations, and it must be properly initialized in the beginning of the engagement. The ACS is continuously rotating the KV, pointing the LOS towards the target, and trying to null out the target's deviations from the center-view. The rotation angles and rates are converted to relative position and velocity errors (Sr and Vr) which command the KV acceleration in the proper direction.

5. Proportional Navigation

Proportional Navigation (PN) is an end-game guidance that calculates the acceleration command in the y and z spacecraft axes. It is a function of the closing velocity V_c, the rate of change of the LOS angle assuming the KV is tracking the target, and the estimated target acceleration $E(A_t)$.

$$A_{com} = N V_c \frac{d(LOS)}{dt} + E(A_t) \qquad (5.1)$$

Equation 5.1 can also be written in the following form

$$A_{com} = \frac{N S_r}{t_{go}^2} + \frac{N V_r}{t_{go}} + E(A_t) \qquad (5.2)$$

Where: Sr and Vr are the relative position and velocity perpendicular to the LOS. N is a constant that can be adjusted, typically between 3 and 4. The time-to-go for an accelerating target is calculated from Equation 4.1. The PN has the advantage that it does not require accurate range measurements but only LOS rate from the seeker and an approximate estimate of the closing motion for the t_{go} calculation. If initialized properly it will converge to zero relative position when t_{go}=0. Its main disadvantage is sensitivity to seeker noise when the target is very far away and the LOS rate is small. It causes the interceptor to be chasing noise and using up a lot of fuel.

5.1 Proportional Navigation Simulation

Figure 5.1 shows a simple PN simulation model. It does not include the state estimator which calculates relative position, velocity and acceleration from the range and seeker measurements. In this example we integrate the relative acceleration in the body frame to calculate the relative velocity and miss distance.

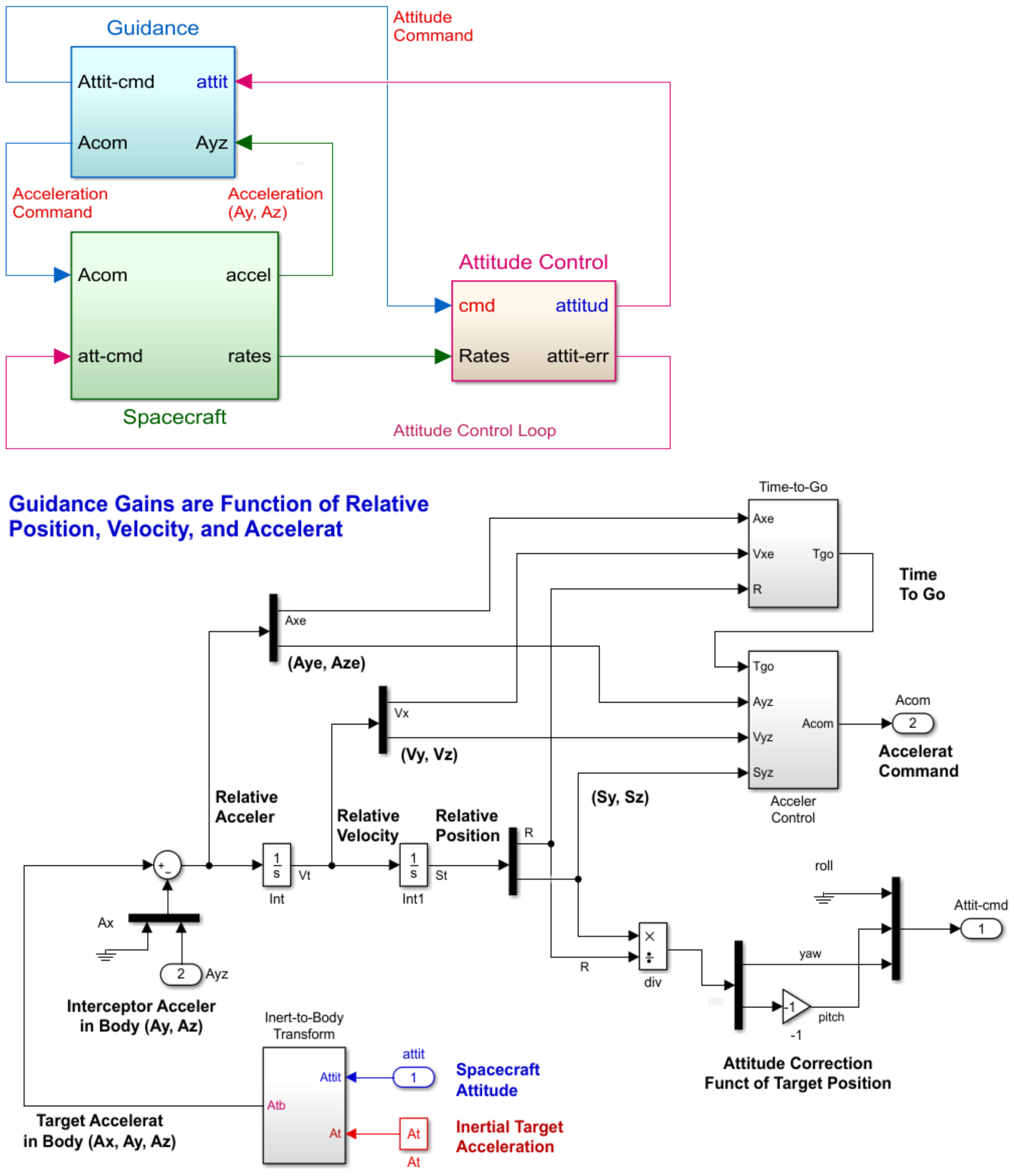


Figure 5.1 End-Game Simulation using Proportional Navigation

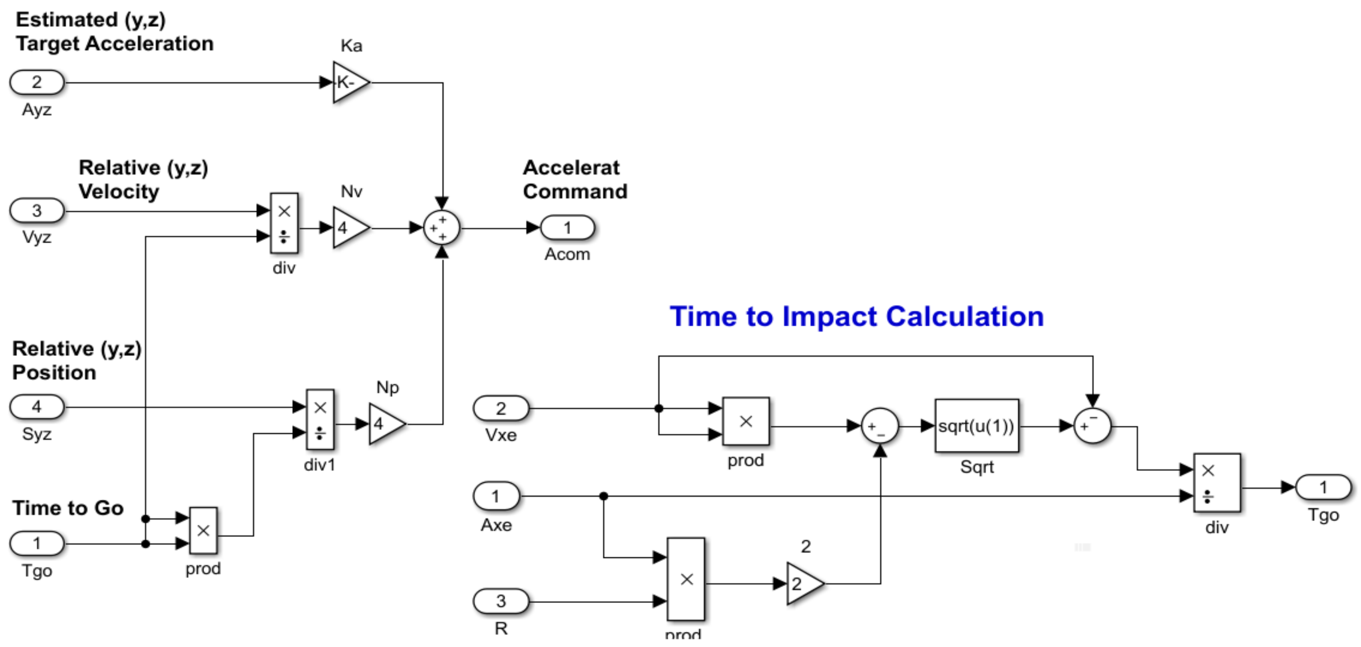


Figure 5.2 Proportional Navigation Guidance

The inertial target acceleration A_T is transformed to KV body axes and the relative acceleration between the two bodies is constructed by subtracting the KV acceleration. In the absence of an estimator, the relative acceleration is integrated twice to calculate relative velocity V_t and relative position S_t. The initialization file “*start.m*” loads the Flixan generated dynamic models and initializes the relative position, velocity and acceleration states in all 3 directions. From the relative state variables and the time-to-go calculation, the proportional navigation algorithm calculates the KV acceleration command $A_{com.}$ This command drives the acceleration control system which fires the appropriate divert thrusters to produce the necessary acceleration intended to bring the final relative position to zero at t_{go}=0.

5.2 PN Simulation Results

Figure 5.3 shows the simulation results obtained from the PN simulation model which is implemented in file “*Sim_6dof.mdl*” in subdirectory “*Examples\Interceptor Spacecraft\2 Proportional-Navigation Sim*”. The spacecraft subsystem uses the Flixan derived spacecraft model “*rigbod_mod.m*” which includes rigid and flex modes. There is also a similar simulation model in file “*Sim_6dof3.mdl*” that uses the spacecraft model “*kkv_acs.m*” that was created using the flight vehicle modeling program. The two systems were previously compared in Figure 1.4.1 using the Simulink file “*Model_Compare.Mdl*”. The simulations are initialized from file “*start.m*” which includes also the relative position and velocity of the two bodies at t=0. The target also has a constant (x,y,z) acceleration. The file Pl.m plots the results as a function of the calculated t_{go}.

The first plot shows how the relative position changes from its initial values in the axial and the two normal directions as a function of t_{go}. It is reduced to zero at impact when $t_{go}=0$, in all 3 directions. The velocity along the LOS is increasing because the target is accelerating towards the KV at 10 (feet/sec^2). The relative velocities and accelerations, however, normal to LOS are also increasing as the KV approaches the target. This is not a desirable condition. The normal A_y and A_z accelerations are noisy due to the thruster firings. The throttle commands to the thrusters are either zero or ±1 because the state-space models are scaled and include the actual thrusts. Throttle inputs of ±1 imply ±Thrust$_{\_Max}$. The results from the second simulation model in file "*Sim_6dof3.mdl*" are almost identical.

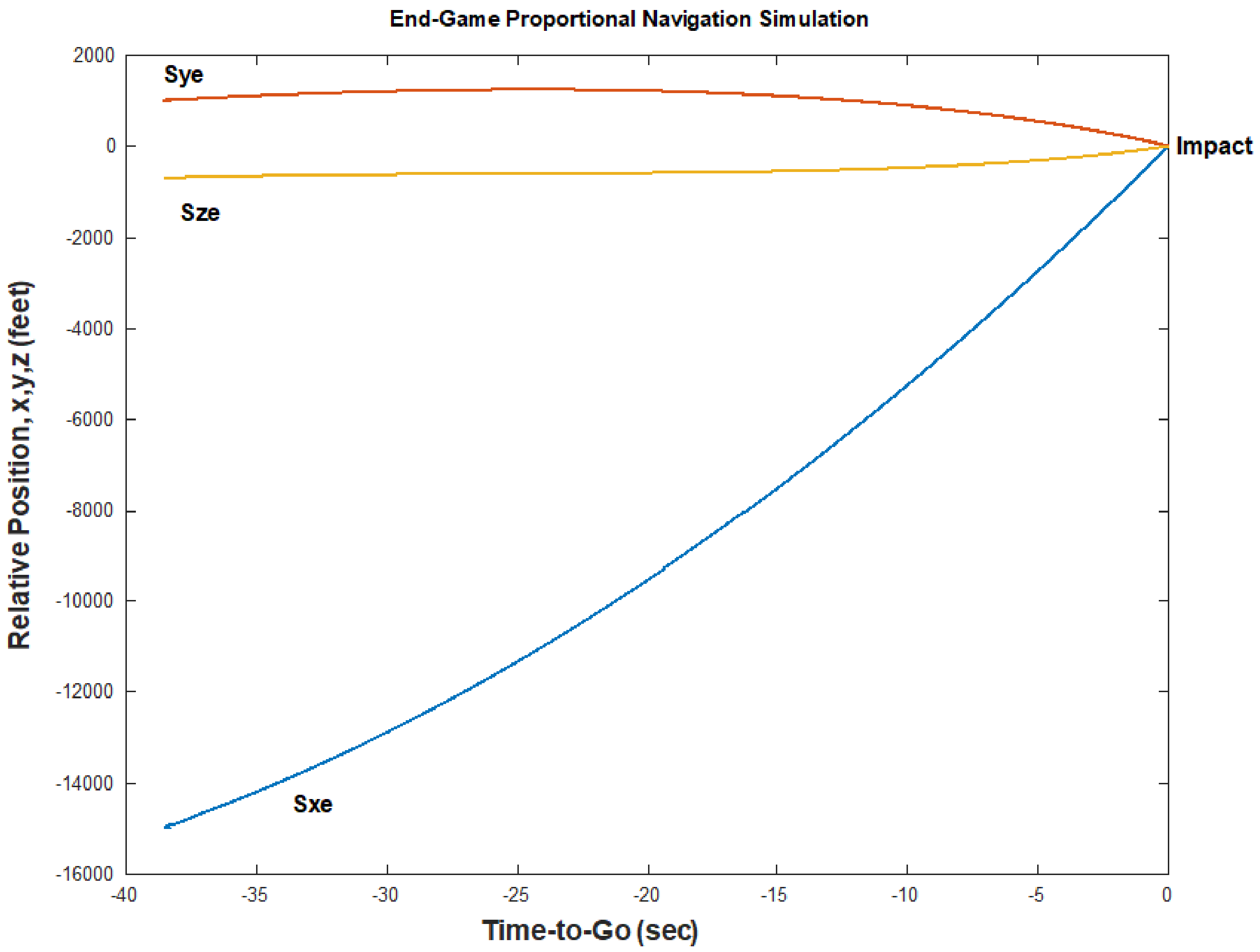
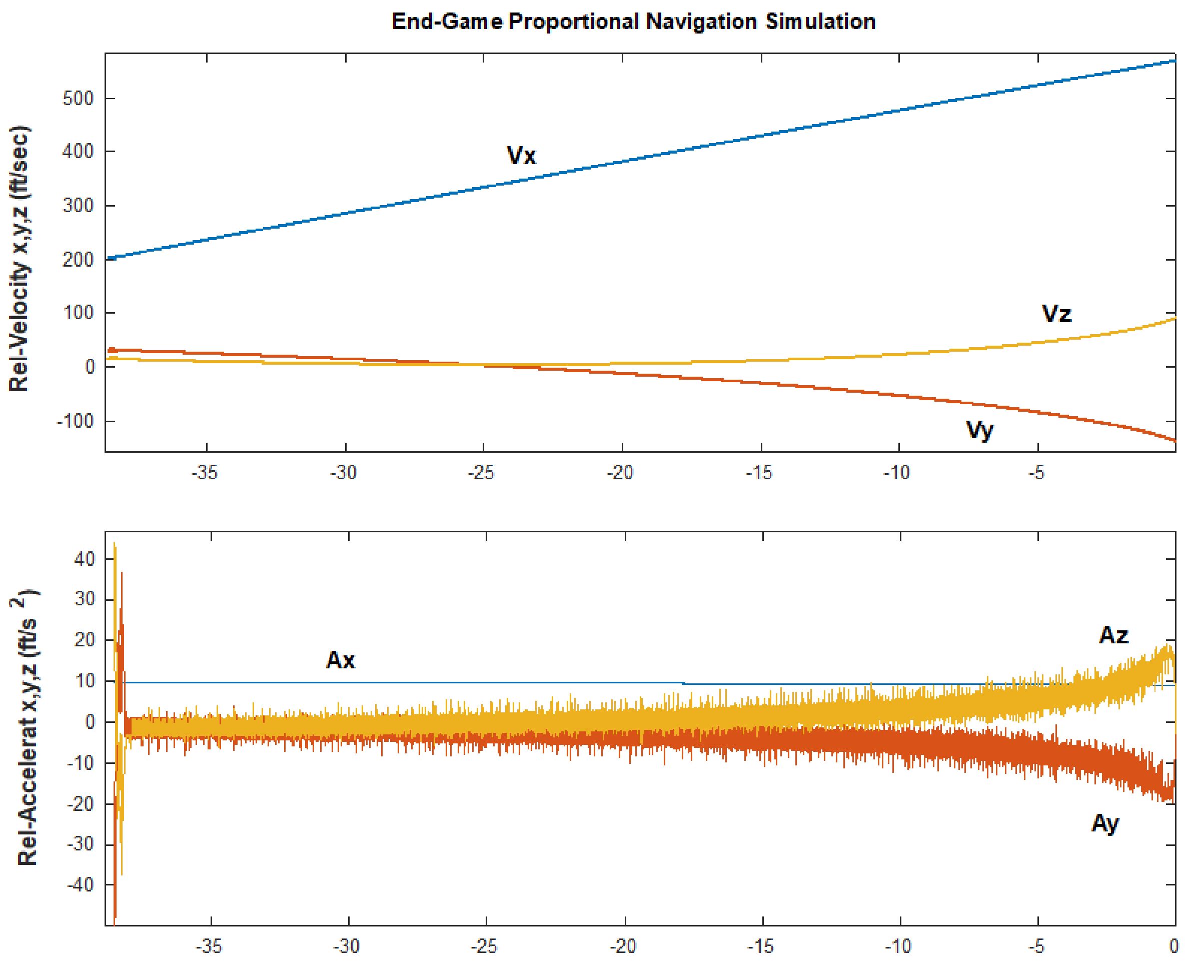
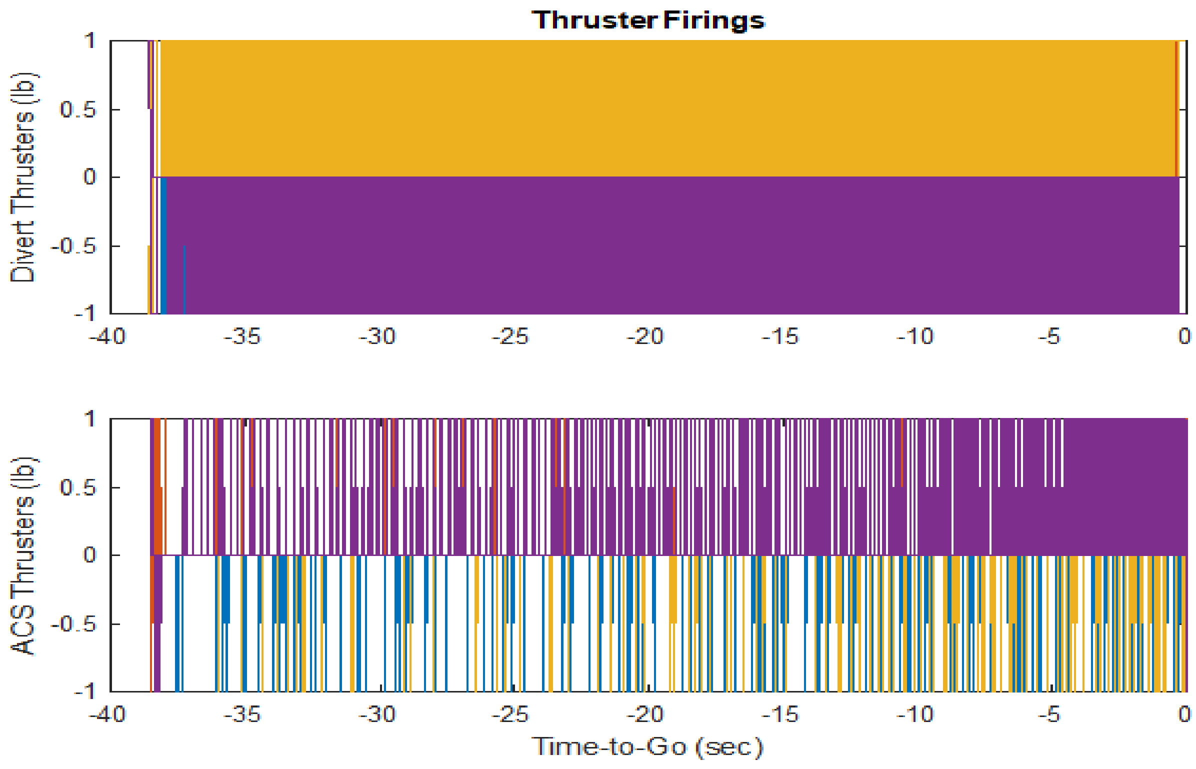


Figure 5.3 Simulation Results Using Proportional Navigation


End-Game Proportional Navigation Simulation
Rel-Velocity x,y,z (ft/sec)
Vx
Vz
Vy
Rel-Accelerat x,y,z (ft/s 2)
Ax
Az
Ay


Thruster Firings
Divert Thrusters (lb)
ACS Thrusters (lb)
Time-to-Go (sec)

6. Optimal Control Guidance

The optimal control guidance uses the Linear Quadratic Regulator solution that calculates a time-varying state-feedback control law that will take out the relative position error at the estimated impact time. It takes into consideration two important issues/ requirements. The first issue is that there is a significant amount of noise in the measurement, especially when the distance from the target is large. We don't want the kill vehicle to be chasing noise because it will consume its propellant fast. Therefore, we need a control system with variable bandwidth. Starting at low bandwidth for fuel efficiency and increasing it inversely proportional with time-to-go in order to improve performance near impact, where it is needed most and the signal to noise ratio is good. The second issue to be considered in the design is the uncertainty in the t_{go} calculation which is based on navigation measurements and subject to delays. We want a successful hit even if it occurs a little sooner or a little later than expected. One way to improve success is to reduce the relative side velocity V_r to zero a short time prior to the estimated impact time, and maintain a high gain system until impact or switch to PN.

6.1 Transient LQR Optimal Control Design

The previously described design requirements can be captured in the performance index J, equation 6.1 that can be optimized by the Transient Linear Quadratic Regulator algorithm. The index J is based on the End-Game dynamic model of Equation 4.2 that describes the relative motion in a direction perpendicular to the LOS, either y or z axis, and at this point we are assuming that all states are available for state feedback.

$$\dot{\underline{x}} = A\underline{x} + Bu$$
$$J = \int_0^{t_f} \left(\underline{x}' Q \, \underline{x + R u^2} \right) dt + \underline{x}(t_f)' P_1 \, \underline{x}(t_f) \qquad (6.1)$$

Where: the matrices Q and P_1 and R, are weights that trade control acceleration versus system performance to disturbances.

Q	is a positive semidefinite matrix that penalizes the state error along the trajectory
R	is a scalar that penalizes the control along the trajectory which is related to fuel
P_1	is a positive semidefinite matrix that penalizes the state vector error only at the final time t_f

They are selected to achieve a satisfactory trade-off between fuel consumption and robustness to miss distance errors, in the presence of seeker noise and range measurement errors. Since we cannot control the target motion but only the interceptor's, for control design we ignore the second input to the dynamic model and keep only the A_{Icom} input. The target acceleration input will be used to inject noise in the analysis.

During the early part of the trajectory where the solution is steady-state, we avoid penalizing much the position and velocity errors perpendicular to the x-axis in the Q matrix. The last term in the performance index equation that includes the matrix P_1 produces the time-varying state-feedback gain. The terminal position and velocity states are heavily penalized in matrix P_1. By adjusting the velocity coefficient we can reduce the terminal velocity V_r to almost zero at impact, that is, in addition to the relative position S_r. Reducing the perpendicular components of the relative velocity V_r near zero at impact makes the optimal control law less sensitive to range errors. The LQR solution in equation 6.2 calculates a time varying state-feedback gain matrix $K_c(t)$ that satisfies the design requirements by optimizing the performance index defined in equation 6.1.

$$\begin{aligned} &u^0(t) = -R^{-1} B^T P(t)\,\underline{x}(t) = -K_c(t)\,\underline{x}(t) \\ &-\dot{P}(t) = Q - P(t)\,B\,R^{-1}\,B^T\,P(t) + P(t)\,A + A^T P(t) \\ &where\text{: } 0 < t < t_f \end{aligned} \qquad (6.2)$$

The time-varying matrix P(t) is always positive definite and symmetric and is obtained by solving the transient Riccati equation 6.2b. Its terminal value at impact is equal to the value of the terminal state weight matrix P_1, i.e. $P(t_f)=P_1$. This property is used for solving the Riccati equation numerically after initializing it at the terminal time t_f and integrating backwards in time to t=0. The resulting control law is a time varying state-feedback that provides normal acceleration to the interceptor as a function of the four states, which at this point we assume that they are all available for feedback. In our next step we will design a Kalman-Filter to estimate the states from the system output. The same control law is used for both: the y and z axes, since the spacecraft is symmetric and there is no coupling between the y and z directions. It is important to mention that the final time t_f is not necessarily the impact time but it may be a time before impact that we wish to switch control laws, to Proportional Navigation, for example. The terminal goal in this case may be to achieve favorable conditions for PN initialization.

The optimal control design boils down to choosing a satisfactory trade-off between two scalars in equation 6.3: the fuel weight R and the terminal state weight p. A large R penalizes the fuel usage. The larger p gets, the more the final normal relative position and velocity are reduced close zero at impact. This, however, is achieved at the expense of propellant consumption or that the control demand may exceed the maximum acceleration capability of the interceptor's thrusters. T_{gi} is a short period before impact when you expect the position and velocity to converge to zero.

$$\begin{aligned} &Q = diag\begin{pmatrix}0.1 & 0.1 & 0 & 0.1\end{pmatrix};\; R = 1 \\ &P_1 = p\begin{bmatrix} 1 & T_{gi}/2 & 0 & 0 \\ T_{gi}/2 & T_{gi}^2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{aligned} \qquad (6.3)$$

6.2 Optimal Control Design and Simulations

We will now use Flixan to design the time-varying control system and analyze it using Matlab simulations. The analysis files for this example are located in folder: "*Examples\Interceptor Spacecraft\3 Optimal Control Design*". The directory includes the input file "*End_Game_s.Inp*", shown below, that contains input data for the continuous LQR design. That is, for the steady-state LQR control, the time-varying state-feedback LQR, the Kalman-Filter, and for the steady-state output-feedback LQG design that combines the steady-state LQR gain K_c and the Kalman-Filter gain K_f to a dynamic output-feedback controller. The input file also includes a batch set for fast data processing, and datasets for Matlab conversions. The systems filename "*End_Game_s.Qdr*" contains the End-Game dynamic model described in Equation 4.2, the control design matrices: Q_c, R_c, and P_1, the Kalman-Filter design matrices Q_{mn}, R_{mn}, the control gain K_c and the Kalman-Filter gain K_f which are generated by the Flixan LQR design program.

```
BATCH MODE INSTRUCTIONS ...............
Batch for preparing End-Game Control design Models
! This batch Generates LQR State-Feedback, Kalman-Filter and Output
! Feedback Dynamic Controller
Retain System    : Simple End-Game Model
Retain Matrix    : State Weight Matrix Qc (4x4)
Retain Matrix    : Output Weight Matrix Qc (2x2)
Retain Matrix    : Control Weight Matrix Rc
Retain Matrix    : Terminal State Weight Matrix P1 (4x4)
Retain Matrix    : Performance Criteria C1
Retain Matrix    : Output Performance Weight Matrix Qc3
Retain Matrix    : Measurement Noise Matrix Rmn (2x2)
Retain Matrix    : Process Noise Matrix Qpn (4x4)
!
LQR Control Des  : LQR Control Design for Simple End-Game Model
State Estimator  : Kalman-Filter Design for Simple End-Game Model
LQG Control Des  : LQG Control Design for Simple End-Game Model
Transient LQR    : Transient LQR Design for Simple End-Game Model
!
To Matlab Format : Simple End-Game Model
To Matlab Format : LQG Control Design for Simple End-Game Model
To Matlab Format : LQR State-Feedback Control for Simple End-Game Model
To Matlab Format : Kalman-Filter Estimator for Simple End-Game Model
---------------------------------------------------------------------------------------------------------
LINEAR QUADRATIC REGULATOR STATE-FEEDBACK CONTROL DESIGN
LQR Control Design for Simple End-Game Model
! Design the State-Feedback Matrix Kc using the Output
! Criteria Matrix C= Identity
!
Plant Model Used to Design the Control System from:       Simple End-Game Model
Criteria Optimization Output is Matrix Identity
State Penalty Weight (Qc) is Matrix:   Qc4                State Weight Matrix Qc (4x4)
Control Penalty Weight (Rc) is Matrix: Rc                 Control Weight Matrix Rc
Continuous LQR Solution Using Laub Method
LQR State-Feedback Control Gain Matrix Kc                 LQR State-Feedback Control for Simple End-
---------------------------------------------------------------------------------------------------------
KALMAN-BUCY FILTER STATE ESTIMATOR DESIGN
Kalman-Filter Design for Simple End-Game Model
! Design the Kalman-Filter Gain Matrix Kf using the
! Process Noise Matrix G = Identity
!
Plant Model Used to Design the Kalman-Filter from:        Simple End-Game Model
Input Process Noise Matrix (G) is the  Identity
Process Noise Covariance Qpn is Matrix Qpn                Process Noise Matrix Qpn (4x4)
Measurement Noise Covariance is Matrix Rmn                Measurement Noise Matrix Rmn (2x2)
Kalman-Filter Estimator is Gain Matrix Kf                 Kalman-Filter Estimator for Simple End-Game
---------------------------------------------------------------------------------------------------------
```

```
DYNAMIC OUTPUT FEEDBACK LQG CONTROL DESIGN
LQG Control Design for Simple End-Game Model
! Combine State-Feedback with KF Gain to Design a Linear Quadratic
! Gaussian Control System for the Plant: Simple End-Game Model
!
Plant Model Used to Design the Control System from:        Simple End-Game Model
State-Feedback (Kc) is Gain Matrix   : Kc                  LQR State-Feedback Control for Simple End-
Kalman-Filter Estim Kf is Gain Matrix: Kf                  Kalman-Filter Estimator for Simple End-Game
----------------------------------------------------------------------------------------------------------
TRANSIENT LQR CONTROL DESIGN WITH TIME-VARYING GAINS
Transient LQR Design for Simple End-Game Model
! To Generate Time-Varying State-Feedback Gain Kc(t)
! as a Function of Time-to-Go, in File: Gains.dat
!
Plant Model Used to Design the Control System from:        Simple End-Game Model
Criteria Optimization Output is Matrix Identity
State Penalty Weight (Qc) is Matrix:   Qc4                 State Weight Matrix Qc (4x4)
Control Penalty Weight (Rc) is Matrix: Rc                  Control Weight Matrix Rc
Terminal State Penalty Weigh P1 Matrix P1                  Terminal State Weight Matrix P1 (4x4)
Continuous LQR Solution, Final Time, Number of Points:     20.00     800.0
----------------------------------------------------------------------------------------------------------
CONVERT TO MATLAB FORMAT ........      (Title, System/Matrix, m-filename)
Simple End-Game Model
System
end_game
----------------------------------------------------------------------------------------------------------
CONVERT TO MATLAB FORMAT ........      (Title, System/Matrix, m-filename)
LQG Control Design for Simple End-Game Model
System
Control
----------------------------------------------------------------------------------------------------------
CONVERT TO MATLAB FORMAT ........      (Title, System/Matrix, m-filename)
LQR State-Feedback Control for Simple End-Game Model
Matrix Kc
----------------------------------------------------------------------------------------------------------
CONVERT TO MATLAB FORMAT ........      (Title, System/Matrix, m-filename)
Kalman-Filter Estimator for Simple End-Game Model
Matrix Kf
----------------------------------------------------------------------------------------------------------
```

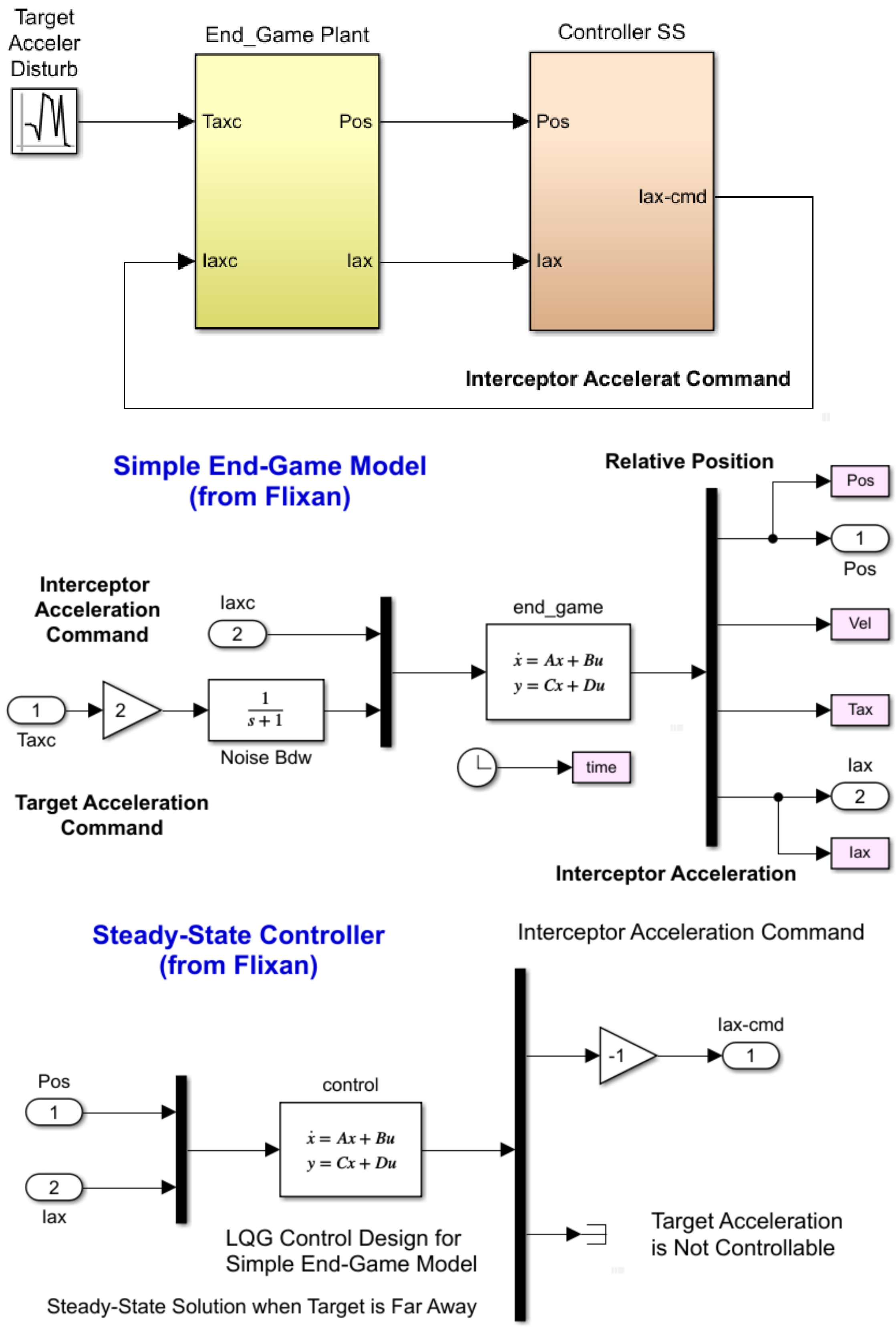
The Flixan LQR transient design program also calculates the time varying state-feedback gain $K_c(t)$ as a function of time-to-go. It requires the weight matrices: Q_c, R_c, and P_1, the initial time-to-go (20 sec), and the number of gain calculation points (800). The gains versus time are saved in file "Gains.dat" with the time-to-go in the first column. This file is used as a look-up table in the simulations. Only the first 4 gains that correspond to the interceptor acceleration command are used in this case. The end-game dynamic model is saved in file "end_game.m", and the steady-state output-feedback control system is saved in "control.m" for Matlab analysis. The gain matrices Kc and Kf are also saved and loaded into Matlab.

6.2.1 Simulation Models

We will first analyze the steady state-performance of the spacecraft and then the transient motion with time varying state-state gains. Two simulation models were created to analyze the system's response from an initial relative condition of position, velocity and acceleration.

6.2.2 Steady-State Simulation

The steady-state analysis is applicable when the target is sufficiently far away from the interceptor and the control gains are constant. The Simulink model is "*EndGame_Sim1s.mdl*", shown in Figure 6.2, and it is located in the same "*Examples\Interceptor Spacecraft\3 Optimal Control Design*" directory.



Target Acceler Disturb
End_Game Plant
Taxc
Pos
Iaxc
Iax
Controller SS
Pos
Iax-cmd
Iax
Interceptor Accelerat Command
Simple End-Game Model (from Flixan)
Relative Position
Pos
1
Pos
Interceptor Acceleration Command
Iaxc
2
end_game
$\dot{x} = Ax + Bu$
$y = Cx + Du$
Vel
1
Taxc
2
$\frac{1}{s+1}$
Noise Bdw
Tax
time
Iax
2
Iax
Target Acceleration Command
Interceptor Acceleration
Steady-State Controller (from Flixan)
Interceptor Acceleration Command
Iax-cmd
-1
1
Pos
1
control
$\dot{x} = Ax + Bu$
$y = Cx + Du$
2
Iax
LQG Control Design for Simple End-Game Model
Target Acceleration is Not Controllable
Steady-State Solution when Target is Far Away

Figure 6.2 Steady-State Simulation Model “EndGame_Sim1s.mdl”

The dynamic model is the Flixan generated system "*Simple End-Game Model*" in file "*end_game.m*". The Flixan generated steady-state control system is "*LQG Control Design for Simple End-Game Model*" in file "*Control.m*". It is also dynamic and it includes parameters from the plant model, the steady-state feedback gain K_c and the Kalman-Filter gain K_f, as shown in Figure 6.3. The two systems and matrices are loaded into Matlab by executing the file "init.m". This script also initializes the state-vector $\underline{x}_0$. Notice, that this configuration cannot be used in the time-varying case because Kc(t) is varying.

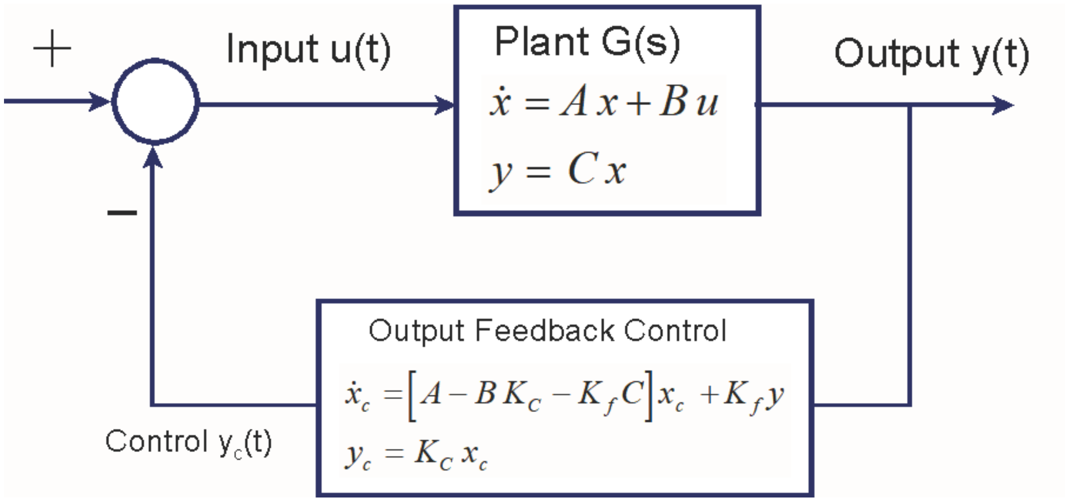


Figure 6.3 Steady-State LQG Output Feedback Controller/ Plant Interconnection

We can use this simulation model to calculate the system's response to an accelerating target disturbance, as shown in Figure 6.4.

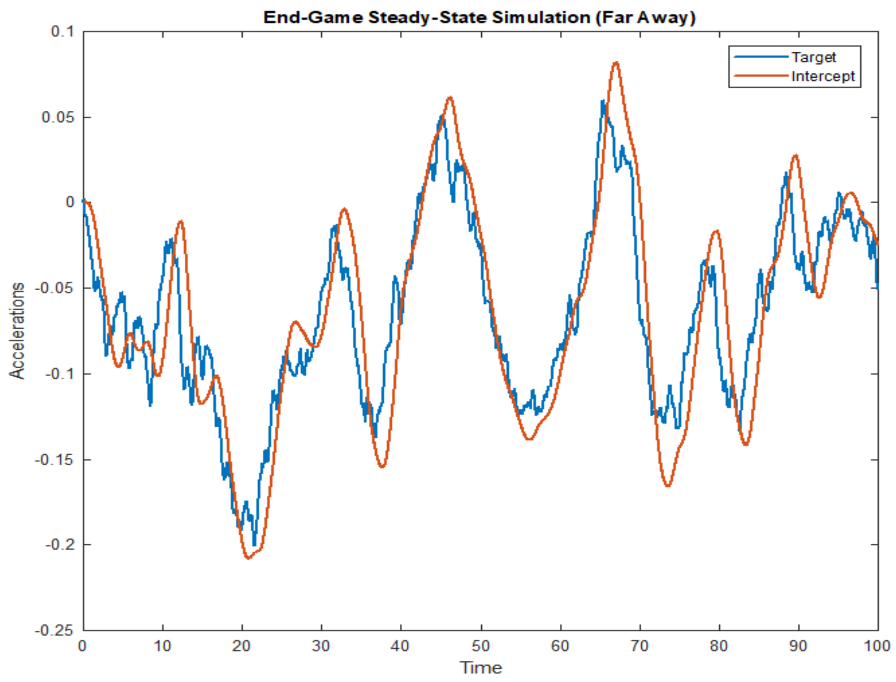


Figure 6.4 Interceptor Acceleration is responding to Noisy Target Accelerations

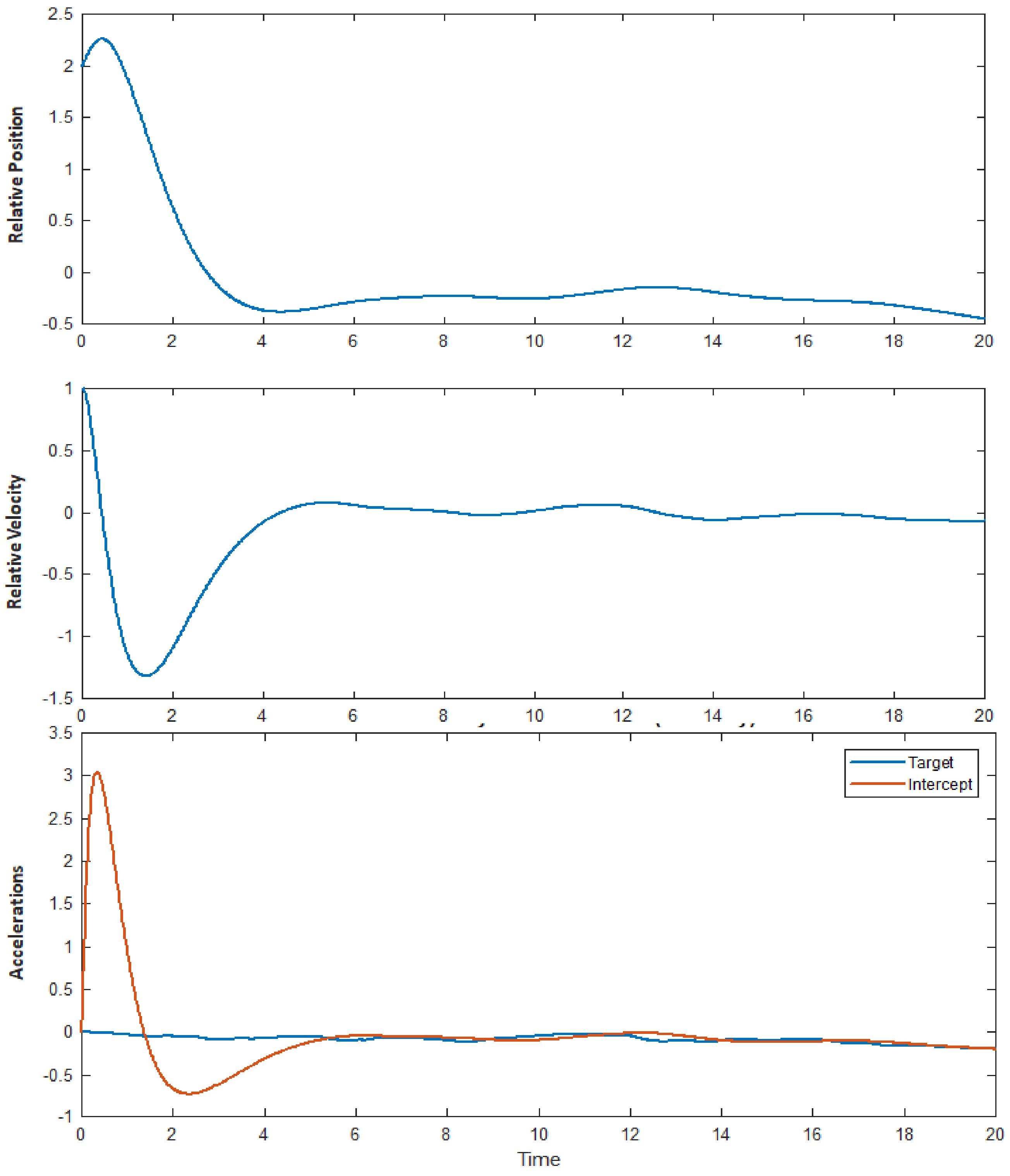
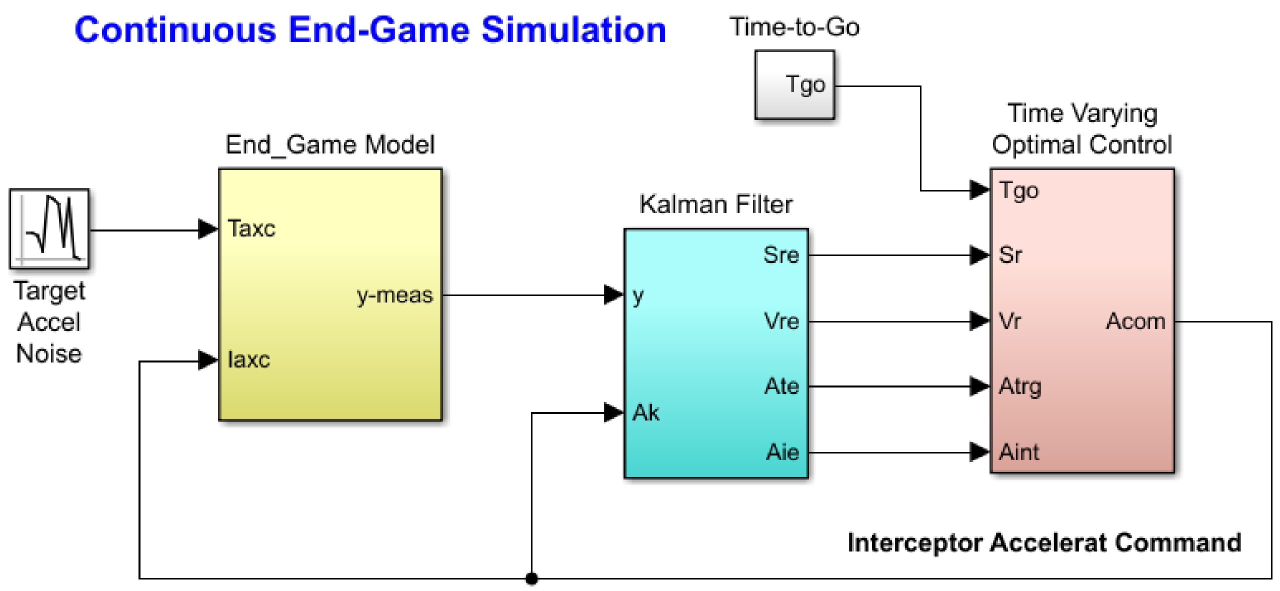


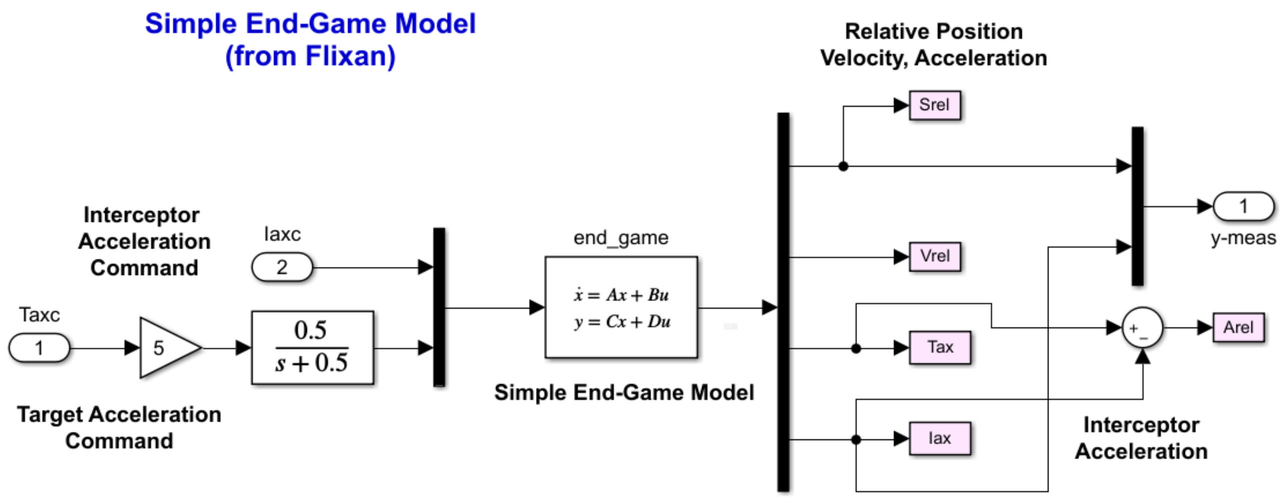
Figure 2.5 System's Response from Non-Zero Initial Conditions of Relative Position and Velocity

This simulation model is also used to calculate the system's response to initial position and velocity errors while tracking an accelerating target, see Figure 6.5. Notice that the system's response at steady-state is intentionally slow in order to save propellant because the target is still far away. This causes noticeable position error due to target acceleration.

6.2.3 Simulation with Time Varying Control Gains

The simulation model in Figure 6.6 is implemented in file “*EndGame_Sim2s.mdl*”. The control gains are time-varying as a function of time-to-go. They were calculated by the Flixan Transient LQR program as already mentioned and are loaded into Matlab look-up tables from file “Gains.dat”. The t_{go} is calculated from the relative axial position, velocity and acceleration and used to look-up the gains from the table. The gains increase as t_{go} gets shorter producing an exponentially increasing control bandwidth.





The Kalman-Filter and the control system are separate subsystems in this simulation. The Kalman-Filter is now used to estimate the four state variables from the relative position and the accelerometer measurements. The estimated states are multiplied with the four time varying gains to produce the KV acceleration command. The gains are functions of t_{go} which is calculated from the relative x-axis acceleration, velocity and range to go.

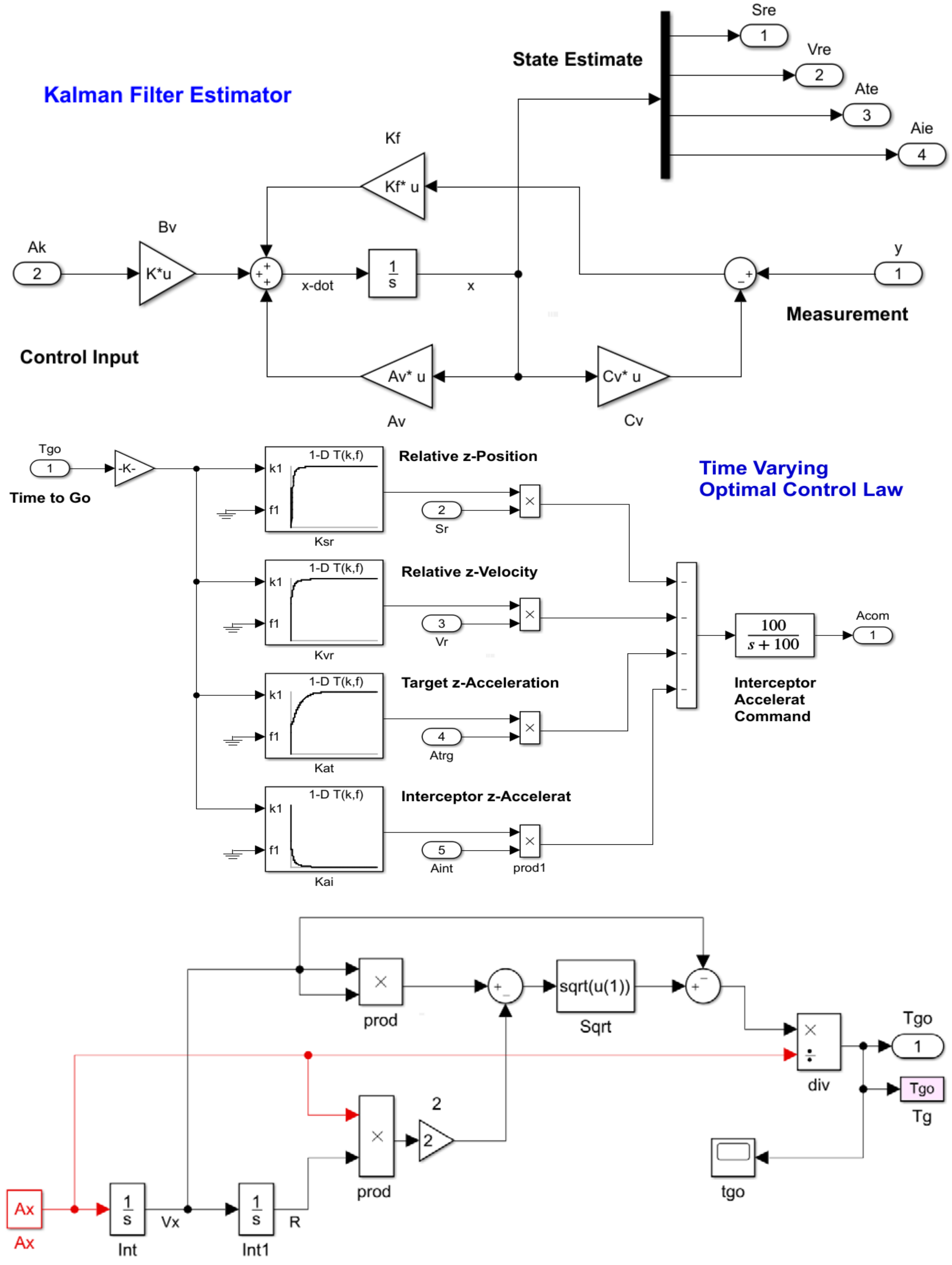


Figure 6.6 Simulation Model “EndGame_Sim2s.mdl” that uses Time-Varying Gains

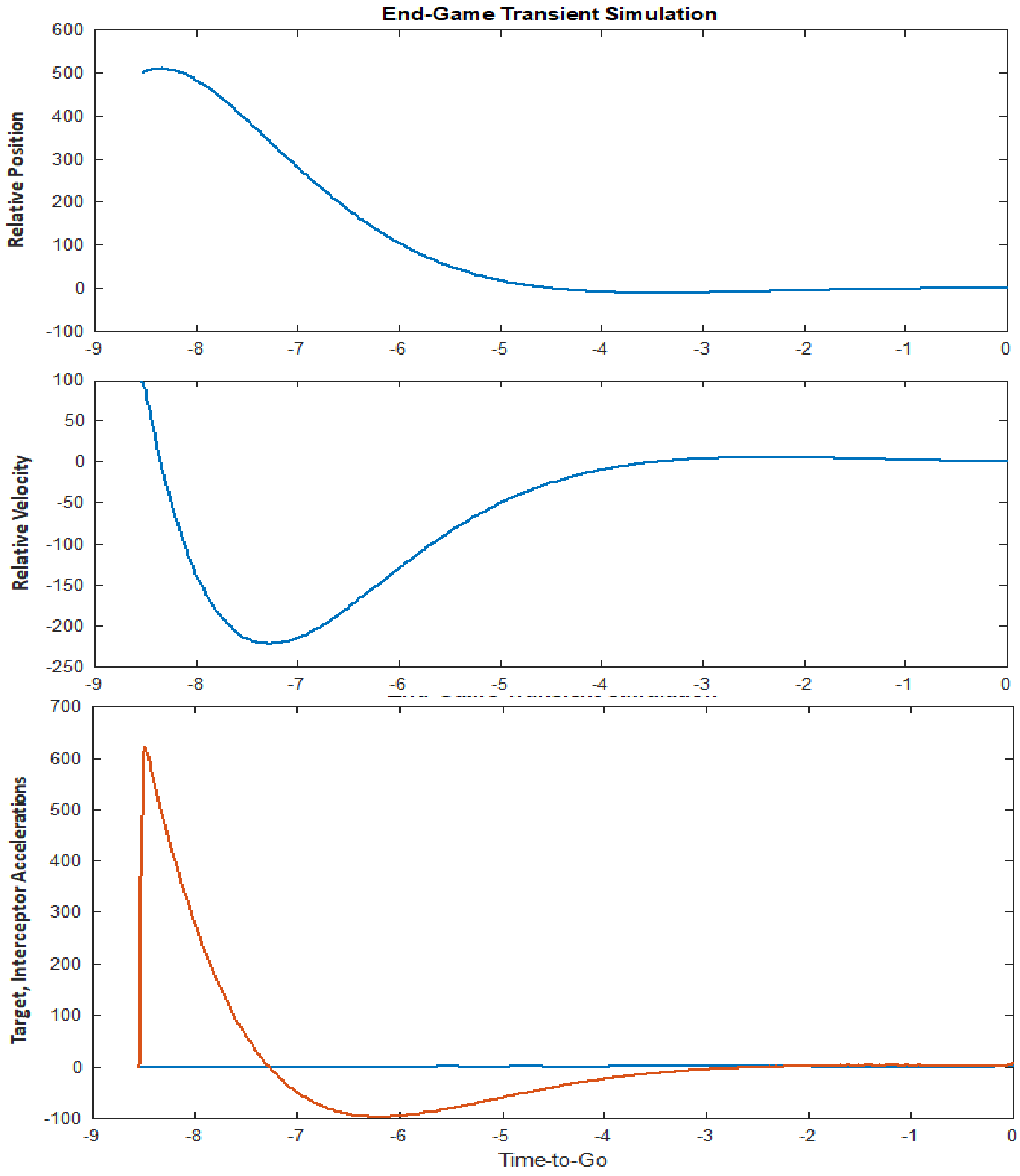


Figure 6.7 System's Response from Non-Zero Initial Conditions of Relative Position and Velocity

Figure 6.7 shows the response of the time-varying control system to non-zero initial conditions versus time-to-go, beginning 8.5 sec before impact when the gains are still at steady state. Beginning with initial position and velocity errors 500 (feet) and 100 (ft/sec) respectively, the interceptor (orange) accelerates in order to bring the final position and velocity errors very close to zero at impact.

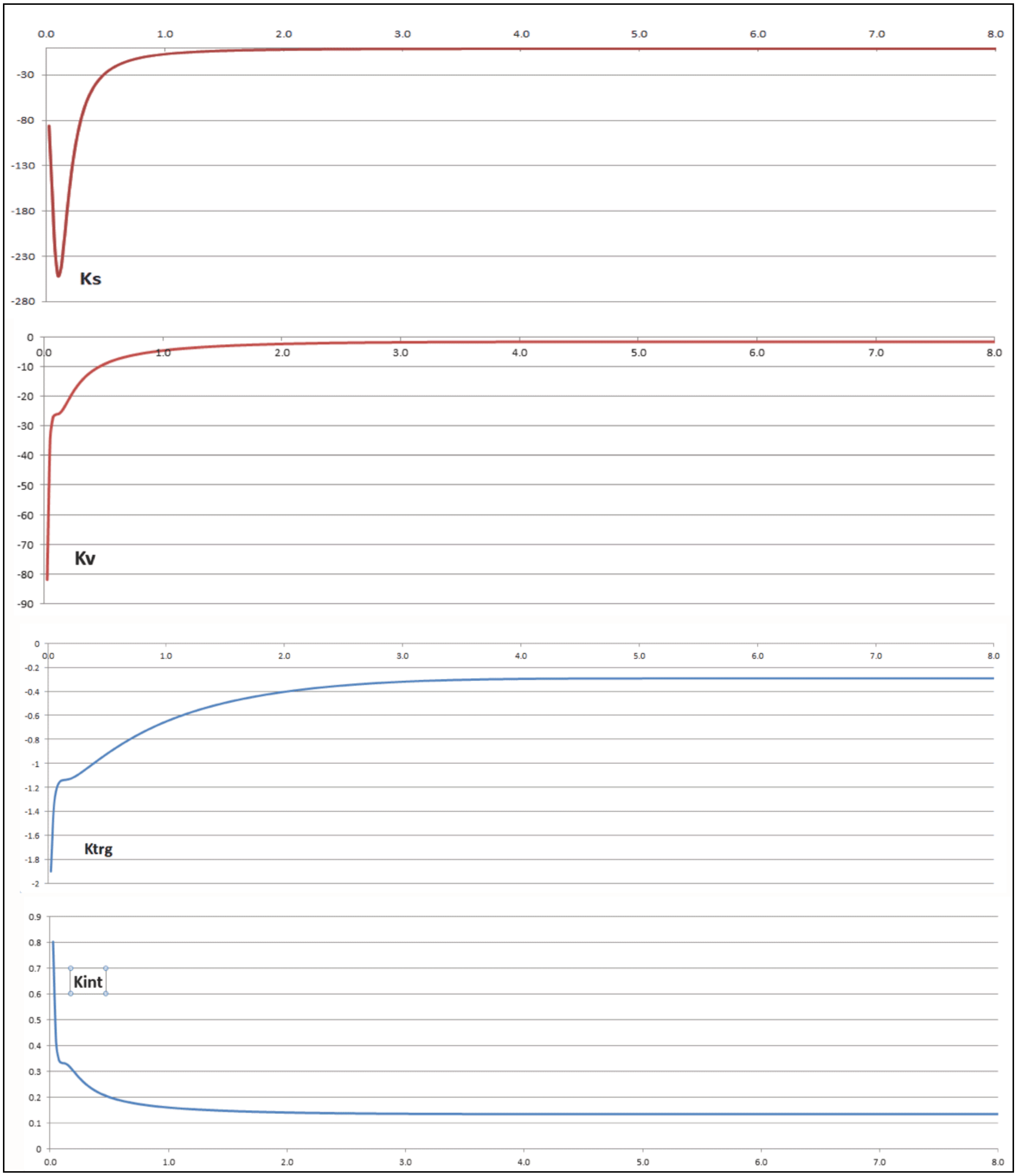


Figure 6.8 Optimal Control State-Feedback Gains as a Function of Time-to-Go

6.2 Optimal Control Law Simulation

The models "*Sim_6dof.mdl*" and "*Sim_6dof2.mdl*" in folder "*Interceptor Spacecraft\4 Optimal Control Sim*", shown in Figure 6.9, are used to implement and simulate the time-varying optimal control law using the already calculated state-feedback gains "gains.dat" as a function of time-to-go. They are similar to the simulations in Section 5.1, and are using the two Flixan generated flex spacecraft models "*rigbod_mod.m*" and "*kkv_acs.m*" respectively.

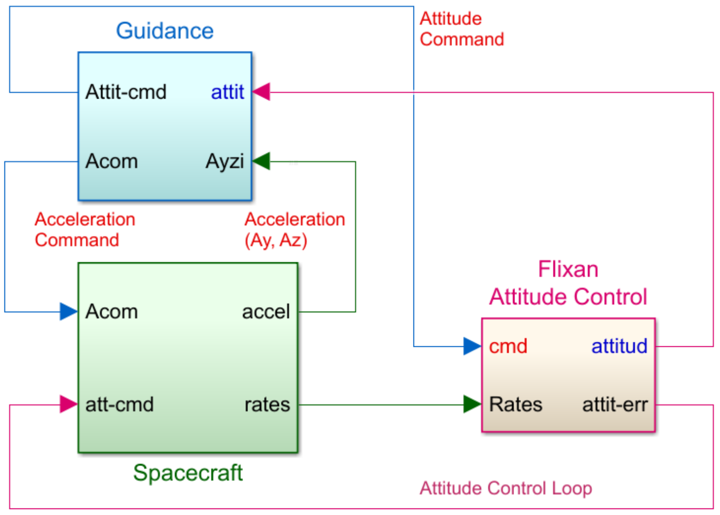


Figure 6.9 Optimal Control Simulation Model in file "Sim_6dof.mdl"

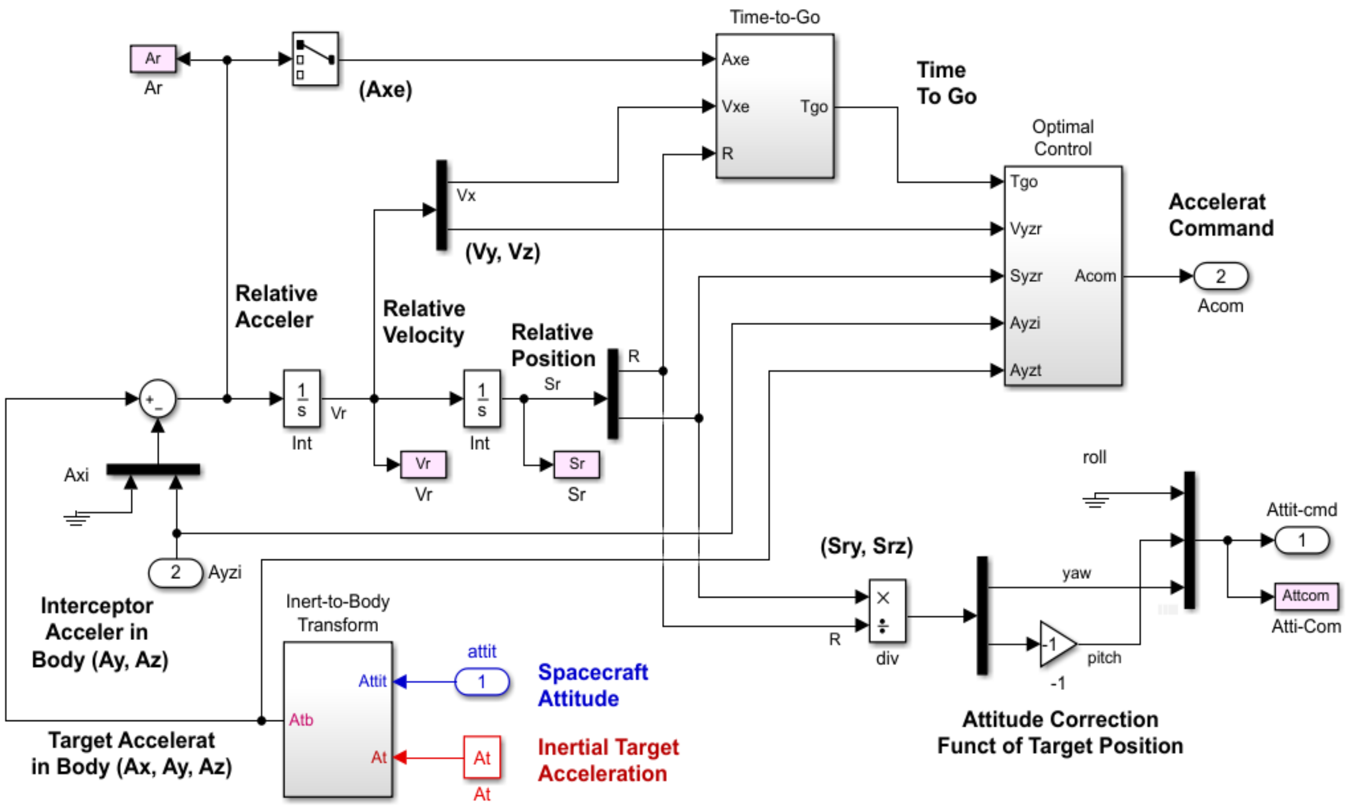
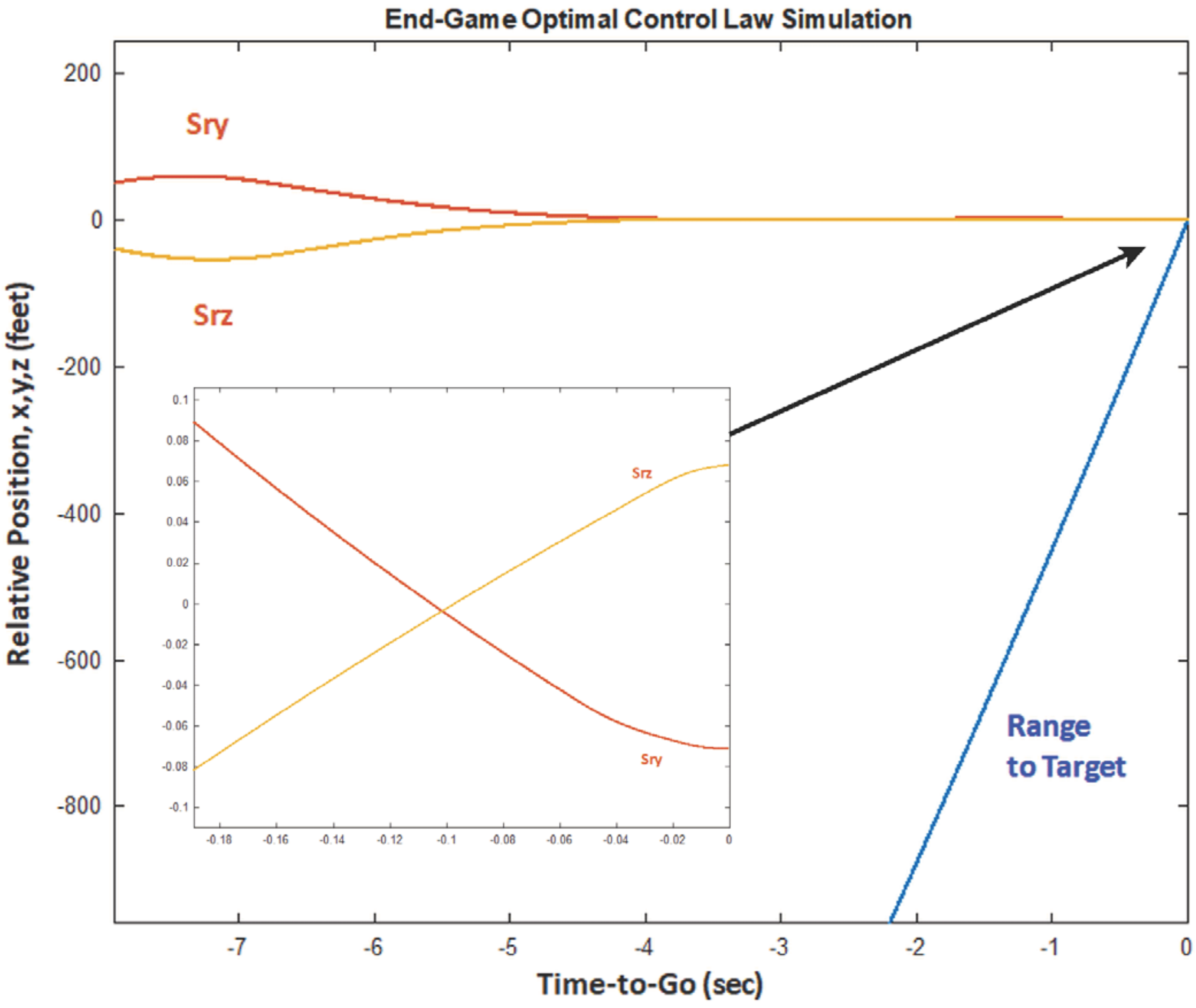
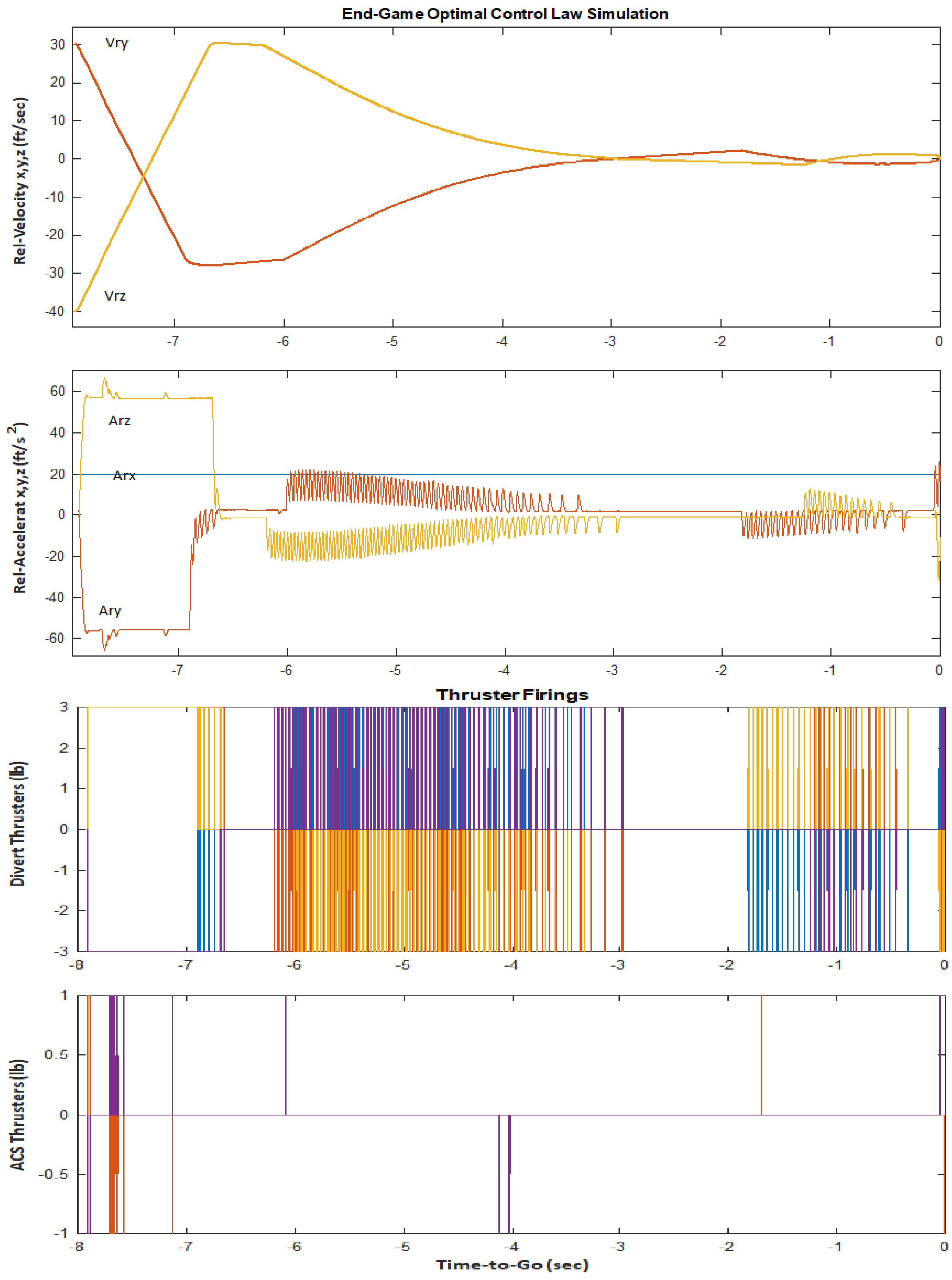


Figure 6.9b Optimal Control Guidance

6.3 Optimal Control Simulation Results

The simulation is initialized from file "*start.m*" which loads the Flixan created models, matrices, optimal control gains and initializes the relative spacecraft position, velocity, and target acceleration. What is different in this model in comparison with the previous simulation in Section 5.1, is that in addition to the relative position which is driven to zero at impact (t_{go}=0), the normal and lateral velocities (V_{ry} and V_{rz}) are also reduced in magnitude. It means that the interceptor direction is eventually shaped along the LOS, in a head-on collision with the target with zero sideslip, which makes it less vulnerable to range errors. Notice, there was a significant amount of side speed in the previous PN simulation. The divert thrust capability, however, was increased to achieve the improved performance.




End-Game Optimal Control Law Simulation
Rel-Velocity x,y,z (ft/sec)
Vry
Vrz
Rel-Accelerat x,y,z (ft/s^2)
Arz
Arx
Ary
Thruster Firings
Divert Thrusters (lb)
ACS Thrusters (lb)
Time-to-Go (sec)

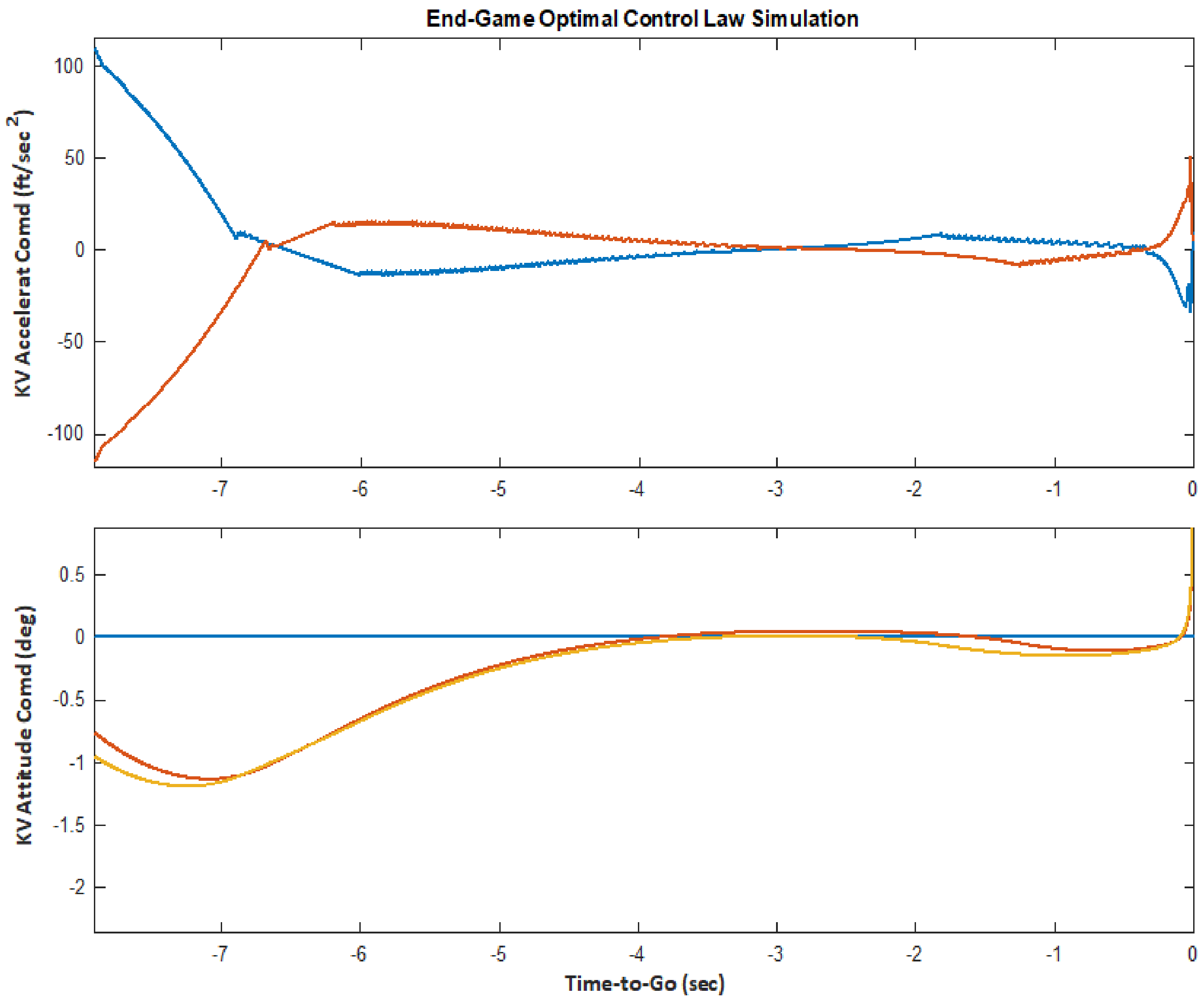


Figure 6.10 Optimal Control Law Simulation file: "Sim_6dof.mdl" Response to Initial Conditions

7. State Estimator Design

The LQR control law requires feedback from the state variables. However, most of the system states are not measurable and our next step is to design a state observer from the two outputs of the dynamic model, the relative position S_r and the interceptor acceleration A_I perpendicular to the LOS. In this section we will use the steady-state Kalman-Bucy filter, an observer that will reconstruct the state vector from the measurements so that we can apply our optimal state-feedback control law. The estimated state approximately converges to the actual state vector. The state observer also requires knowledge of the dynamic model in equation 4.2. We will also assume that the system is corrupted with two types of noise: state excitation noise, and measurement noise, as shown in equation 7.1. They are white noise, zero mean, and uncorrelated

$$\dot{\underline{x}}(t) = A\underline{x}(t) + Bu(t) + G\,w(t)$$

$$\underline{y}(t) = C\,\underline{x} + \underline{v}(t) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} S_r \\ V_r \\ A_T \\ A_I \end{pmatrix} + \underline{v}(t) \qquad (7.1)$$

Where:

$\underline{x}(t)$	is the state vector of dimension n
$\underline{u}(t)$	is the control input vector of dimension m
$\underline{y}(t)$	is the measurement vector of dimension r (r≤n)
$\underline{w}(t)$	is the process noise of dimension l, where (l≤n) and has a covariance matrix Q_{pn}, where: $Q_{pn}= Q_{pn}' \geq 0$
$\underline{v}(t)$	is the measurement noise with intensity $R_{mn}= R_{mn}' \geq 0$

The solution to the Kalman Filter is obtained by minimizing the quantity in equation 7.2, where the matrix W is (nxn) positive semi-definite. The state vector estimate $\hat{\underline{x}}$ from the KF output will converge to the actual system state $\underline{x}$. Figure 7.1 is a functional block diagram representation of the Kalman-Filter showing the output and its interconnection with the plant input $\underline{u}$ and output $\underline{y}$.

$$J = \lim E\left[\left(\underline{x}(t) - \hat{\underline{x}}(t)\right)' W \left(\underline{x}(t) - \hat{\underline{x}}(t)\right)\right] \qquad as\ t \to \infty \qquad (7.2)$$

The estimate is obtained by solving the following differential equation 7.3, where K_f is the Kalman-Filter gain. The solution exists when the pair (A',C') is stabilizable and the pair (A, $GQ_{pn}G^T$) is detectable. The state estimate is initialized with the expected initial state vector at t=0, $\hat{x}(0) = E\left[\underline{x}(0)\right]$.

$$\dot{\hat{\underline{x}}}(t) = A\hat{\underline{x}}(t) + Bu(t) + K_f\left[y(t) - C\,\hat{\underline{x}}(t)\right]$$
$$K_f = P\,C^T R_{mn}^{-1} \qquad (7.3)$$

The matrix P is symmetric positive semi-definite and it is obtained from the steady-state solution of the asymptotic Riccati equation

$$A P + P A^T + G Q_{pn} G^T - P C^T R_{mn}^{-1} C P = 0 \qquad (7.4)$$

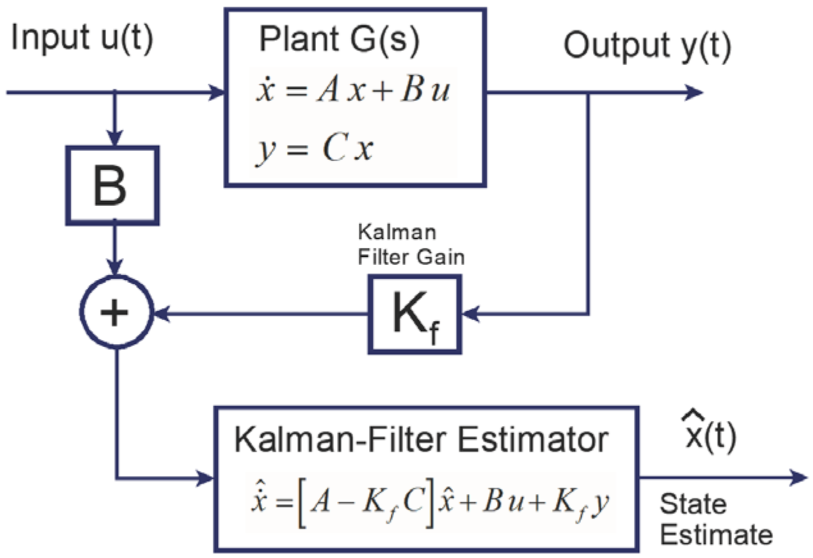
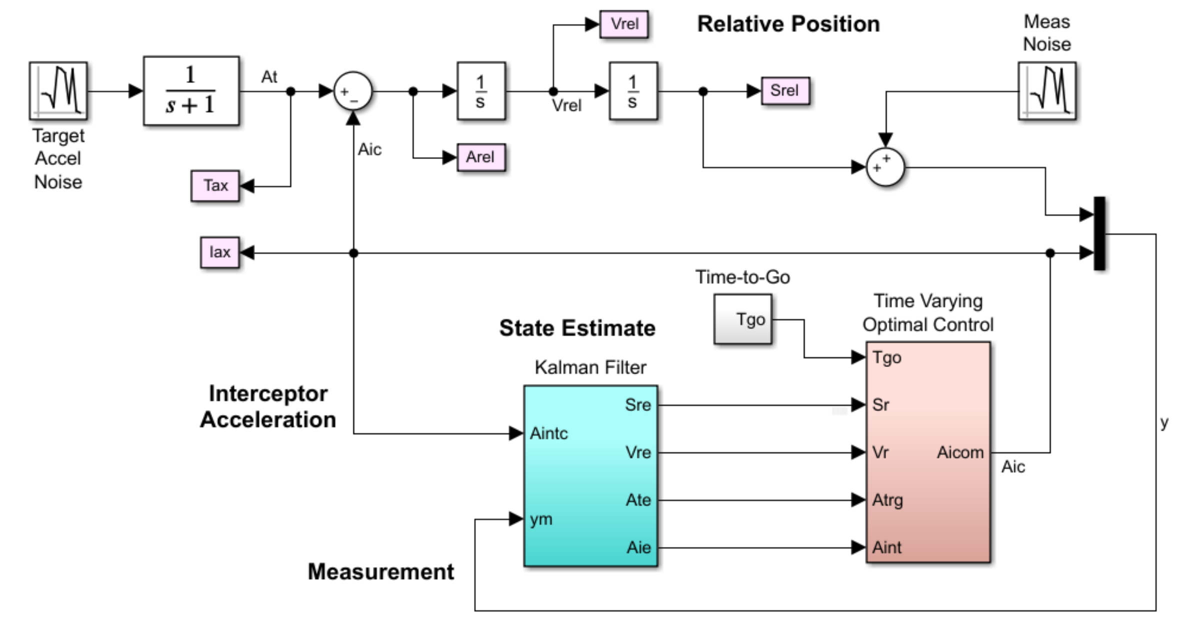
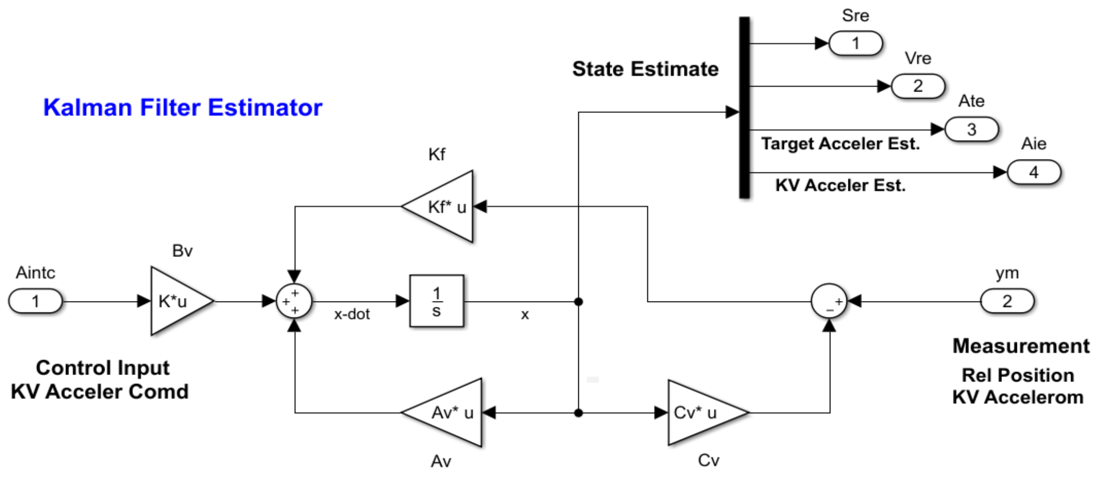


Figure 7.1 Kalman-Filter State-Vector Estimator

7.1 Simple End-Game Simulation Model with KF Estimator

The end-game simulation in Figure 7.1 is similar to the simple model used in Section 6.2. It uses the end-game model of equations 4.2 and it includes a Kalman-Filter to estimate the four states from the 2 measurements which are: relative position and KV accelerometer perpendicular to the line-of-sight. It also uses the time-varying control law with the exception that the 4 state-vector feedback is now replaced with the output of the Kalman-Filter. The states are initialized to the same values as the actual states since they are replacing the original states. The simulation files are in folder "*Interceptor Spacecraft\5 Simple KF Estimator*", and the Simulink model is in file "*KF_Sim.mdl*". The initialization file "*init.m*" loads the dynamic model "*end_game.m*", the gains from "*Gain.mat*" which are functions of t_{go}, and the Kalman gain K_f that was obtained by solving equation 7.3 in Flixan. It also initializes the dynamic model and the Kalman-Filter states. The relative position, velocity, and KV acceleration responses in Figure 7.2 are identical to those obtained by using direct state-feedback. The effects of noise are more visible near the end of the trajectory where the gains are high.





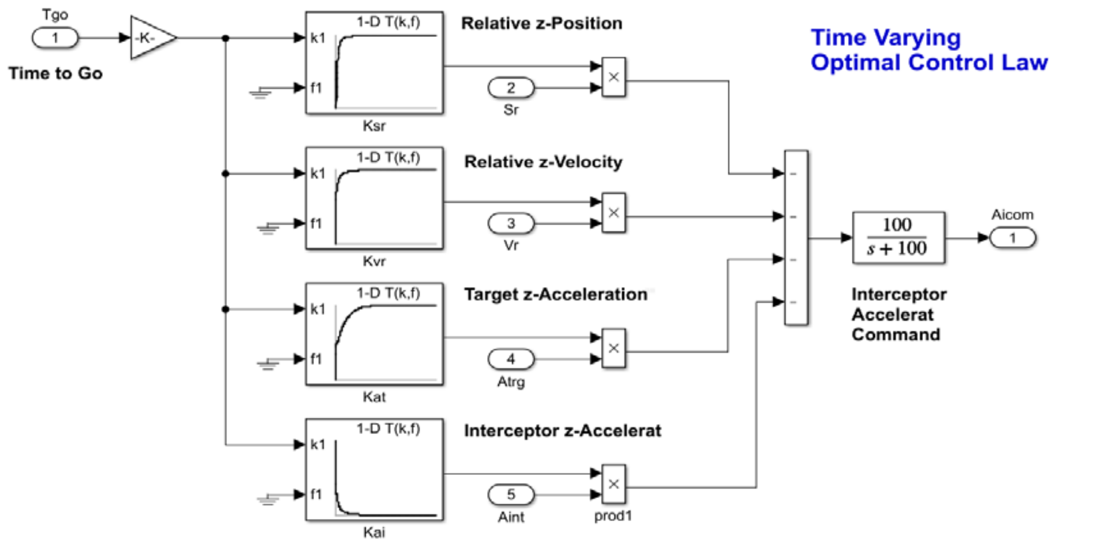


Figure 7.1 Simple End-Game Simulation “KF_Sim.mdl” that uses Feedback from the Kalman-Filter Output

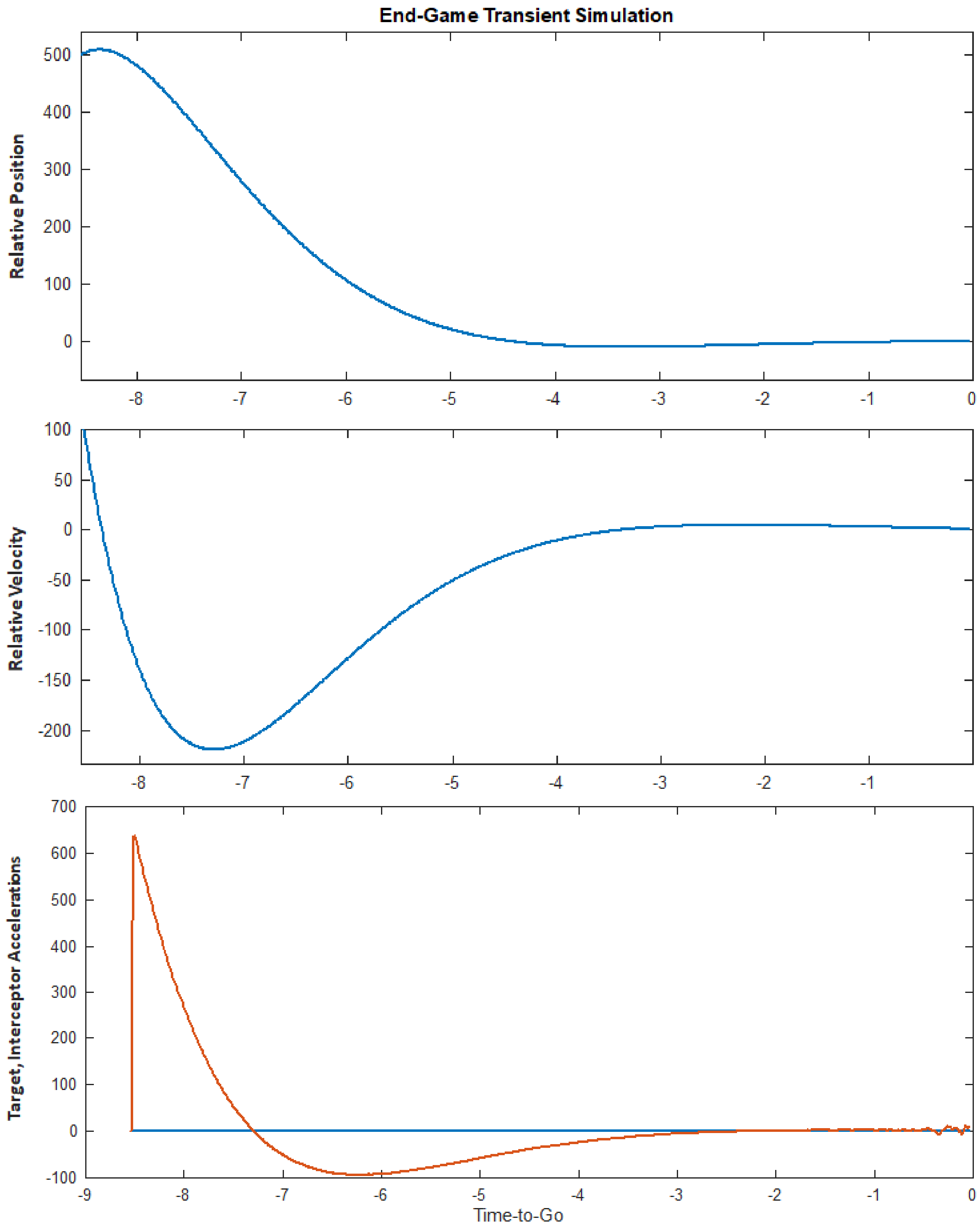
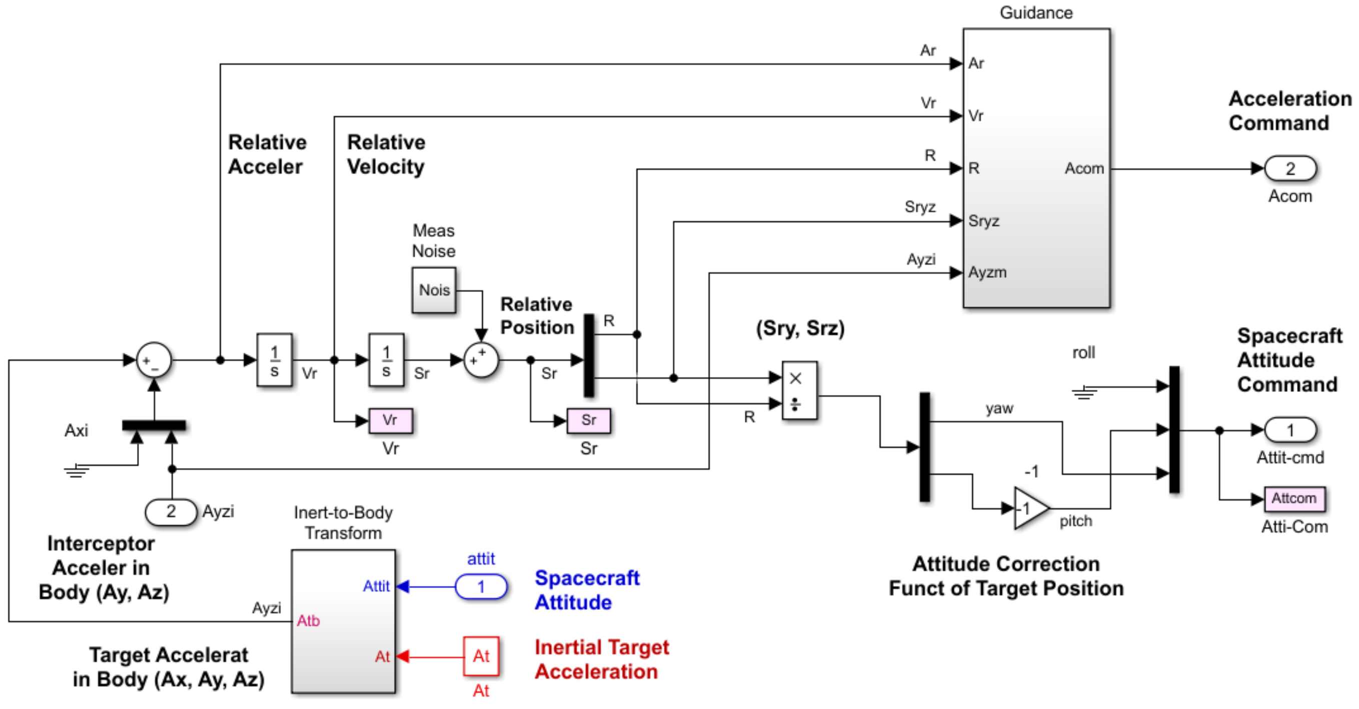


Figure 7.2 Response of the End-Game Simulation Model “KF_Sim.mdl” to Initial Condition and Noise

8. Dual Mode Simulation Model that Uses Combination of Optimal Control and PN

We will now apply our control law and estimator design using the detailed spacecraft models that includes flexibility. We will combine the two control laws in one model to achieve better impact results, beginning the trajectory with the optimal control law to reduce the normal components of velocity to almost zero prior to impact. Assuming that our range estimates are not very accurate and hence the t_{go} calculation, we will try to bring the normal velocity to almost zero, one second before impact and at this point we switch to fixed gains Proportional Navigation, and remain in this mode until impact. We also introduce noise in the position measurements. The relative measurements of the target position (S_{ry}, S_{rz}) and velocity (V_{ry}, V_{rz}) are calculated from the spacecraft azimuth and elevation attitude as it is tracking the target.



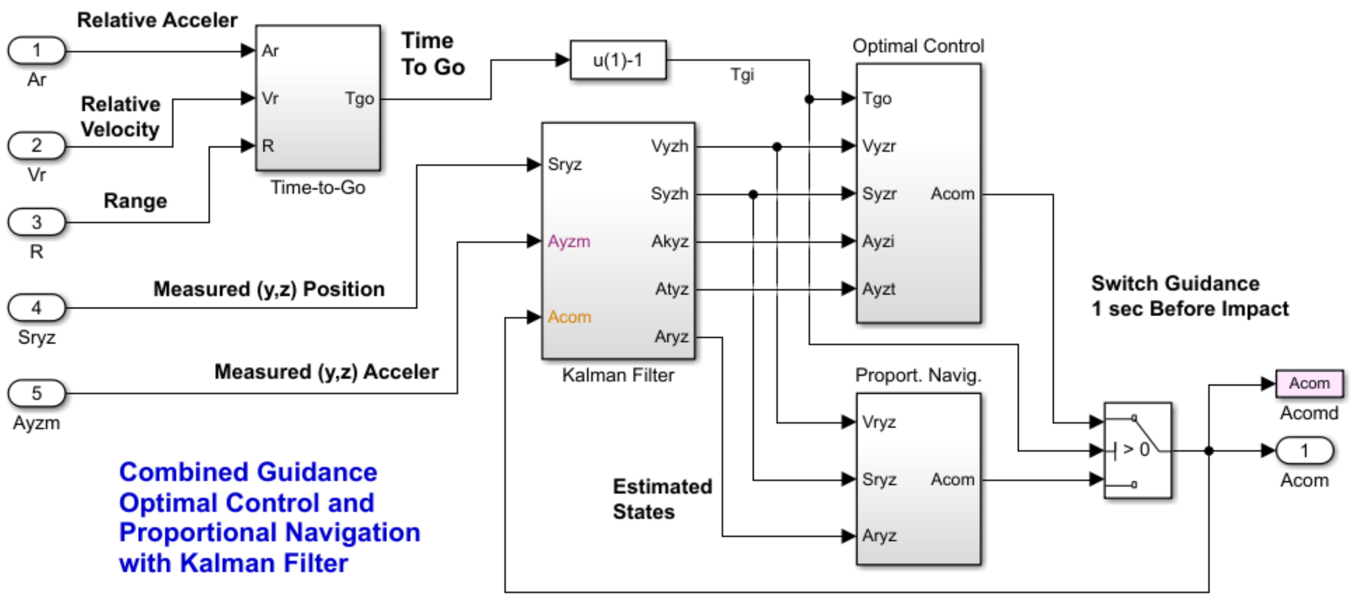


Figure 8.1 Combined Guidance Using Optimal Control with Proportional Navigation; the Relative Motion State Vector is estimated by a Kalman-Filter

The optimal control logic in Figure 8.1 uses the time-varying gains from the file calculated in Section 6 using the transient LQR program. The T_{go} input is shifted one second earlier than expected, in order to accommodate the uncertainty in range and the exact impact time. It also reduces the relative velocities closer to zero which is a good property for initializing PN. The switch activates the PN control law 1 sec before the expected impact time which guides it all the way to impact. Since the side velocities are small the sensitivity to range error is reduced and a successful impact may occur even if the target is a little closer or a little further away than expected.

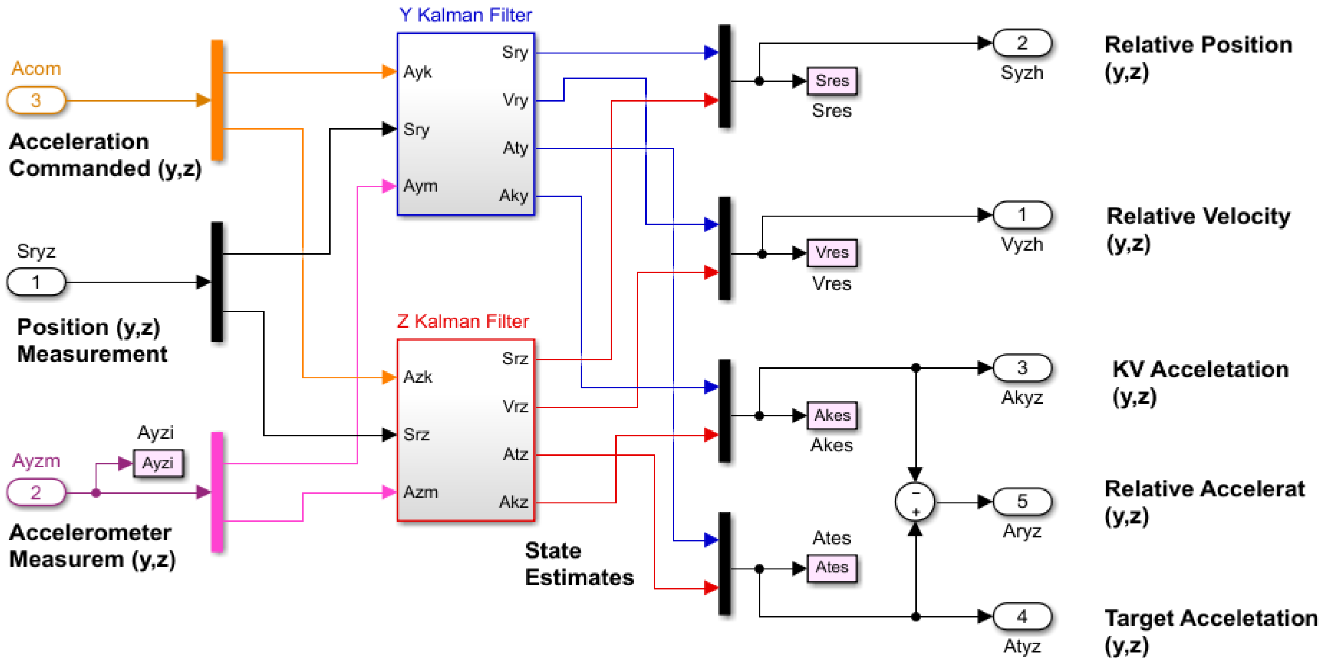


Figure 8.2 Normal and Lateral Motion Kalman-Filters Estimate the Relative Vehicle Position, Velocity, and Acceleration

Figure 8.2 shows the Kalman-Filter which estimates the relative spacecraft position and velocity, and also the accelerations in the y and z directions. The KF inputs are relative position calculated from the interceptor's attitude, and accelerometer measurements. It also receives the acceleration command calculated from the control law. The simulation model is in file "*Sim_6dof.mdl*" located in folder "*Interceptor Spacecraft\6 Combined End Game Sim*". It uses the Flixan generated flex spacecraft system "*rigbod_mod.m*". There is a similar model in file "*Sim_6dof2.mdl*" that uses the Flixan generated flight vehicle system "*kkv_acs.m*". The results from both simulations are identical.

8.1 Dual Mode Simulation Results

The target-to-KV relative position errors measured by the seeker and the accelerometer measurements are used to estimate the state-vector and generate acceleration commands to take out the position errors. The relative velocity errors are also significantly reduced by the optimal control law prior to switching to PN, that is 1 sec before impact. The ACS is tracking the target position by rotating the KV. Noise is introduced in the position measurement which makes the response more noisy, due to structural flexibility.

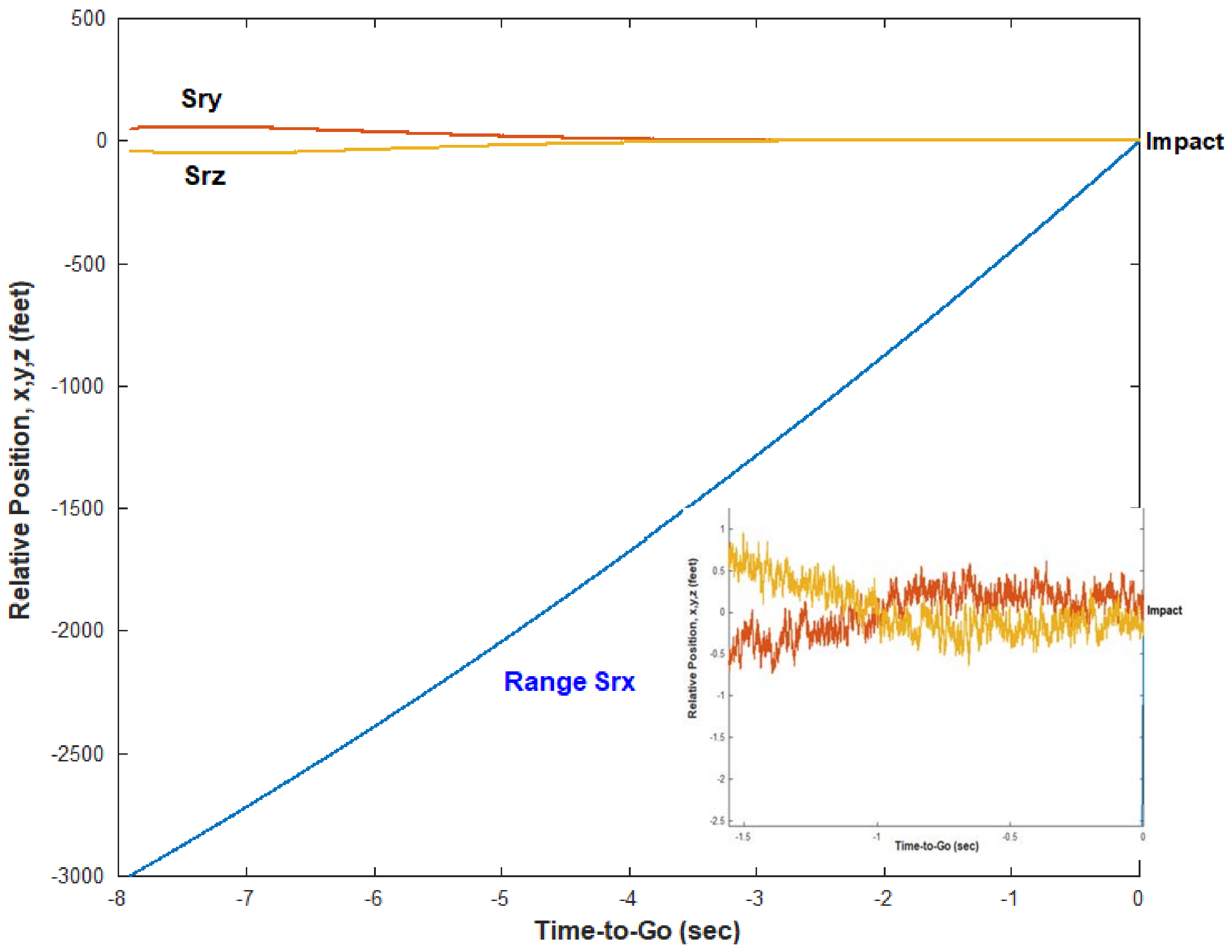
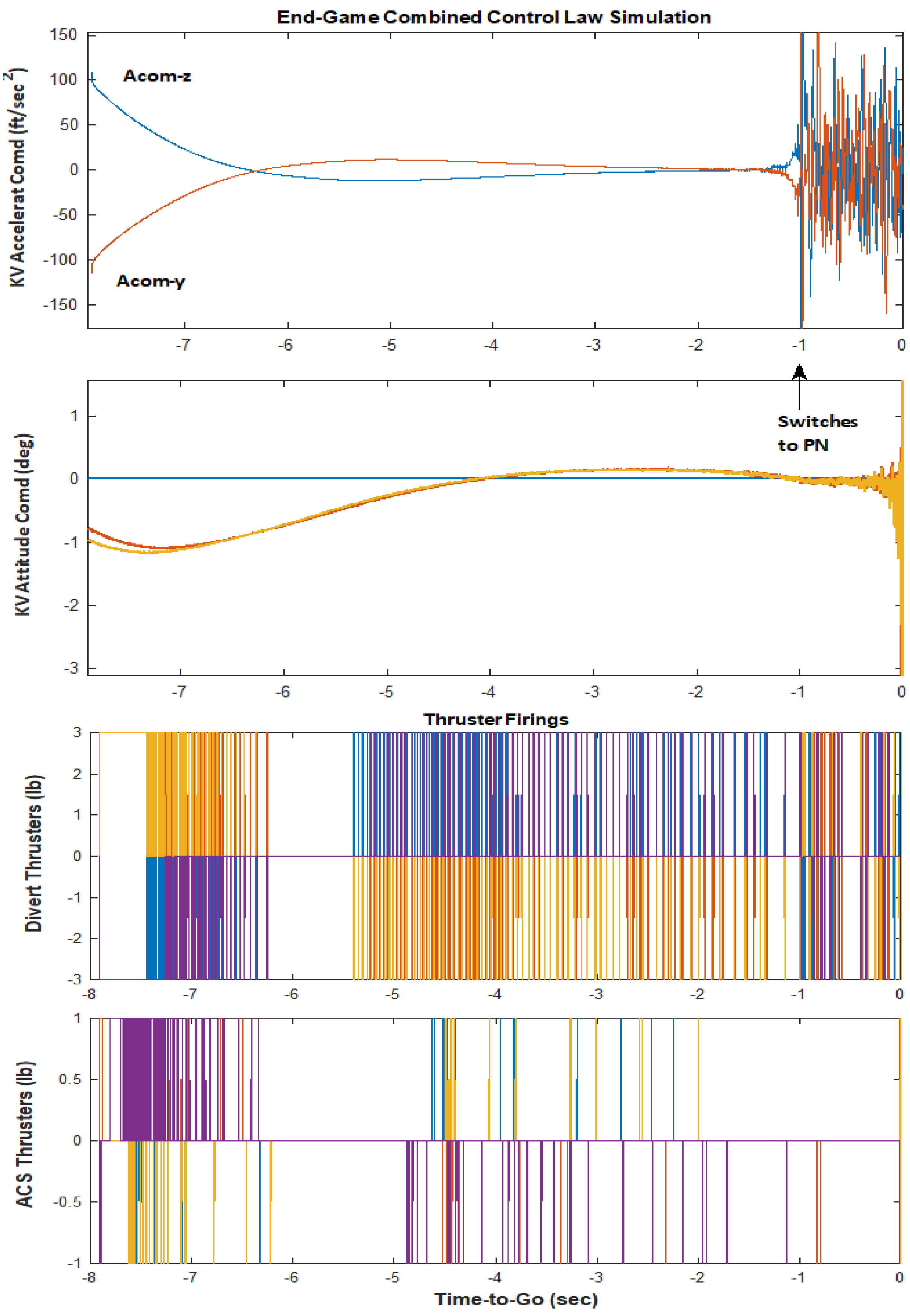
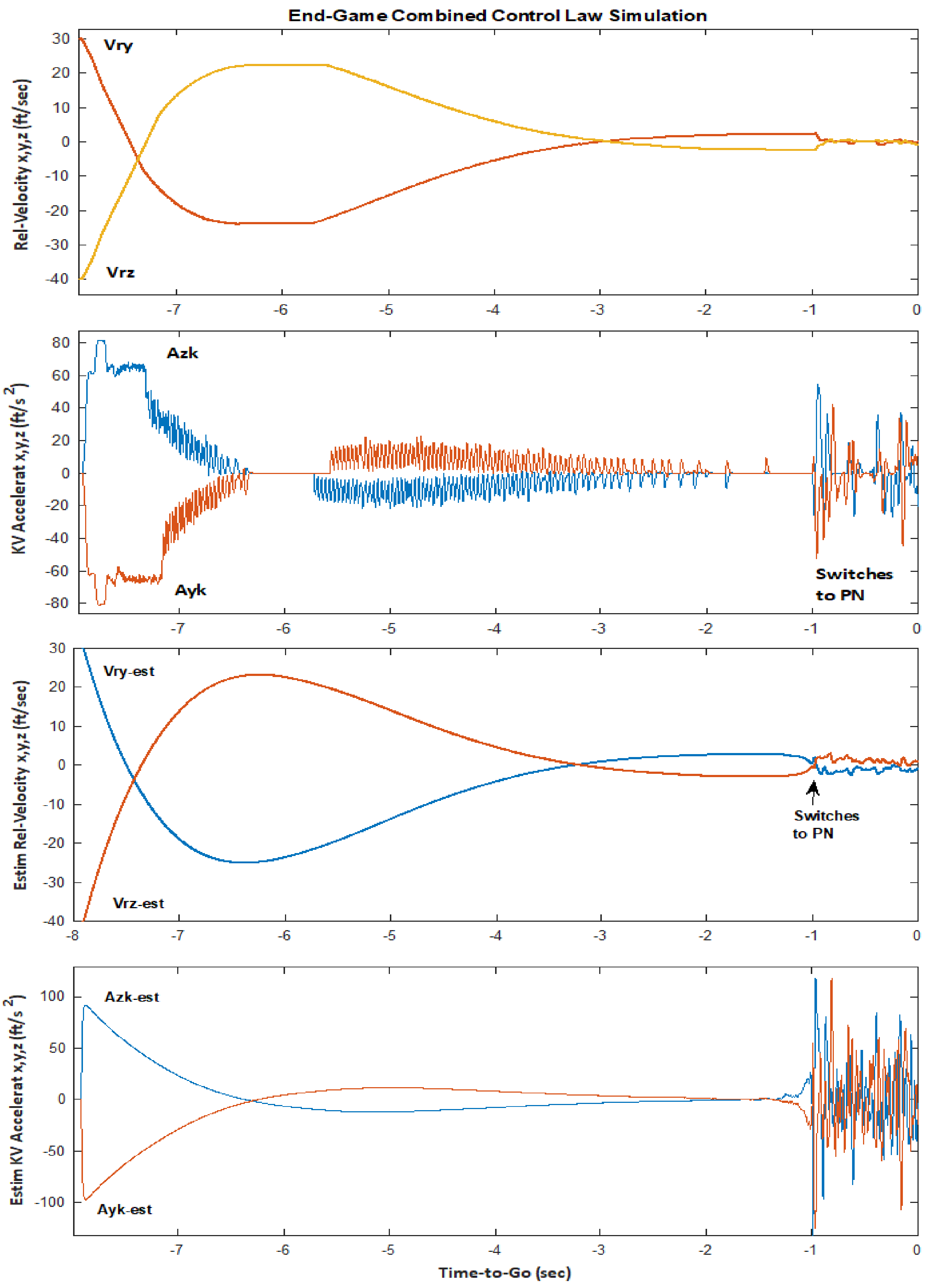


Figure 8.3 The Relative Position is less than 0.8 feet at Impact Using Dual Mode OC/ PN Control



End-Game Combined Control Law Simulation
KV Accelerat Comd (ft/sec^2)
Acom-z
Acom-y
150
100
50
0
-50
-100
-150
-7
-6
-5
-4
-3
-2
-1
0
Switches to PN
KV Attitude Comd (deg)
1
0
-1
-2
-3
Thruster Firings
Divert Thrusters (lb)
3
2
1
0
-1
-2
-3
-8
ACS Thrusters (lb)
1
0.5
0
-0.5
-1
Time-to-Go (sec)


End-Game Combined Control Law Simulation
Rel-Velocity x,y,z (ft/sec)
Vry
Vrz
KV Accelerat x,y,z (ft/s 2)
Azk
Ayk
Switches to PN
Estim Rel-Velocity x,y,z (ft/sec)
Vry-est
Vrz-est
Switches to PN
Estim KV Accelerat x,y,z (ft/s 2)
Azk-est
Ayk-est
Time-to-Go (sec)