Technische Universität München

Lehrstuhl für Flugsystemdynamik

University of Pisa

Modeling and Implementation of the Atmosphere in MATLAB/Simulink for flight simulation

Author:

Sara Perfetto

Supervisors:

Prof. Eugenio Denti (Univ. of Pisa)
Prof. Giovanni Mengali (Univ. of Pisa)
Prof. Florian Holzapfel (TUM)
Dipl.-Ing. Stefan Hager (TUM)

A.A. 2012 - 2013

Sommario

La presente tesi descrive i risultati del lavoro svolto dall'autrice presso l'Istituto di Dinamica dei Sistemi di Volo (FSD) dell'Università Tecnica di Monaco di Baviera (TUM).

Scopo del lavoro è lo sviluppo e l'implementazione di un Modello di Atmosfera in MATLAB/Simulink da utilizzare in un dispositivo di simulazione del volo.

Il Modello di Atmosfera realizzato è costituito da quattro sottosistemi: un Modello di Atmosfera Standard (ISA Model), un Modello di Vento (Wind Model), un Modello di Turbolenza (Turbulence Model) ed un Modello di Raffica (Gust Model).

Particolare attenzione è stata prestata all'implementazione di un modello aggiuntivo che consente di prendere in considerazione la correlazione che esiste tra le velocità dell'aria percepite in punti diversi dello spazio: il Modello di Correlazione (Correlation Model). Tale modello garantisce un più corretto funzionamento del Modello di Turbolenza nel caso in cui sia necessario considerare più aerei che volano in un'area limitata.

Dopo una breve introduzione iniziale, vengono presentate le caratteristiche generali del Modello di Atmosfera e del Modello di Correlazione.

In ogni capitolo sono descritti in dettaglio i sottosistemi del Modello di Atmosfera: requisiti, ipotesi alla base della modellazione, ambiti di validità, limitazioni, algoritmi utilizzati per l'implementazione, specifiche dal punto di vista dell'architettura, layout strutturale, piano di verifica e risultati.

In seguito è illustrato in dettaglio il Modello di Correlazione.

Infine, l'ultimo capitolo è dedicato all'assemblaggio finale delle varie parti.

Abstract

The present thesis describes the results of the work made by the author at the Flight System Dynamic Institute (FSD) at the Technische Universität München (TUM).

The aim of the work is the development and the implementation of an Atmosphere Model in MATLAB/Simulink to be used in a flight simulation device.

The Atmosphere Model consists of four main subsystems: the International Standard Atmosphere (ISA) Model, the Wind Model, the Turbulence Model and the Gust Model.

A particular attention has been spent for the implementation of an additional model which allows to take into account the correlation that exists between the air velocities perceived in different points of the space: the Correlation Model. This model guarantees a more correct operation of the Turbulence Model in the case in which it is necessary to consider more aircraft flying in a restricted area.

After a brief initial introduction, the general characteristics of the Atmosphere Model and of the Correlation Model are presented.

In each chapter the subsystems of the Atmosphere Model are described in detail: requirements, modeling assumptions, scope of validity, limitations, algorithms for implementation, architecture specification, structural layout, verification plan and results.

Afterwards, the Correlation Model is explained in detail.

Finally, the last chapter is dedicated to the final assembly of the different parts.

Content

List of Figures

List of Tables

Introduction

The present thesis describes the results of the work made by the author at the Flight System Dynamic (FSD) Institute at the Technische Universität München (TUM), under the supervision of Prof. Florian Holzapfel (TUM), of Engineer Stefan Hager (TUM), of Prof. Eugenio Denti (University of Pisa) and of Prof. Giovanni Mengali (University of Pisa). The main purpose of the work is the development of a simulation model of the atmosphere in MATLAB® and Simulink®, as part of a flight simulation device of a turboprop trainer aircraft.

The Atmosphere Model is a part of a wider project that requires for the construction of the models the exclusive use of Libraries and Toolboxes property of the FSD Institute. For this reason and in order to meet specific requirements of the client, the work has involved the rewriting of some models often already available in Simulink®.

The first step has focused on the research and on the study of the mathematical models necessary for the development of the systems, afterwards the existing models in Simulink® were analyzed. Thereafter, the actual implementation in MATLAB® and Simulink® has been made.

The Atmosphere Model consists of four main subsystems: the International Standard Atmosphere (ISA) Model, the Wind Model, the Turbulence Model and the Gust Model.

Each subsystem allows to obtain specific output quantities after providing the input signals: all the parameters necessary for the operation of the model and the variables in output are then listed and explained in detail.

In order to take into account the possibility in which more aircraft fly in a restricted area an additional model was developed: the Correlation Model.

This model allows to take into account the correlation between the air velocities perceived in twenty different points of the space, as required by the client, and guarantees the correct operation of the Turbulence Model.

For this reason, during the implementation of the Turbulence Model, an important part of the work has focused on the study of the mathematical models that describe the correlation between the air velocities perceived in different points of the space. This effort was made in order to establish a method for the computation of correlated random signals, which are input signals for the Turbulence Model.

Since the Correlation Model requires signals that contain the data of twenty aircraft, the integration with the Turbulence Model (working for individual aircraft) was made possible through systems that build such signals.

All the simulation models have been produced to be integrated with other models made within the FSD Institute for the same flight simulation device. Therefore, in order to allow an easy integration of the different parts, the models have been designed to comply with the standards adopted by the Institute: nomenclature, symbols, layout and so on.

The models were made in order to meet specific requirements of the client for the project, making however always refer to MIL-F-8785C.

The main information about the nomenclature and symbols used are contained in Appendix B.

In addition, an important part of the work has involved the verification and validation of the models; each model was tested in order to verify the accuracy of the results obtained and compared with other models that carry out similar computations using alternative methods.

Finally, during the last period spent at the Institute, the work has mainly focused on the drafting of the reports attached to each system developed, produced according to the standard adopted.

1 Systems Description

1.1 Introduction

The aim of the work is the modeling and implementation of the atmosphere in MATLAB® and Simulink® for a flight simulation device.

The *Atmosphere Model* allows to determine some important and useful quantities concerning the atmosphere around the aircraft.

It shall be incorporated into the simulation model of an atmospheric flight simulation device relative to a turboprop trainer aircraft.

In particular, the *Atmosphere Model* consists of four main subsystems:

- ISA Model
- Wind Model
- Turbulence Model
- Gust Model

In order to take into account the case in which twenty aircraft flying in a restricted area, a fifth part was developed:

- Correlation Model

This model allows to take into account the correlation that exists between the air velocities perceived by different aircraft.

1.2 General Structure

The *Atmosphere Model* is a top level system consists of four children systems.

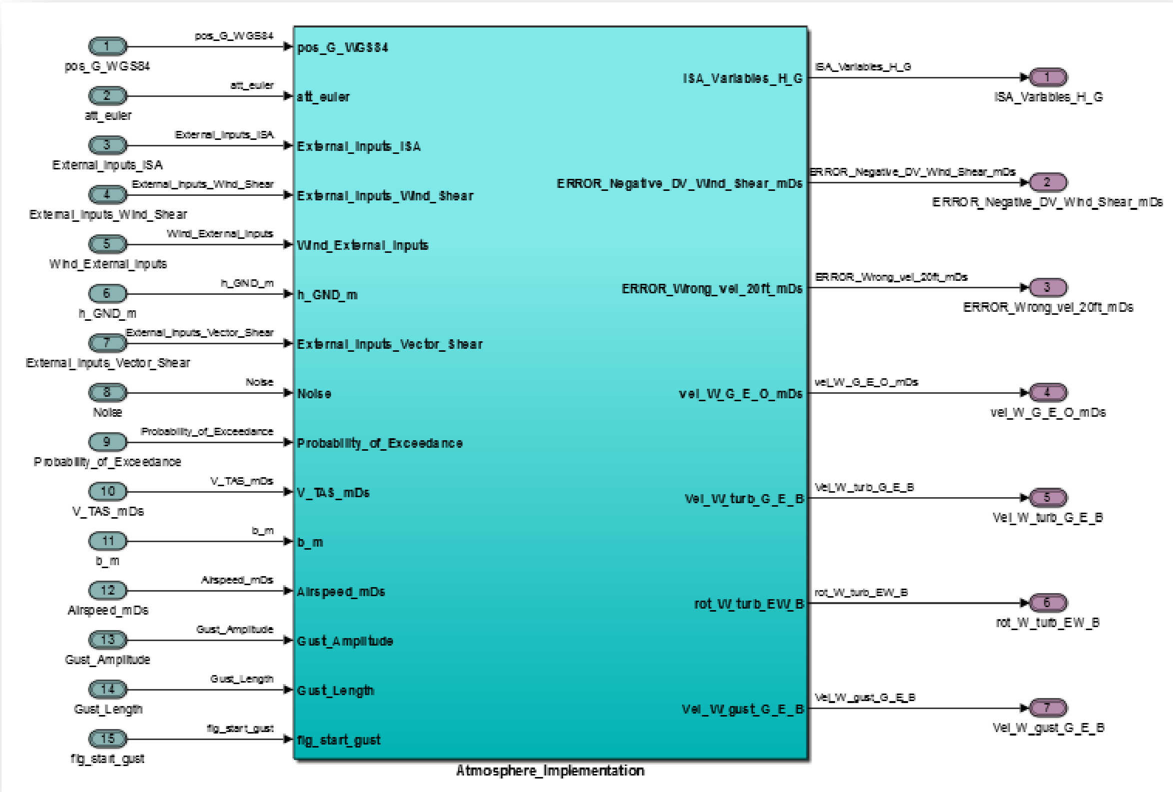


Figure 1-1 Atmosphere Model

The *Correlation Model* was implemented separately and then properly integrated with other systems.

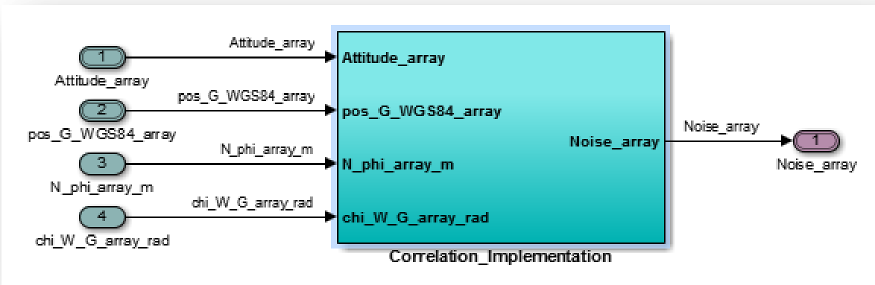


Figure 1-2 Correlation Model

In the following figure can be displayed the structure of the *Atmosphere Model*:

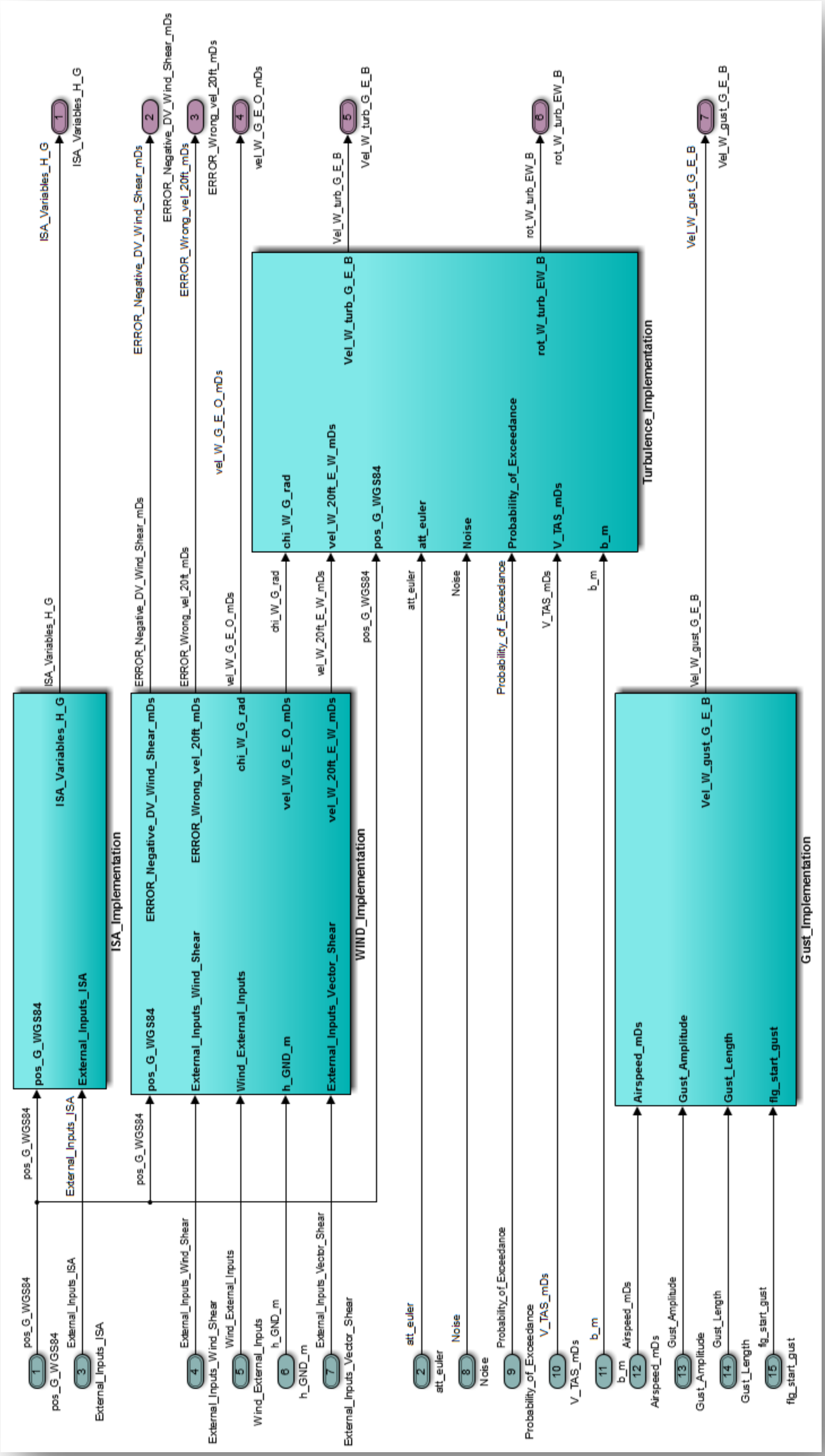


Figure 1-3 Structure of Atmosphere Model: four children systems

In the following chapters there is a detailed description of each subsystem, accompanied by all the opportune verifications for correct operation.

In addition, there is a detailed description of the *Correlation Model* and the verification carried out.

1.3 Solutions: two different proposals

The need to assess the correlation between the air velocities perceived by each of the twenty aircraft suggested to consider a solution that would allow to evaluate the quantities of interest for all aircraft simultaneously.

For this reason, were made two different models of the *Atmosphere Model*:

<u>Model 1</u>

This model allows to perform the computations for all twenty aircraft: input and output signals are all vectors of size [1,20] (referred to as *array*).

Since the *Correlation Model* requires data of all twenty aircraft, this proposed solution is directly compatible and integrable with it.

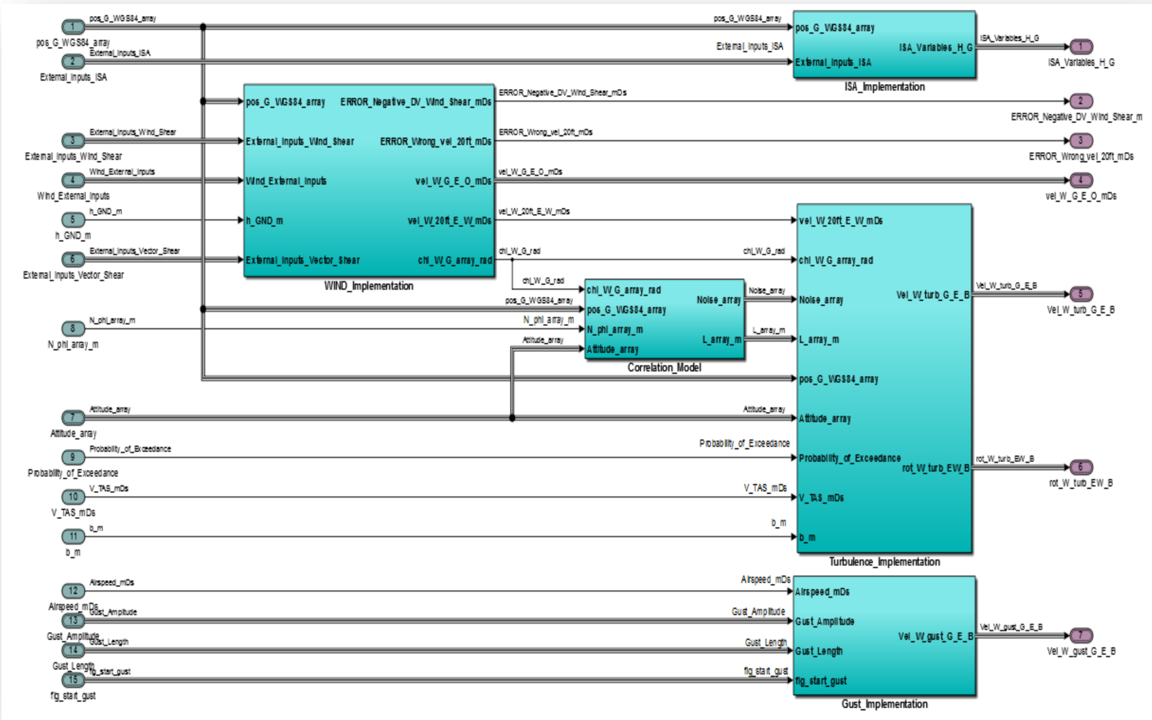


Figure 1-4 Model 1: Solution size [1,20]

Some of the input signals necessary for the operation of the *Correlation Model* are the same as those to the other subsystems and the output signals of this *Model* are directly usable by the others.

<u>Model 2</u>:

This *Atmosphere Model* works for individual airplanes. To know the variables of twenty aircraft is therefore necessary to use twenty similar systems, each with inputs and outputs related to a single aircraft.

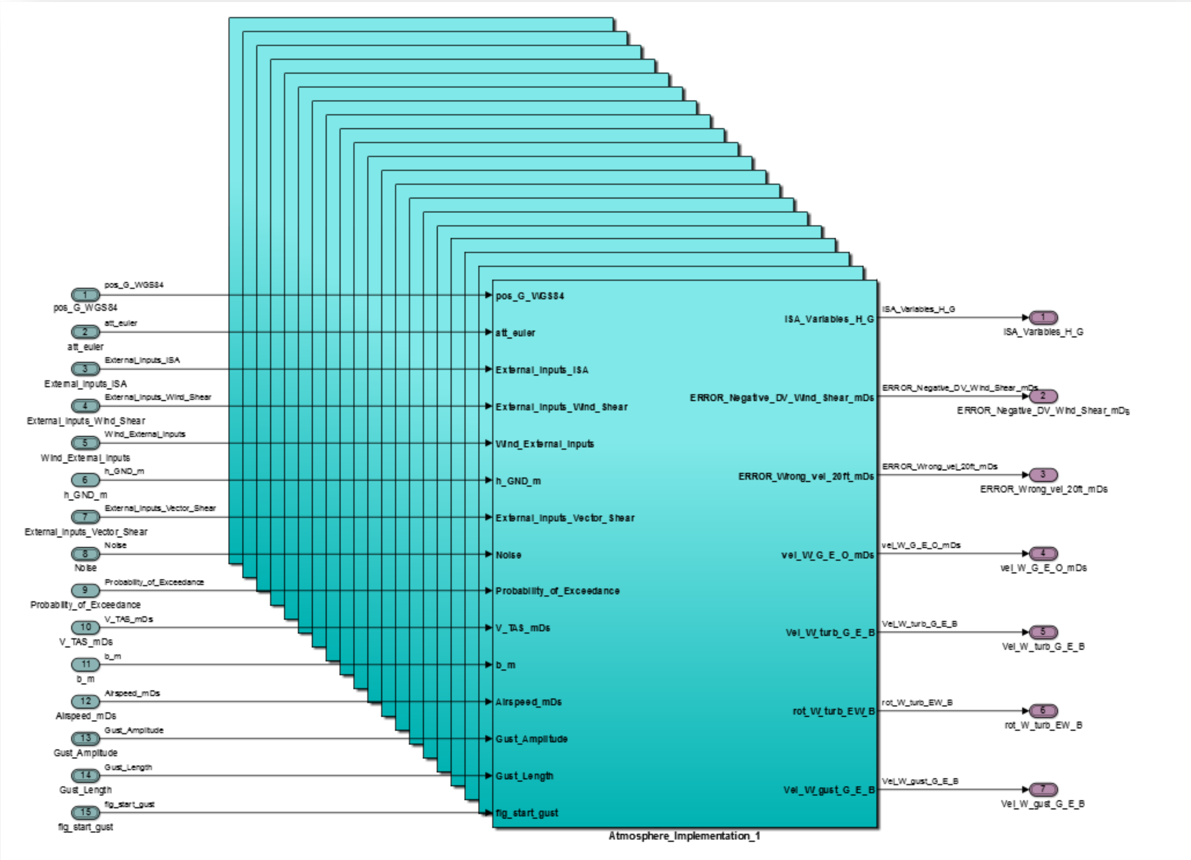


Figure 1-5 Model 2: Solution size [1,1] - twenty similar systems

To make the figure more readable, here only inputs and outputs of the first aircraft are represented.

In this case, it was necessary to devise a method of integration of the *Correlation Model* , which works with signals size twenty, with twenty *Atmosphere Models,* each of which works with signals of unitary dimension.

The compatibility was guaranteed through the use of two Embedded Matlab Functions, represented in the following figure.

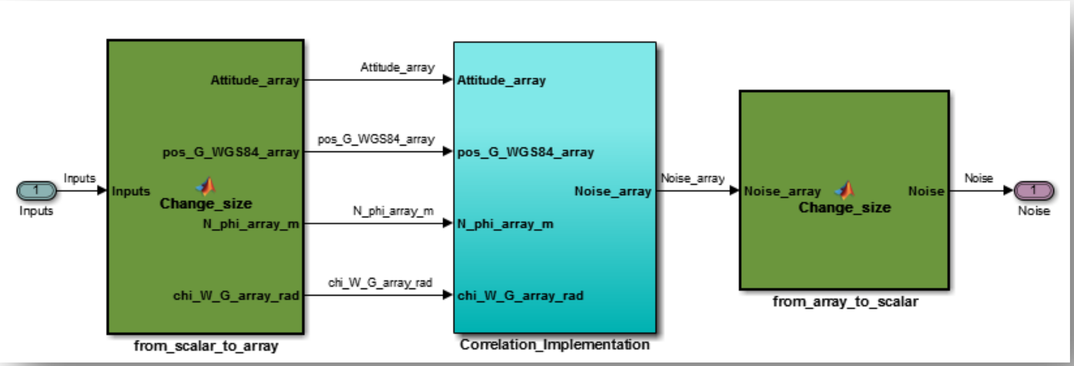


Figure 1-6 Correlation Model: Embedded Matlab Functions

More details will be provided in the following chapters and in Appendix A.

1.4 The choice of the optimal solution

The choice of the optimal solution was done evaluating the number of signals in input and output needed in the two different models.

In particular, considering that the model is part of a wider project of a flight simulator in which the systems operate with signals of unitary dimension (then for a single aircraft), the use of Model 1 would require the creation of many signals of dimension twenty.

Is therefore much more convenient to use twenty unitary models and create signals of size twenty only for the use of the *Correlation Model*, made compatible with other systems.

The choice of Model 2 is then appeared as the most suitable.

2 <u>ISA Model</u>

2.1 Introduction

The most recent definition of the International Standard Atmosphere (ISA) is the *U.S. Standard Atmosphere, 1976* [1] developed jointly by NOAA, NASA and the USAF. It is a revision of the *U.S. Standard Atmosphere, 1962* and was generated under the impetus of increased knowledge of the upper atmosphere obtained over the past solar cycle.

The *U.S. Standard Atmosphere, 1976* is an idealized steady-state representation of the earth's atmosphere from the surface to 1000 km and it is assumed to exist in a period of moderate solar activity.

This Standard is identical with the earlier *U.S. Standard Atmosphere, 1962* up to 51 geopotential kilometers (km') and the tables are based on traditional definitions.

For heights from 51 km' to 86.852 km' (from 51.413 to 86 geometrical kilometers) the tables are based upon the average on atmospheric data dating back to 1976.

Up to 86.852 km' the model assumes that there is hydrostatic equilibrium in which the air is treated as a homogeneous mixture of the several constituent gases.

At greater heights, the definitions governing the Standard are more sophisticate and the hydrostatic equation, applied to a mixed atmosphere, gives way to the general equations which takes into account the change of composition with height.

In the *U.S. Standard Atmosphere, 1976* is used the International System of metric units (SI).

The International Organization for Standardization (ISO) publishes the ISA as an international standard, ISO 2533:1975 [2].

2.2 Description of the Functional and Operational Intent

The system is intended to compute the main proprieties of the standard atmosphere: the Temperature (T) , the Pressure (P), the Density (ρ) , the Speed of Sound (a), the Dynamic Viscosity (μ) and the Kinematic Viscosity (ν).

These proprieties can be obtained for the troposphere (0 m' - 11000 m') and the lower stratosphere (11000 m' - 20000 m').

Altitudes in this model are referred to geopotential altitudes (H^G) rather than geometrical altitudes (h^G) but the units are indicated without superscript. The final model will be used as part of a simulation model, which shall be incorporated into the simulation framework of a flight simulation device. Therefore it is necessary that all sub-systems are compliant to code generation requirements.

2.3 Requirements

Requirements that the model must satisfy, are essentially of three types: functional, operational and implementation requirements. These requirements are summarized in the following tables and are named by the acronyms that allow a more rapid identification.

2.3.1 Functional Requirements

Requirement Name **Computation of Temperature**	Requirement ID R-FUN-ENV_ISA_01
Derived from Purpose of the system.	
Requirement Definition The Temperature of the atmosphere in the troposphere and lower stratosphere, referred to geopotential altitudes, shall be computed.	

Table 2-1 R-FUN-ENV_ISA_01

Requirement Name **Computation of Pressure**	Requirement ID R-FUN-ENV_ISA_02
Derived from Purpose of the system.	
Requirement Definition The Pressure of the atmosphere shall be computed in the troposphere and lower stratosphere, referred to geopotential altitudes.	

Table 2-2 R-FUN-ENV_ISA_02

Requirement Name	Requirement ID
Computation of Density	R-FUN-ENV_ISA_03
Derived from Purpose of the system.	
Requirement Definition The system shall compute the Density of the atmosphere in the troposphere and lower stratosphere, referred to geopotential altitudes.	

Table 2-3 R-FUN-ENV_ISA_03

Requirement Name	Requirement ID
Computation of Speed of Sound	R-FUN-ENV_ISA_04
Derived from Purpose of the system.	
Requirement Definition The Speed of Sound of the atmosphere in the troposphere and lower stratosphere, referred to geopotential altitudes, shall be computed.	

Table 2-4 R-FUN-ENV_ISA_04

Requirement Name	Requirement ID
Computation of Dynamic Viscosity	R-FUN-ENV_ISA_05
Derived from Purpose of the system.	
Requirement Definition The Dynamic Viscosity shall be computed in the troposphere and lower stratosphere, referred to geopotential altitudes.	

Table 2-5 R-FUN-ENV_ISA_05

Requirement Name	Requirement ID
Computation of Kinematic Viscosity	R-FUN-ENV_ISA_06
Derived from Purpose of the system.	
Requirement Definition The system shall compute the Kinematic Viscosity in the troposphere and lower stratosphere, referred to geopotential altitudes.	

Table 2-6 R-FUN-ENV_ISA_06

2.3.2 Operational Requirements

Requirement Name	Requirement ID
Incorporation into Flight Simulator Simulation Model	R-OPS-ENV_ISA
Derived from	
Usage intents	
Requirement Definition	
The model shall be incorporated as a child system into the simulation model of an (atmospheric) flight simulation device. Therefore it is necessary that all components support code generation.	

Table 2-7 R-OPS-ENV_ISA

2.3.3 Implementation Requirements

Requirement Name	Requirement ID
Numeric Efficiency	R-NUM-ENV_ISA
Derived from	
Global Implementation Guidelines	
Requirement Definition	
The coded algorithm must not contain any of the numerical inefficient programming techniques listed below unless detailed justification substantiates indispensability.	
Programming Techniques to be Avoided:	
Unused / Dead Code Branches	
Computational Redundancies	
Matrix Inversions	
Scalar Expansion of Vector / Matrix Math	
Circle Computations	
Inefficient Lookup Table Programming	
Algebraic Loops	

Table 2-8 R-NUM-ENV_ISA

Requirement Name	Requirement ID
Input / Output Interface Compliance to Parent System and Child Systems	R-IOC-ENV_ISA
Derived from Global Implementation Guidelines	
Requirement Definition Input and Output interface must comply with parent system.	
Compliance required to:	
Global bus object definitions	
I/O signal name matching to parent system	
I/O signal unit matching to parent system	
I/O signal data type matching to parent system	
I/O signal data range compatibility matching to parent system	

Table 2-9 R-IOC-ENV_ISA

Requirement Name	Requirement ID
Implementation Compliance to FSD Style Guidelines	R-SGC-ENV_ISA
Derived from Global Implementation Guidelines	
Requirement Definition Only a subset of SIMULINK blocks is allowed to be implemented.	
Allowed Libraries and Toolboxes:	
FSD Compliant Base	
Use of other Libraries and Toolboxes is Forbidden!	

Table 2-10 R-SGC-ENV_ISA

Requirement Name	Requirement ID
Implementation Standards Compliance	R-ISC-ENV_ISA
Derived from Global Implementation Guidelines	
Requirement Definition The coded algorithm must not contain any programming technique listed below unless detailed justification substantiates indispensability.	
Forbidden Programming Techniques:	
Discrete Switches *	
Memory Blocks	
Time Delays	
Time Dependent / Non-Autonomous Elements	
In-lined Integrations	
Hysteresis and Quantized Elements	
Stochastic / Random Elements	
Normal atan Blocks	
Operations with Sign Loss	
Value Flipping and Range Limiting	
Math Function out of Range	
Division by Zero	
Finite State Transition	

Table 2-11 R-ISC-ENV_ISA

* The switches in the system do not affect the correct operation

2.4 Function Specification

2.4.1 Algorithm Abstract

The system is intended to compute the main proprieties of the standard atmosphere: the temperature (T) , the pressure (P), the density (ρ) , the speed of sound (a), the dynamic viscosity (μ) and the kinematic viscosity (ν).

2.4.2 Modeling Assumptions, Scope of Validity & Limitations

- Temperature as a linear function of height in the Troposphere:
  In the model of standard atmosphere, the temperature is a linear function of height;
- Temperature constant in the lower Stratosphere:
  The standard atmosphere have defined temperature constant in the lower Stratosphere;
- The air is homogeneous:
  At heights sufficiently below 86 km the atmosphere is assumed to be homogeneously mixed;
- The air is treated as perfect gas:
  The air is treated as a perfect gas and the total temperature $\boldsymbol{T}$, the total pressure $\boldsymbol{P}$ and total density $\boldsymbol{\rho}$ at any point are related by the perfect state law;
- Temperature and pressure at MSL depend on the climate:
  In order to take into account the climatic conditions, it is possible to change the value of the temperature and pressure at sea level.

2.4.3 Detailed Algorithm Description

2.4.3.1 Defining Constants

The ISA Atmosphere model defines nine basic constants:

- ISA Temperature at MSL: $T_s = 288.15\,K$
- ISA Pressure at MSL: $P_s = 101325\,Pa$
- Specific gas constant for air: $R = 287.058\ \ J/kg \cdot K$
- Gravity constant: $g_0\ = 9.80665\ m/s^2$

- Temperature lapse rate: $\lambda = \frac{dT}{dh}$ $\begin{cases} \lambda = -0.0065\frac{K}{m} & (\mathit{Troposphere}) \\ \lambda = 0\frac{K}{m} & (\mathit{Lower\ Stratosphere}) \end{cases}$
- Heat capacity ratio: $\kappa = 1.4$
- Earth's radius: $r_E = 6356766\,m$
- $\beta = 1.458 \cdot 10^6\,kg/s \cdot m \cdot K^{-1/2}$
- Sutherland constant: $S = 110.4\,m$

2.4.3.2 Geopotential Altitude

All the following considerations and formulas are taken from reference [1].

Viewed from a reference frame fixed in the earth, the atmosphere is subject to the force of gravity. The force of gravity is the vector sum of two forces: the gravitational attraction and the centrifugal force as a consequence of the choice of a frame rotating with the Earth. The gravity field can be derived from the gravity potential energy per unit mass, Φ. This is given by:

$$\Phi = \Phi_G + \Phi_C \qquad \text{2-1}$$

where Φ_G and Φ_C are respectively the potential energy of gravitational attraction and the potential energy associated with the centrifugal force, per unit mass. The gravity, per unit mass, is:

$$\boldsymbol{g} = \nabla\Phi \qquad \text{2-2}$$

where $\nabla\Phi$ is the gradient of the geopotential. The acceleration due to gravity is denoted by g and is defined as the magnitude of $\boldsymbol{g}$**:**

$$g = |\boldsymbol{g}| = |\nabla\Phi| \qquad \text{2-3}$$

Consider two surfaces Φ_1 and Φ_2, infinitely close to each other; moving along an external normal from any point on the surface Φ_1 to a point on the surface Φ_2 , it follows that

$$\Phi_2 = \Phi_1 + d\Phi \qquad \text{2-4}$$

and the incremental work performed by shifting a unit mass from the first surface to the second surface will be:

$$d\Phi = g \cdot dh \qquad \text{2-5}$$

$$\Phi = \int_0^{h^G} g \cdot dh \qquad \text{2-6}$$

Therefore

$$H^G = \frac{\Phi}{g_0} = \frac{1}{g_0} \cdot \int_0^{h^G} g \cdot dh \qquad \text{2-7}$$

$$g_0 \cdot dH^G = g \cdot dh \qquad \text{2-8}$$

$$g = g_0 \cdot \left(\frac{r_E}{r_E + h^G}\right)^2 \qquad \text{2-9}$$

Integration of eq. 2-7, with the substitution of eq. 2-9 for g, yields

$$H^G = \left(\frac{r_E \cdot h^G}{r_E + h^G}\right) \qquad \text{2-10}$$

The transformation from h^G to H^G is necessary for altitude variation between the surface and 86 km.

2.4.3.3 Atmospheric Temperature

Traditionally, in the model of standard atmosphere the temperature is defined as a linear function of height:

$$T = T_{ref} + \lambda \cdot \left(h^G - h_{ref}\right) \qquad \text{2-11}$$

Troposphere

$(0\,m < H^G < 11000\,m)$:

$$T_{ref} = T_0 \quad (2\text{-}12)$$

$$\lambda = -0.0065\frac{K}{m} \quad (2\text{-}13)$$

$$h_{ref} = 0\,m \quad (2\text{-}14)$$

The model allows to change the temperature at MSL with $T_0 = T_s + \Delta T$ and uses the geopotential altitude H^G instead of the geometrical altitude h^G.

The temperature is calculated as:

$$T = T_0\left(1 + \frac{\lambda \cdot H^G}{T_0}\right) \quad (2\text{-}15)$$

Lower Stratosphere

$(11000\,m < H^G < 20000\,m)$:

$$T_{ref} = T_{11} \quad (2\text{-}16)$$

$$\lambda = 0\frac{K}{m} \quad (2\text{-}17)$$

$$h_{ref} = {H^G}_{11} \quad (2\text{-}18)$$

T_{11} is the temperature at $H^G = 11000$ m (${H^G}_{11}$).

The temperature is constant:

$$T = T_{11} \quad (2\text{-}19)$$

The value of T_{11} takes into account the possible change of the temperature at MSL.

2.4.3.4 Atmospheric Pressure

The air is assumed to be dry and, at heights sufficiently below 86 km, the atmosphere is assumed to be homogeneously mixed. The air is treated as a perfect gas and the total temperature T, the total pressure P and total density ρ at any point are related by the perfect state law:

$$P = \rho \cdot R \cdot T \qquad 2\text{-}20$$

where R is the specific gas constant for air.

Within the height region of complete mixing, the atmosphere is assumed to be in hydrostatic equilibrium and to be horizontal stratified so that dP, the differential of pressure, is related to dh , the differential of geometric height, by the relationship:

$$dP = -g_0 \cdot \rho \cdot dh \qquad 2\text{-}21$$

The eq. 2-21, with the substitution of eq. 2-20 for ρ, yields:

$$\frac{dP}{P} = -\frac{g_0}{RT}\,dh \qquad 2\text{-}22$$

The eq. 2-22, after the substitution of eq. 2-11 for T, yields:

$$\frac{dP}{P} = -\frac{g_0}{R\left[T_{ref} + \lambda \cdot \left(h^G - h_{ref}\right)\right]}\,dh \qquad 2\text{-}23$$

<u>Troposphere</u>

$(\,0\,m < H^G < 11000\,m\,)$:

The eq. 2-23, after the substitution of eq. 2-12 and 2-14, yields:

$$\frac{dP}{P} = -\frac{g_0}{R\,[T_0 + \lambda \cdot (h^G)]}\,dh \qquad 2\text{-}24$$

Hence,

$$\int_{P_0}^{P}\frac{dP}{P} = \int_0^{H^G} -\frac{g_0}{R\,[T_0 + \lambda \cdot (h^G)]}\,dh \qquad 2\text{-}25$$

The model allows to change the temperature at MSL with $P_0 = P_s + \Delta P$.

The pressure is calculated as:

$$\frac{P}{P_0} = \left[\frac{T_0}{T_0 + \lambda \cdot H^G}\right]^{-\frac{g_0}{\lambda \cdot R}} \mapsto \frac{P}{P_0} = \left[\frac{T}{T_0}\right]^{-\frac{\lambda \cdot R}{g_0}} \quad 2\text{-}26$$

Lower Stratosphere

$(11000\, m < H^G < 20000\, m)$:

The eq. 2-23, after substitution of eq. 2-16, 2-17 and 2-18, yields:

$$ln\left(\frac{P}{P_{11}}\right) = -\frac{g_0}{R\,[T_{11}]}\,(H^G - H^G{}_{11}) \quad 2\text{-}27$$

The value of P_{11} takes into account the possible change of the pressure at MSL.

2.4.3.5 Atmospheric Density

The atmospheric density is calculated by the perfect state law 2-20, using the values of temperature and pressure calculated by the above formulas.

2.4.3.6 Speed of Sound

The formula adopted for the speed of sound a is:

$$a = \sqrt{(\kappa \cdot R \cdot T)} \quad 2\text{-}28$$

where κ is the ratio between the specific heat of air at constant pressure and the specific heat of air at constant volume and is taken to be exactly equals to 1.4 (dimensionless).

2.4.3.7 Dynamic Viscosity

The coefficient of dynamic viscosity μ is defined as a coefficient of internal friction developed where gas regions move adjacent to each other at different velocities. The following expression uses constants derived from experiment:

$$\mu = \frac{\beta \cdot T^{3/2}}{T + S} \qquad \text{2-29}$$

where $\beta = 1.458 \cdot 10^{-6} \frac{kg}{s \cdot m \cdot K^{1/2}}$ and *S* is the Sutherland's constant equal to $110.4\ K$.

2.4.3.8 Kinematic Viscosity

The kinematic viscosity ν is defined as the ratio between the dynamic viscosity of a gas and the density of that gas, that is:

$$\nu = \frac{\mu}{\rho} \qquad \text{2-30}$$

2.4.3.9 Algorithm for Implementation

In the implemented system, the equations from 2-10 to 2-20 and equations from 2-26 to 2-30 are used.

2.5 Architecture Specification

2.5.1 Parent / Child Systems

2.5.1.1 Parent System

The system will be embedded into the system “Atmosphere”, which will be embedded into the parent system "Environment", whose purpose it is to simulate all processes regarding the environment of the aircraft (atmosphere, terrain model, earth model, etc.).

2.5.1.2 Child Systems

This system does not contain any child systems.

2.5.2 Signal Definitions

In the following tables essential information about input and output signals are collected.

2.5.2.1 Inputs

Inputs							
Symbol	**Name**	**Size**	**Components**	**Data Type**	**Min**	**Max**	**Description**
h^G	h_G_WGS84_m	[1 1]	-	double	-500	20000	Height of the aircraft's center of gravity above the WGS84 reference ellipsoid.
ΔT	delta_T_K	[1 1]	-	double	-100	100	Temperature change at MSL to take into account climatic conditions
ΔP	dellta_P_Pa	[1 1]	-	double	-5000	5000	Pressure change at MSL to take into account climatic conditions

Table 2-12 Inputs

2.5.2.2 Outputs

Outputs							
Symbol	**Name**	**Size**	**Components**	**Data Type**	**Min**	**Max**	**Description**
T	T_K	[1 1]	-	double	0	-	Atmospheric temperature relative to the geopotential altitude.
P	P_Pa	[1 1]	-	double	0	-	Atmospheric pressure relative to the geopotential altitude.
ρ	rho_kgDm3	[1 1]	-	double	0	-	Atmospheric density relative to the geopotential altitude.
a	a_mDs	[1 1]	-	double	0	-	Atmospheric speed of sound relative to the geopotential altitude.
μ	mu_sPa	[1 1]	-	double	-	-	Atmospheric dynamic viscosity relative to the geopotential altitude.
ν	nu_m3sPaDkg	[1 1]	-	double	-	-	Atmospheric kinematic viscosity relative to the geopotential altitude.

Table 2-13 Outputs

2.5.2.3 Bus Structure

To facilitate the transport of signals, were often created the buses.

In the following tables, buses created for input and output signals are then collected.

Inputs

L	Bus Name	Elements	Element Types
0	*pos_G_WGS84_Bus*	lambda_G_WGS84_rad phi_G_WGS84_rad h_G_WGS84_m	double double double
0	*External_Inputs_ISA_Bus*	delta_T_K delta_P_Pa	double double

Table 2-14 Inputs Bus Structure

Outputs

L	Bus Name	Elements	Element Types
0	*ISA_Variables_H_G_Bus*	T_K P_Pa rho_kgDm3 a_mDs mu_sPa nu_ m3sPaDkg	double double double double double double

Table 2-15 Outputs Bus Structure

The buses allow to select only the necessary signal.

2.6 Structural Layout

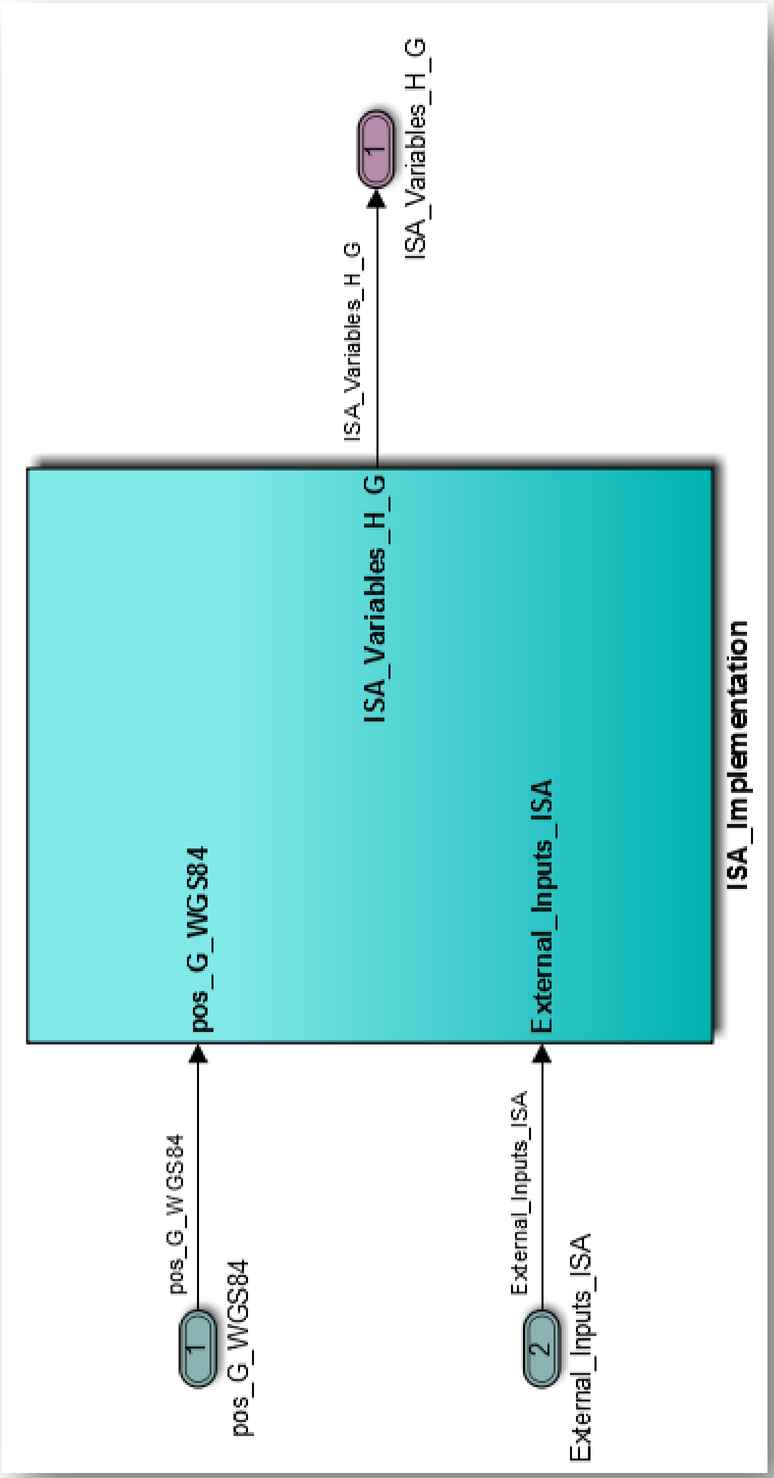


Figure 2-1 L0: ENV_ISA

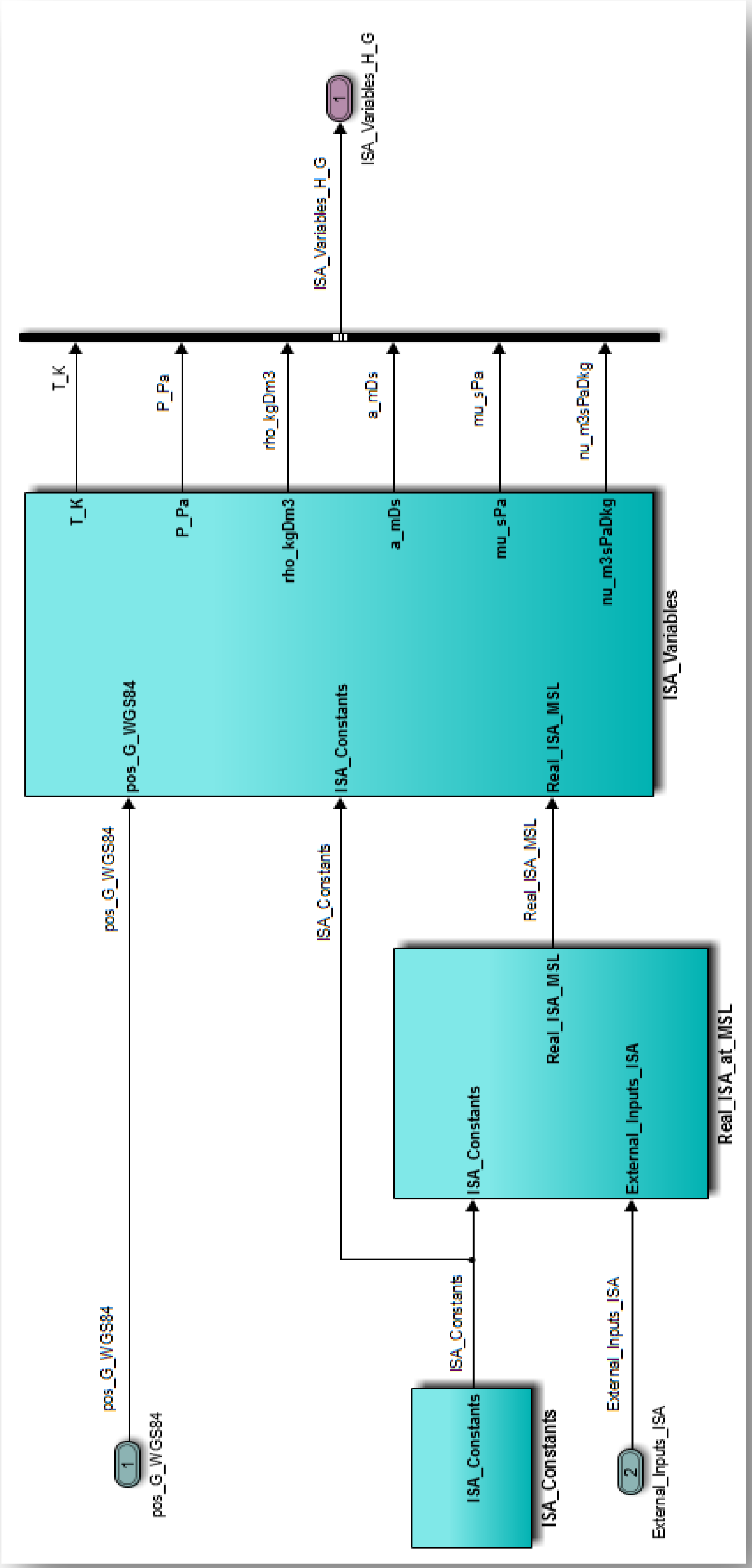


Figure 2-2 L1: ISA_Implementation

Figure 2-3 L2: ISA_Constants

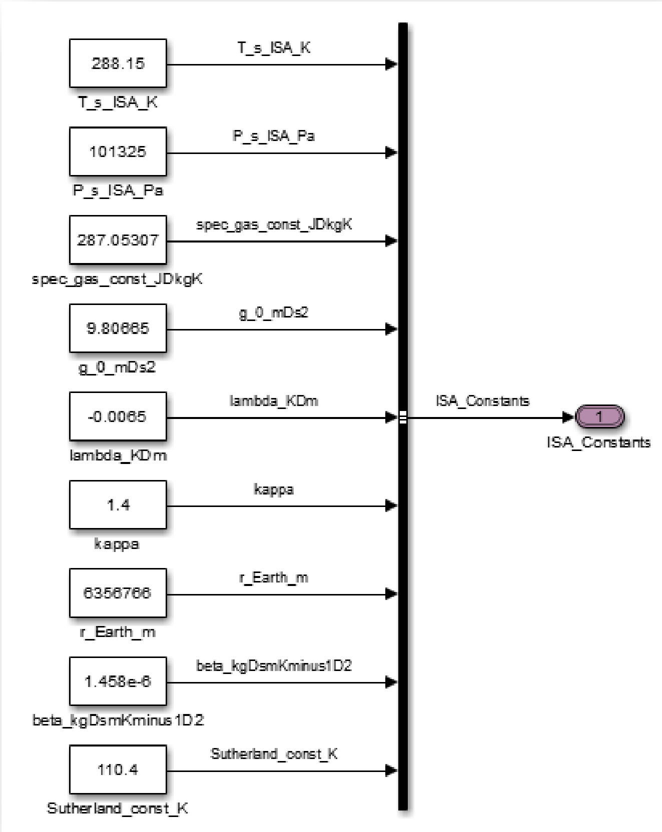
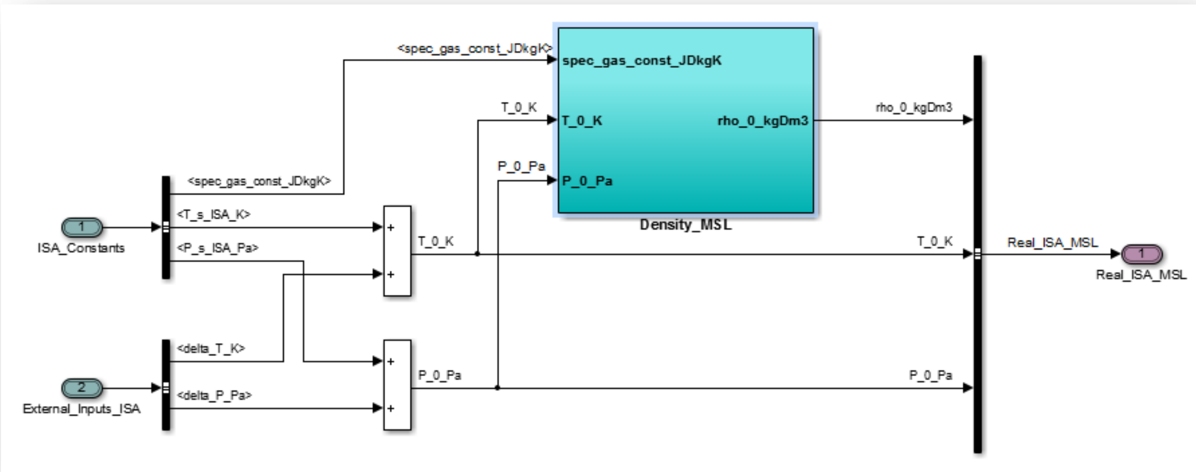


Figure 2-4 L2: Real_ISA_at_MSL



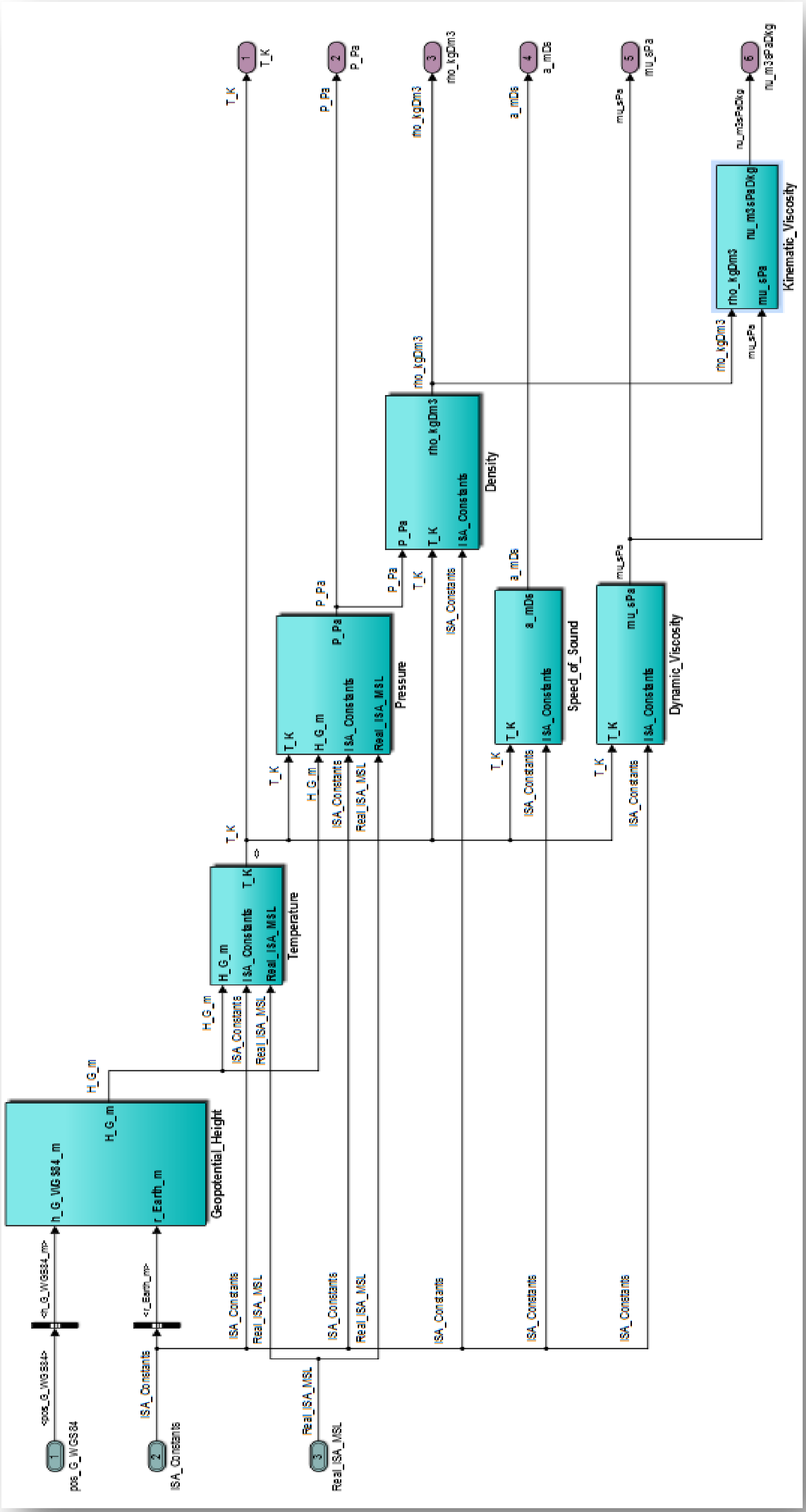


Figure 2-5 L2: ISA_Variables

Figure 2-8 L3: Density_MSL

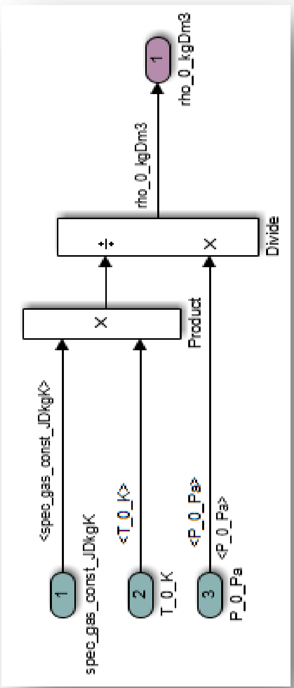


Figure 2-6 L3: Geopotential_Height

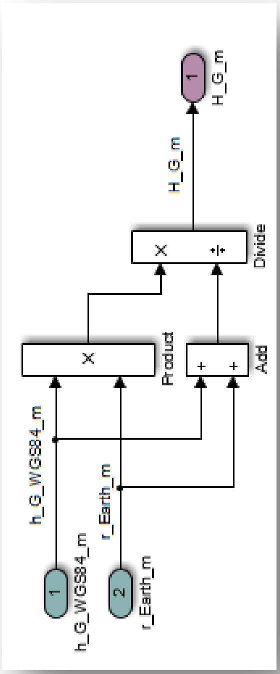
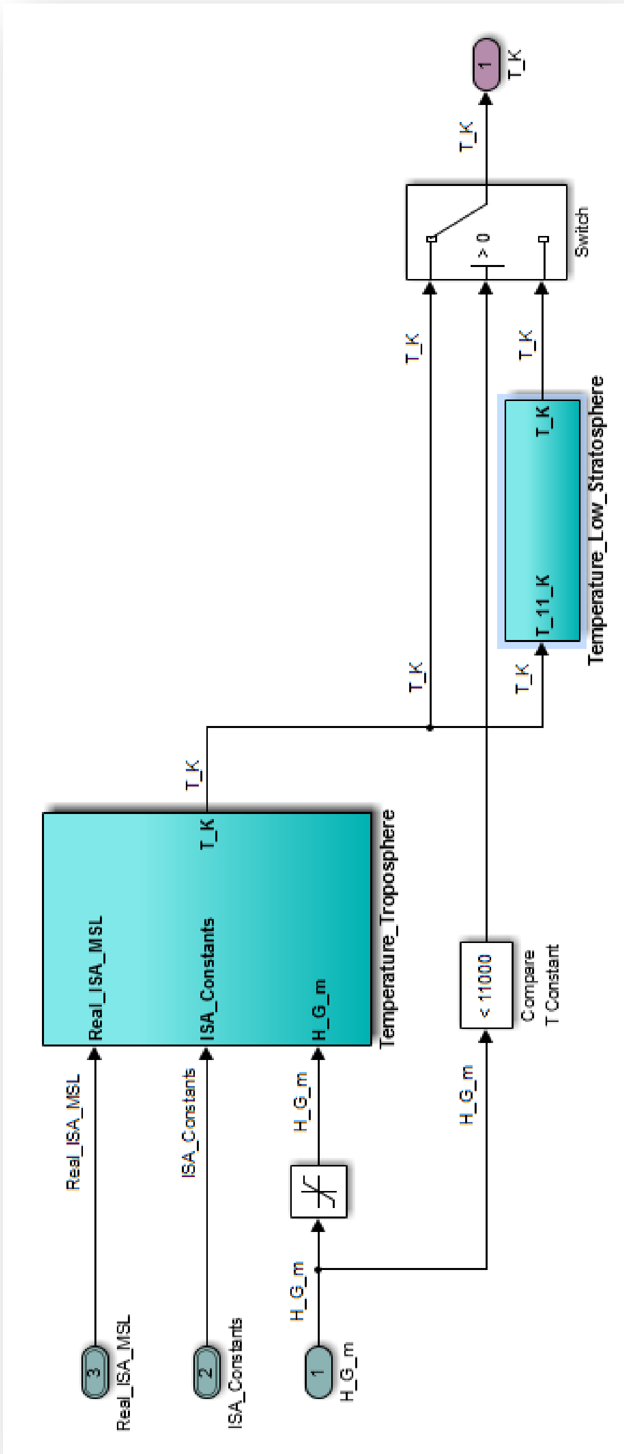


Figure 2-7 L3: Temperature



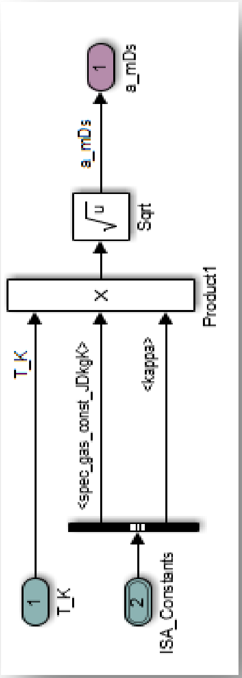


Figure 2-9 L3: Speed_of_Sound

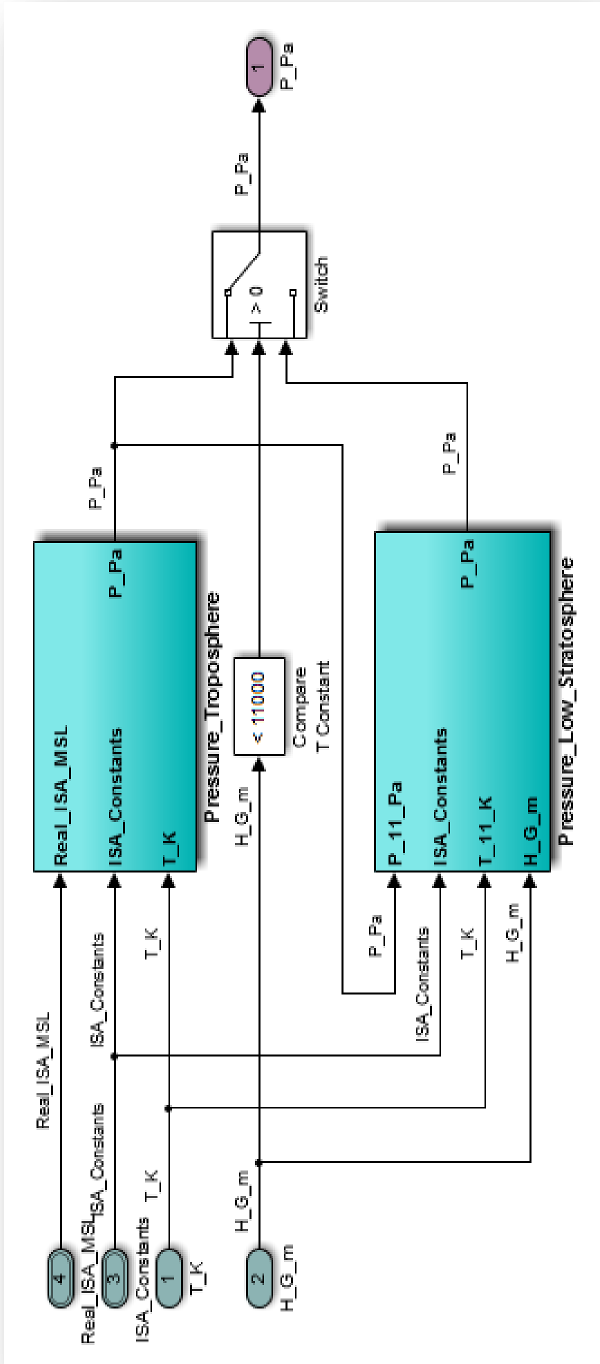


Figure 2-10 L3: Pressure

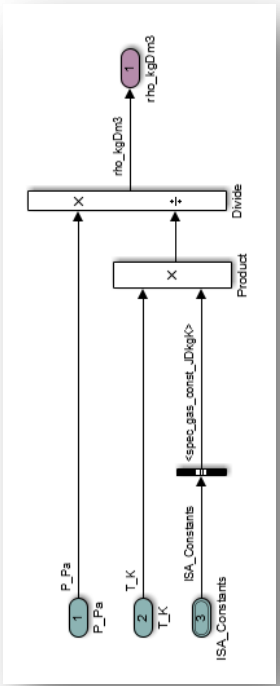


Figure 2-11 L3: Density

Figure 2-12 L3: Dynamic_Viscosity

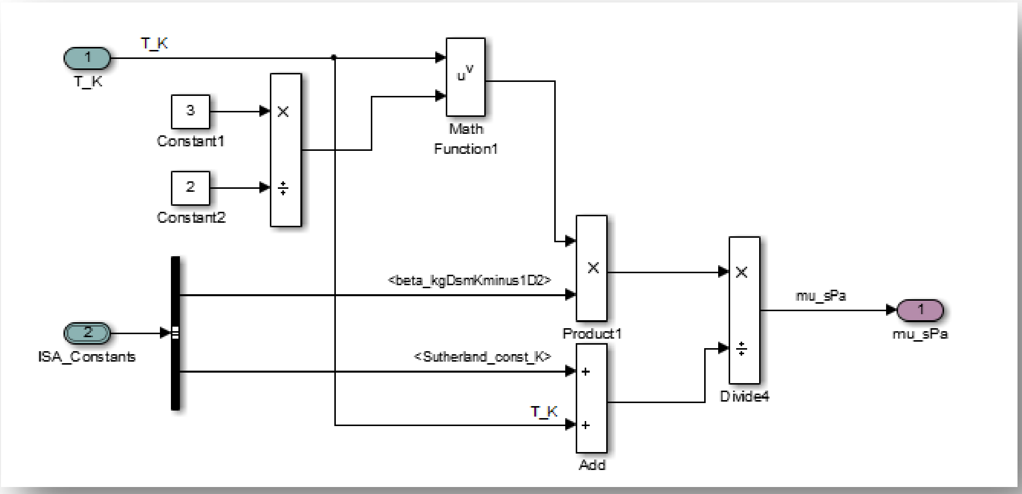


Figure 2-13 L3: Kinematic Viscosity

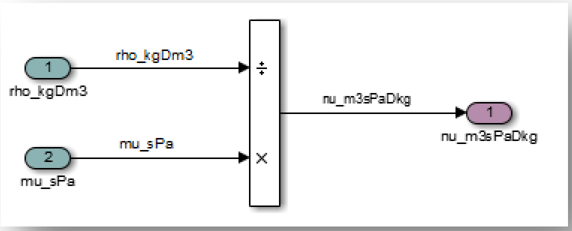


Figure 2-14 L4: Temperature_Troposphere

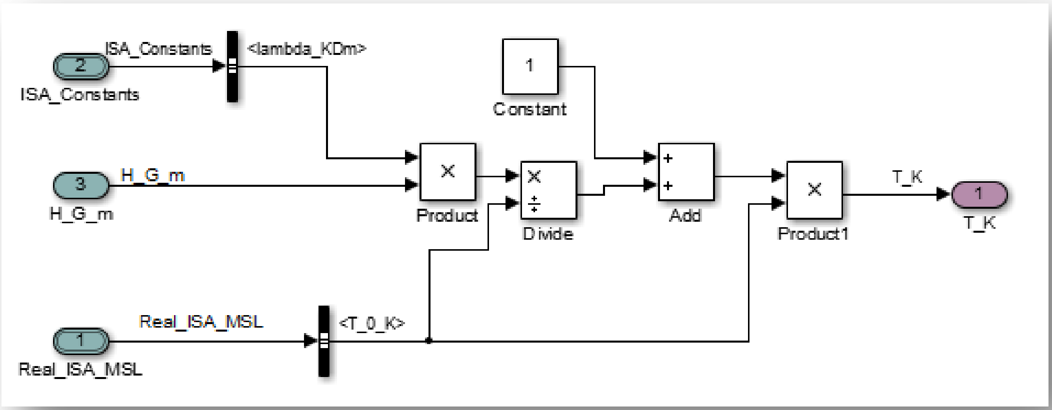


Figure 2-15 L4:Temperature_Low_Stratosphere

Figure 2-16 L4: Pressure_Troposphere

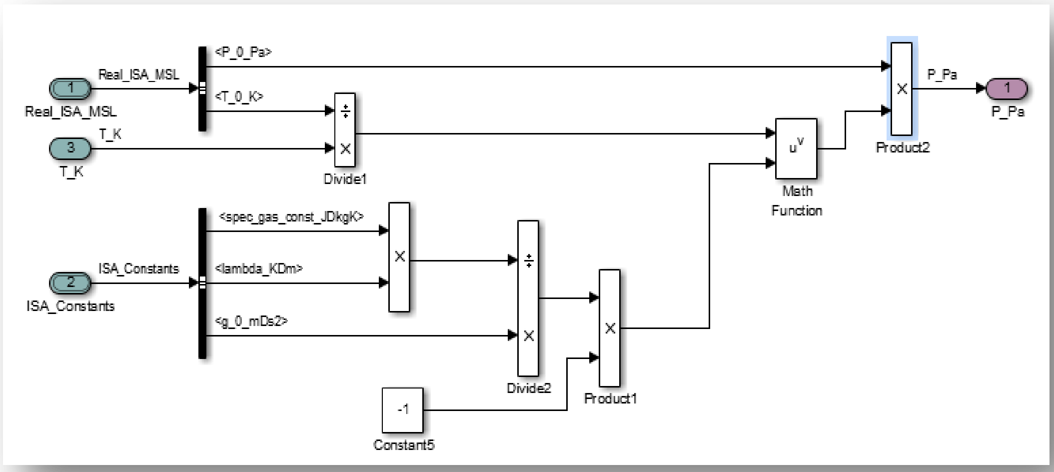
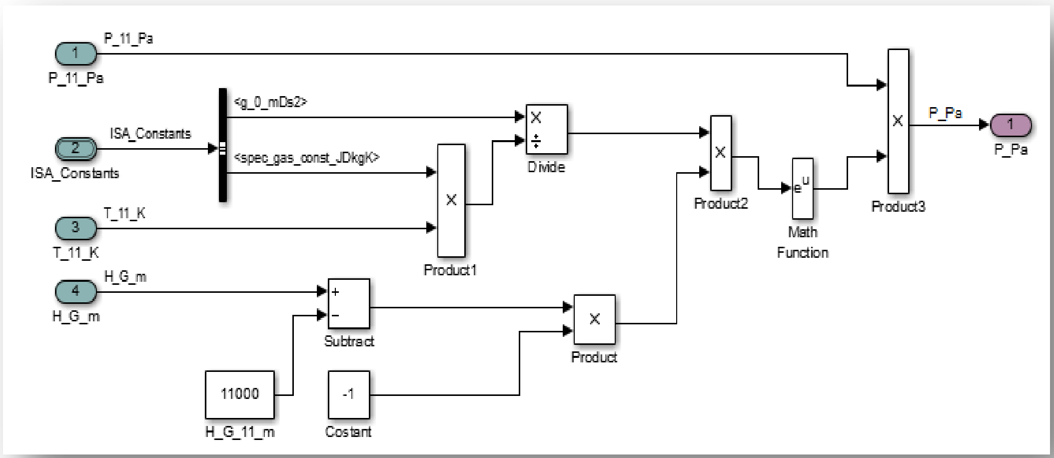


Figure 2-17 L4: Pressure_Low_Stratosphere



2.7 Verification Plan

2.7.1 Methods Used for Verification

2.7.1.1 Methods for Testing Functional Requirements

Requirement Name and ID	Description of Verification Method
R-FUN-ENV_ISA_01 Computation of Temperature	Correct computation of Temperature demonstrated in ENV_ISA-Nominal_TC1, comparison to dissimilar implementation in ENV_ISA-Nominal_TC2 (see section 2.7.2.1).
R-FUN-ENV_ISA_02 Computation of Pressure	Correct computation of Pressure demonstrated in ENV_ISA-Nominal_TC1, comparison to dissimilar implementation in ENV_ISA-Nominal_TC2 (see section 2.7.2.1).
R-FUN-ENV_ISA_03 Computation of Density	Correct computation of Density demonstrated in ENV_ISA-Nominal_TC1, comparison to dissimilar implementation in ENV_ISA-Nominal_TC2 (see section 2.7.2.1).
R-FUN-ENV_ISA_04 Computation of Speed of Sound	Correct computation of Speed of Sound demonstrated in ENV_ISA-Nominal_TC1, comparison to dissimilar implementation in ENV_ISA-Nominal_TC2 (see section 2.7.2.1).
R-FUN-ENV_ISA_05 Computation of Dynamic Viscosity	Correct computation of Dynamic Viscosity demonstrated in ENV_ISA-Nominal_TC1, comparison to dissimilar implementation in ENV_ISA-Nominal_TC2 (see section 2.7.2.1).
R-FUN-ENV_ISA_06 Computation of Kinematic Viscosity	Correct computation of Kinematic Viscosity demonstrated in ENV_ISA-Nominal_TC1, comparison to dissimilar implementation in ENV_ISA-Nominal_TC2 (see section 2.7.2.1).

Table 2-16 Methods for testing Functional Requirements

The above table gives the information about the checks that have been performed on the system. Each verification is indicated by a name and refers to a certain requirement, easily identifiable through the ID.

2.7.1.2 Methods for Testing Implementation Requirements

Requirement Name and ID	Description of Verification Method
R-NUM-ENV_ISA Numeric Efficiency	Manual review of the implemented model. Records according to section 2.8.1.1.
R-IOC-ENV_ISA Input / Output Interface Compliance to Parent System	Compliance to parent system will be verified at integration with parent system.
R-SGC-ENV_ISA Implementation Compliance to FSD Style Guides	Manual review of the implemented model. Records according to section 2.8.1.2
R-ISC-ENV_ISA Implementation Standards Compliance	Manual review of the implemented model. Records according to section 2.8.1.3.

Table 2-17 Methods for Testing Implementation Requirements

The verification methods underlined, are considered to be verified at the time of assembly of all systems.

2.7.1.3 Methods for Testing Operational Requirements

Derived Standard Requirement Matrix:		
SIMULINK Modes	Simulink Offline Simulation	Demonstrated during test ENV_ISA-Nominal_TC1, ENV_ISA-Nominal_TC2 and ENV_ISA-Operational_TC1 (see sections 2.7.2.1 and 2.7.3.1).
	Simulink Pseudo Real Time Simulation	Will be demonstrated on a higher level of the simulation model.
External Control for SIMULINK Execution	Bypassing of Non-Autonomous Elements	Not applicable as there are no non-autonomous elements present in the model.
	Single Point Execution	Demonstrated during test ENV_ISA-Nominal_TC1, ENV_ISA-Nominal_TC2 and ENV_ISA-Operational_TC1 (see sections 2.7.2.1 and 2.7.3.1).
	Online Integration Freeze and Reset	Not applicable as there are no integrators present in the model.
	Workspace Initialization	Not applicable as there are no variables present to be initialized in the workspace.
	Runtime Parameter Tuning	Not applicable as there are no tunable parameters present in the model
RTW Code Generation	S-function	Generation of S-function demonstrated during test ENV_ISA-Operational_TC2 (see section 2.7.3.2).
Code Modes	Stand-alone Batch Simulation	Will be demonstrated on a higher level of the simulation model.
	Stand-alone Real Time Simulation	Will be demonstrated on a higher level of the simulation model.

Table 2-18 Methods for Testing Operational Requirements

2.7.2 Verification Plan for Functional Requirements

2.7.2.1 Nominal Testing Procedure

Test Name:	**Correct Calculation of ISA Variables**
Test ID:	ENV_ISA-Nominal_TC1
Related Requirements:	R-FUN-ENV_ISA_01: Computation of Temperature R-FUN-ENV_ISA_02: Computation of Pressure R-FUN-ENV_ISA_03: Computation of Density R-FUN-ENV_ISA_04: Computation of Speed of sound R-FUN-ENV_ISA_05: Computation of Dynamic Viscosity R-FUN-ENV_ISA_06: Computation of Kinematic Viscosity
Verification Data:	See section 2.8.2.1

The implemented simulation model is excited with different input combinations.

Afterwards, the course of the Temperature, of the Pressure, of the Density, of the Speed of Sound, of the Dynamic Viscosity and of the Kinematic Viscosity are plotted over the geopotential height. The following figure shows the scheme used for the test:

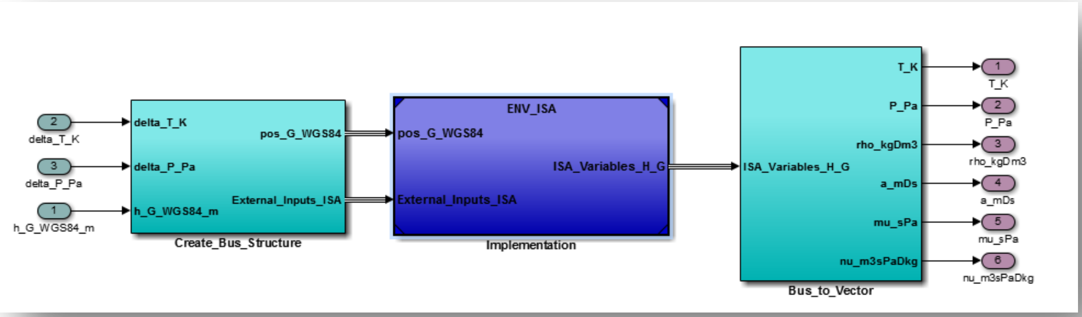


Figure 2-18 ENV_ISA-Nominal_TC1

It was necessary to create, first of all, the buses to generate inputs and then select the individual signals from the output bus. The test is then put into operation using a Matlab script. To perform the test, the input signals have been made change in the following ranges:

- ΔT: from $-20\ K$ to $20\ K$
- ΔP: from $-3000\ Pa$ to $3000\ Pa$
- h^G: from 0 to 20000 m

Test Name:	**Equivalence of Implementation and Dissimilar Implementation**
Test ID:	ENV_ISA-Nominal_TC2
Related Requirements:	R-FUN-ENV_ISA_01: Computation of Temperature R-FUN-ENV_ISA_02: Computation of Pressure R-FUN-ENV_ISA_03: Computation of Density R-FUN-ENV_ISA_04: Computation of Speed of Sound R-FUN-ENV_ISA_05: Computation of Dynamic Viscosity R-FUN-ENV_ISA_06: Computation of Kinematic Viscosity
Verification Data:	See section 2.8.2.1

The implemented model and a dissimilar implementation (Embedded Matlab Function) are both excited with the same input signals. Afterwards, the output is checked for deviations. As long as the relative deviations stay below a certain threshold both implementations are considered equivalent and the test is passed.

The following figure shows the scheme used for the test:

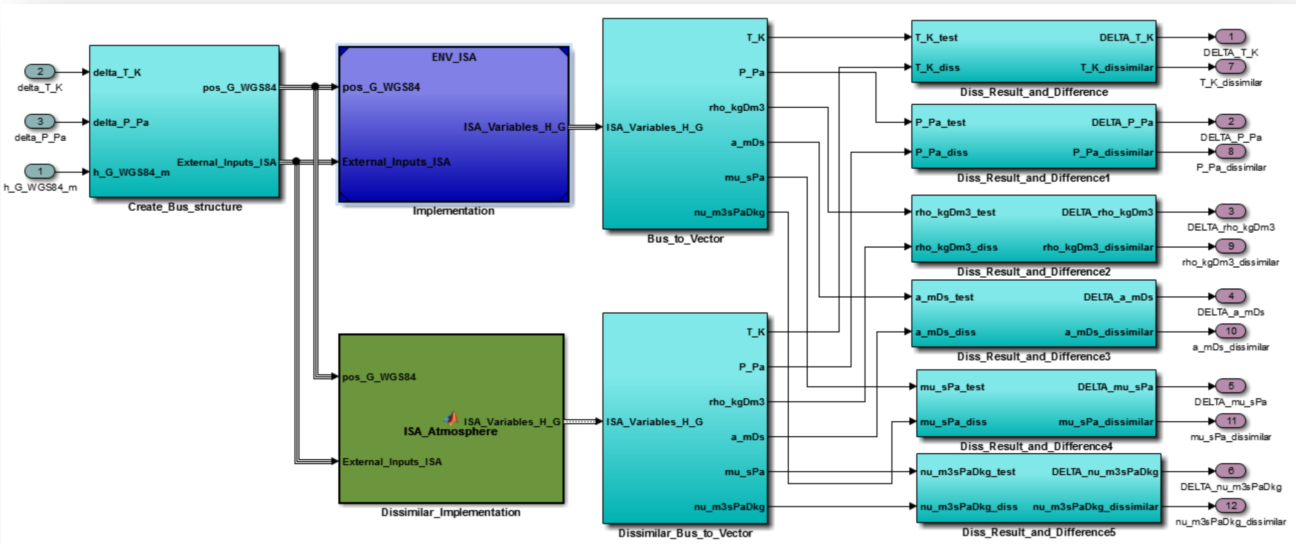


Figure 2-19 ENV_ISA-Nominal_TC2

To perform the test, the input signals have been made change in the following ranges:

- ΔT: from $-20\ K$ to $20\ K$
- ΔP: from $-3000\ Pa$ to $3000\ Pa$
- h^G: from 0 to 20000 m

2.7.3 Verification Plan for Operational Requirements

2.7.3.1 Point Execution Reproducibility and Determinism Testing

Test Name:	**Point Execution Reproducibility and Determinism Test**
Test ID:	ENV_ISA-Operational_TC1
Related Requirements:	R-OPS-ENV_ISA: Operation Standard Requirements Matrix
Verification Data:	See section 2.8.3.1

The implemented simulation model is excited with different input combinations. Afterwards, the calculation is repeated with the inputs in reverse order and with varying step sizes. The purpose of this test is to check for hidden non-autonomous elements in the model.

As long as the relative deviations between the forward and backward runs and the multistep runs stay below a certain threshold the implementations are considered as being equivalent.

The following figure shows the scheme used for the test:

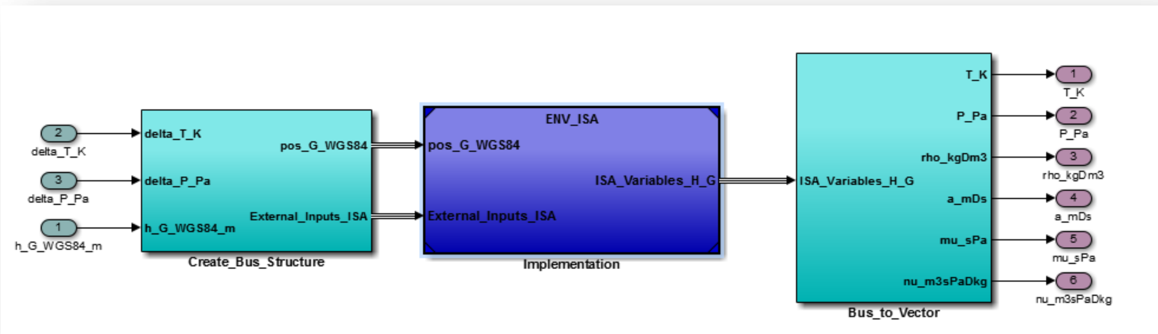


Figure 2-20 ENV_ISA-Operational_TC1

The inputs to the system are varied randomly between the following bounds:

- ΔT: from $-100\ K$ to $100\ k$
- ΔP: from $-5000\ Pa$ to $5000\ Pa$
- h^{G}: from -500 to 20000 m

2.7.3.2 Code Generation and Equivalence Testing

Test Name:	**Code Generation and Equivalence Testing**
Test ID:	ENV_ISA-Operational_TC2
Related Requirements:	R-OPS-ENV_ISA: Operation Standard Requirements Matrix
Verification Data:	See section 2.8.3.2

The implemented simulation model running in normal mode as well as a compiled version (SIL) and a S-function are excited with random inputs signals.

Afterwards the outputs are checked for deviations. As long as the deviations stay below a certain threshold the test is passed.

The following figure shows the scheme used for the test:

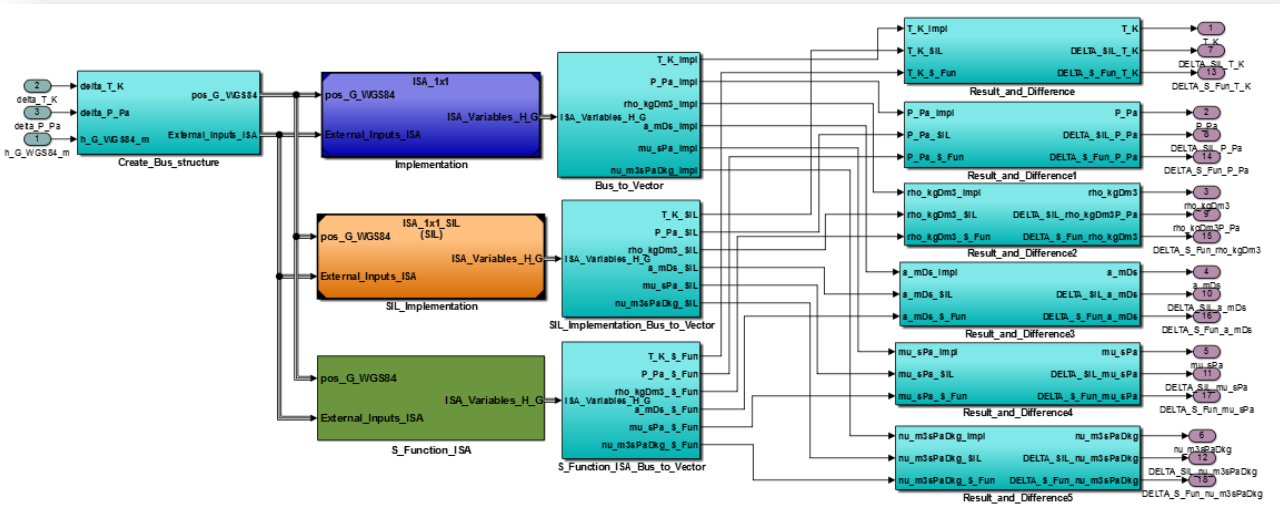


Figure 2-21 ENV_ISA-Operational_TC2

The inputs to the system are varied randomly between the following bounds:

- ΔT: from $-100\ K$ to $100\ k$
- ΔP: from $-5000\ Pa$ to $5000\ Pa$
- h^G: from -500 to 20000 m

2.8 Verification Data

2.8.1 Verification of Implementation Requirements

2.8.1.1 Numeric Efficiency

Requirement Name **Numeric Efficiency**	Requirement ID R-NUM-ENV_ISA
Requirement is violated if the model contains one or more of the following items:	
Unused / Dead Code Branches Description	YES ☐ NO ☒
Computational Redundancies Description	YES ☐ NO ☒
Matrix Inversions Description	YES ☐ NO ☒
Scalar Expansions of Vector/Matrix Math Description	YES ☐ NO ☒
Circle Computations Description	YES ☐ NO ☒
Inefficient Lookup Table Programming Description	YES ☐ NO ☒
Algebraic Loops Description	YES ☐ NO ☒
Numeric Efficiency met?	**YES ☒ NO ☐**

Table 2-19 R-NUM-ENV_ISA

2.8.1.2 Implementation Compliance to FSD Style Guidelines

Requirement Name **Implementation Compliance to FSD Style Guidelines**	Requirement ID R-SGC-ENV_ISA	
Requirement is violated if the model contains other blocks than the specified ones.		
Non-Specified Blocks within the Model? Description	YES ☐	NO ☒
Style Guide Compliance met?	**YES ☒**	**NO ☐**

Table 2-20 R-SGC-ENV_ISA

2.8.1.3 Implementation Standards Compliance

Requirement Name **Implementation Standards Compliance**	Requirement ID R-ISC-ENV_ISA	
The coded algorithm must not contain any programming technique listed below unless detailed justification substantiates indispensability.		
*Discrete Switches**	YES ☒	NO ☐
Memory Blocks	YES ☐	NO ☒
Time Delays	YES ☐	NO ☒
Time Dependent / Non-Autonomous Elements	YES ☐	NO ☒
In-lined Integrations	YES ☐	NO ☒
Hysteresis and Quantized Elements	YES ☐	NO ☒
Stochastic / Random Elements	YES ☐	NO ☒
Normal atan Blocks	YES ☐	NO ☒
Operations with Sign Loss	YES ☐	NO ☒
Value Flipping and Range Limiting	YES ☐	NO ☒
Math Function out of Range	YES ☐	NO ☒
Division by Zero	YES ☐	NO ☒
Finite State Transition	YES ☐	NO ☒
Implementation Standards Compliance met?	**YES ☒**	**NO ☐**

Table 2-21 R-ISC-ENV_ISA

* The switches in the system do not affect the correct operation

2.8.2 Verification of Functional Requirements

2.8.2.1 Results for Nominal Testing

Test Name:	**Correct Calculation of the Variables over Geopotential Height**
Test ID:	ENV_ISA-Nominal_TC1
Verification Plan:	See section 2.7.2.1

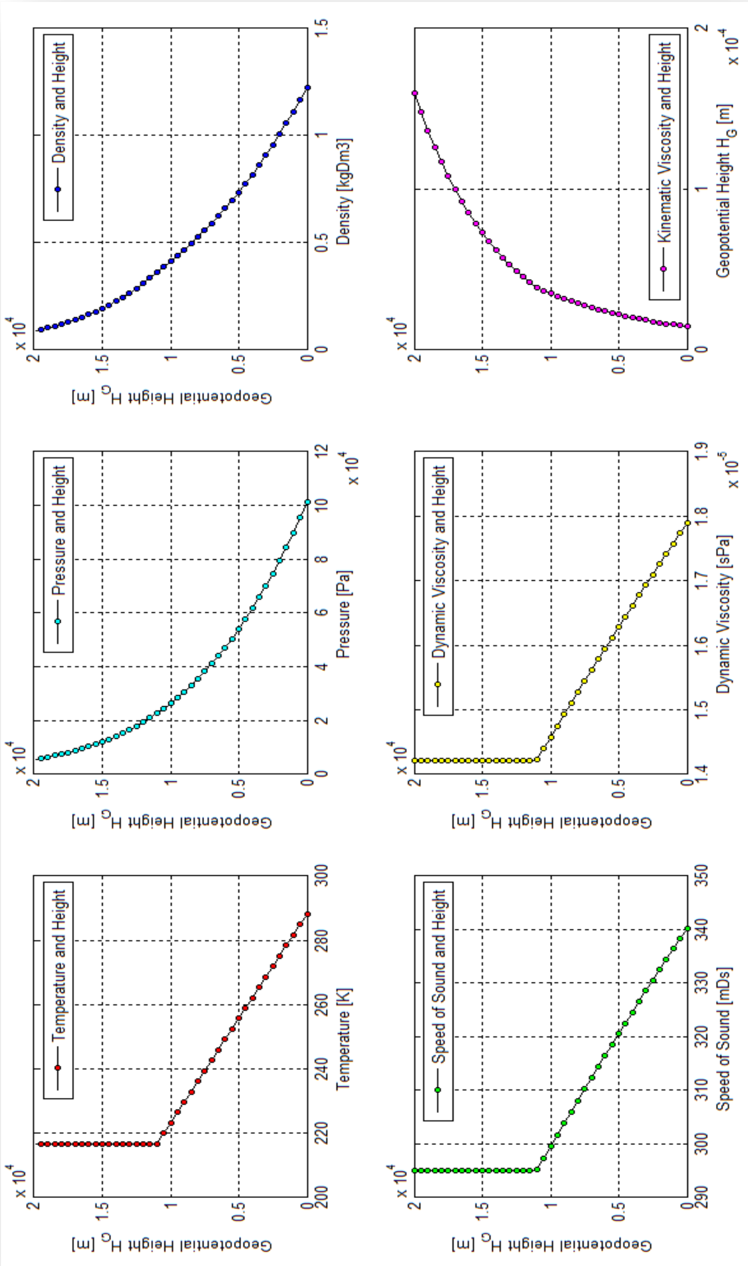


Figure 2-22 Calculation of the Variables over Geopotential Height

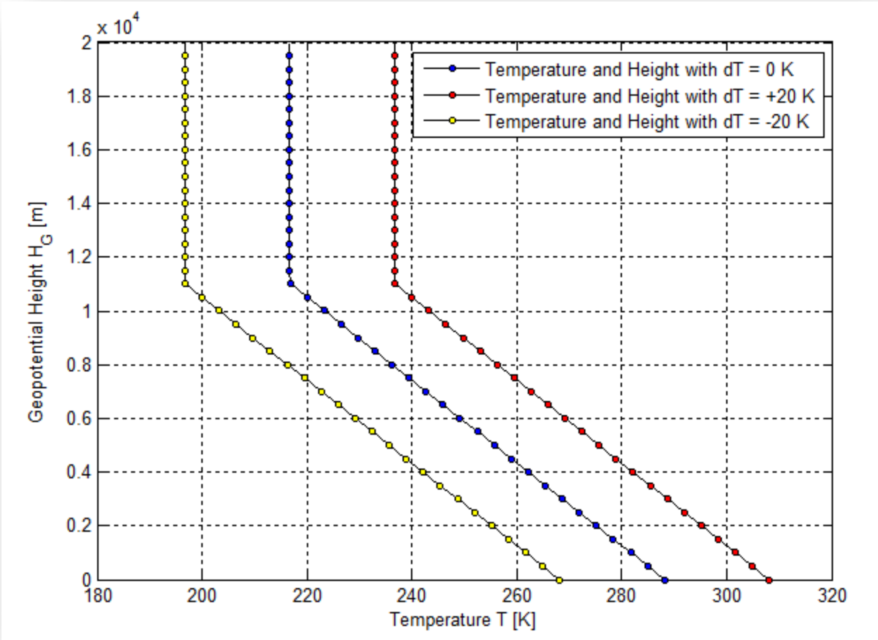


Figure 2-23 Calculation of the Temperature with dT = $\pm 20\ K$ **over Geopotential Height**

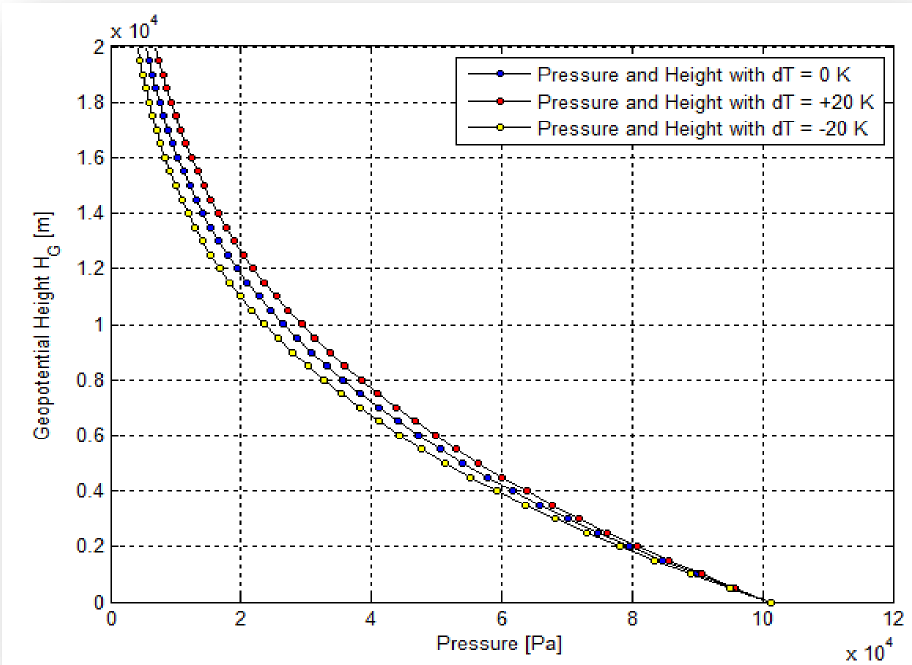


Figure 2-24 Calculation of the Pressure with dT = $\pm 20\ K$ **over Geopotential Height**

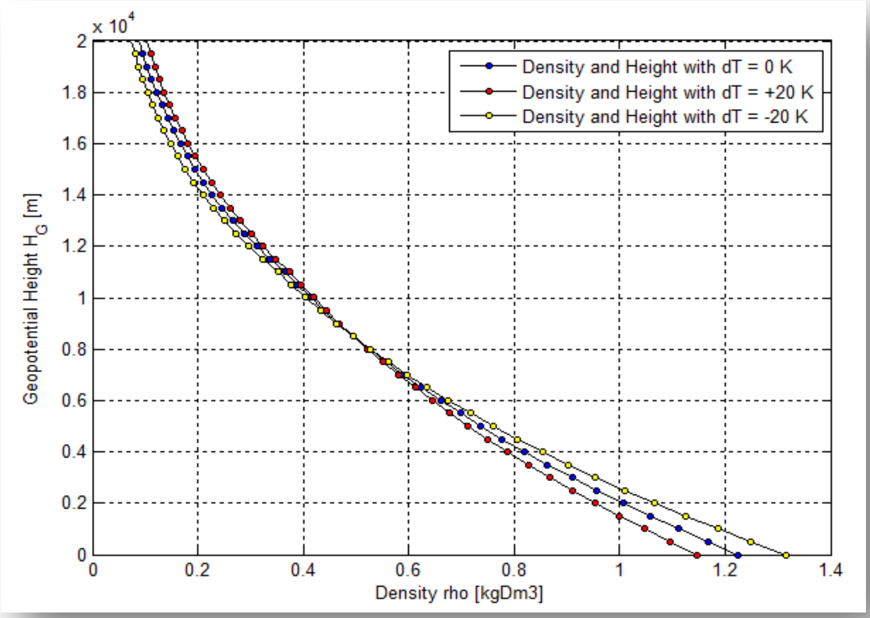


Figure 2-25 Calculation of the Density with dT = $\pm 20\ K$ **over Geopotential Height**

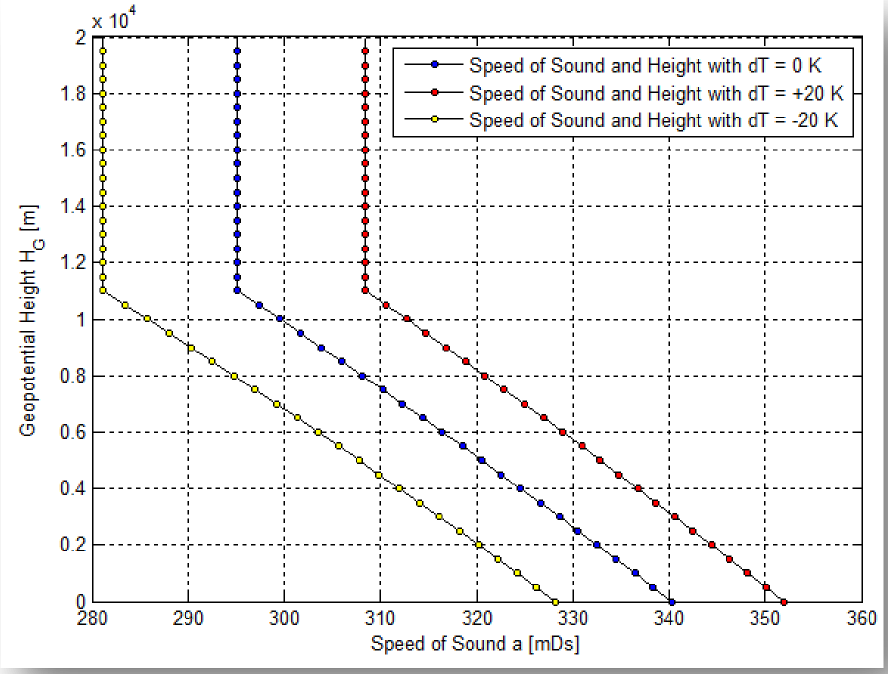


Figure 2-26 Calculation of the Speed of Sound with dT = $\pm 20\ K$ **over Geopotential Height**

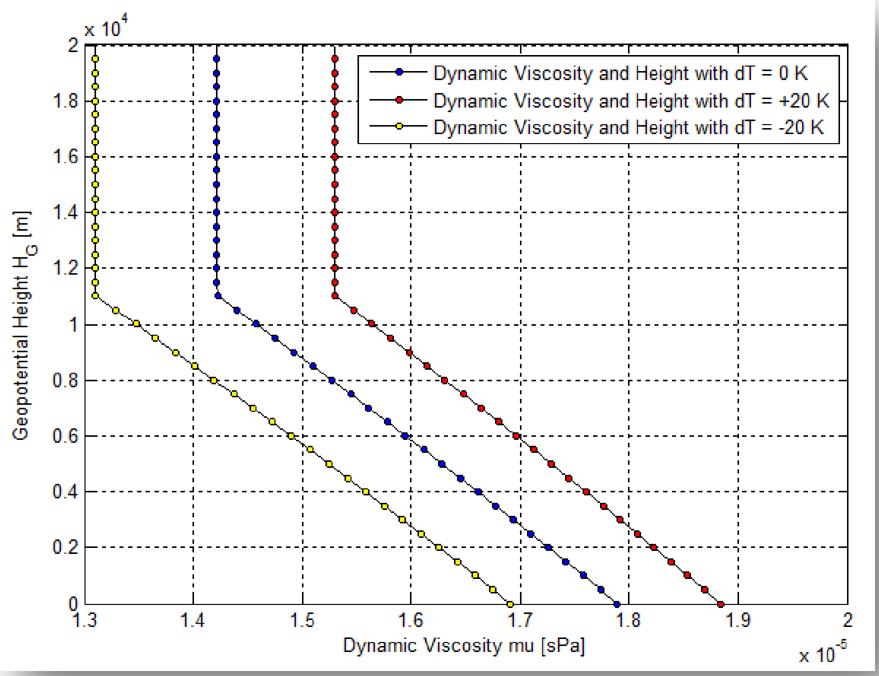


Figure 2-27 Calculation of the Dynamic Viscosity with dT = $\pm 20\ K$ over Geopotential Height

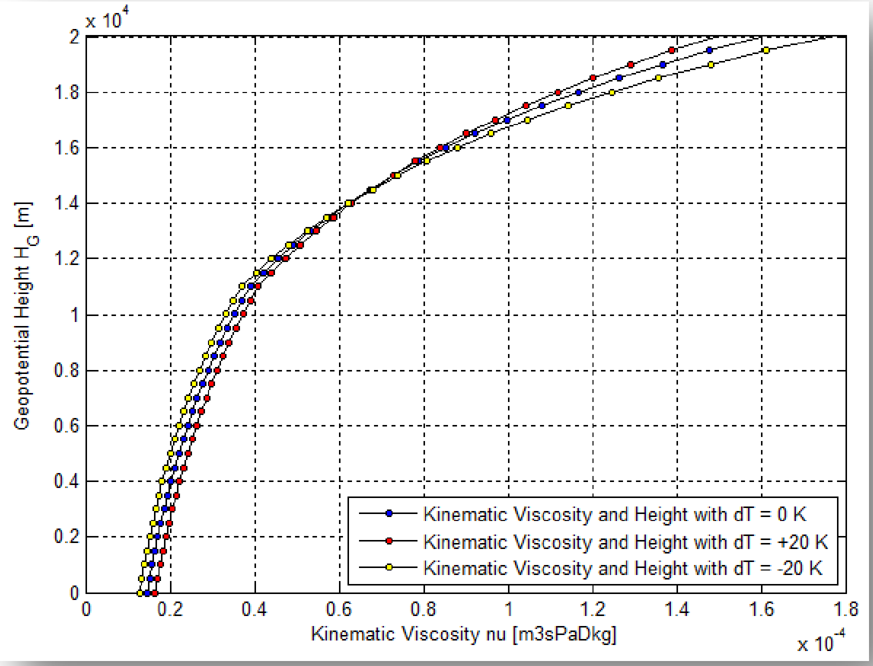


Figure 2-28 Calculation of the Kinematic Viscosity with dT = $\pm 20\ K$ over Geopotential Height

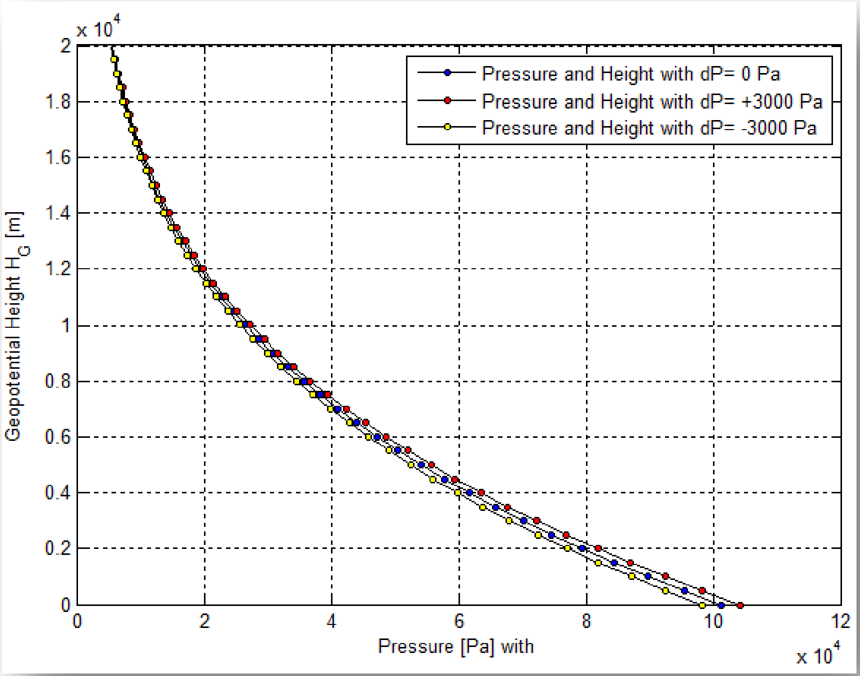


Figure 2-29 Calculation of the Pressure with dP = $\pm 3000\ Pa$ over Geopotential Height

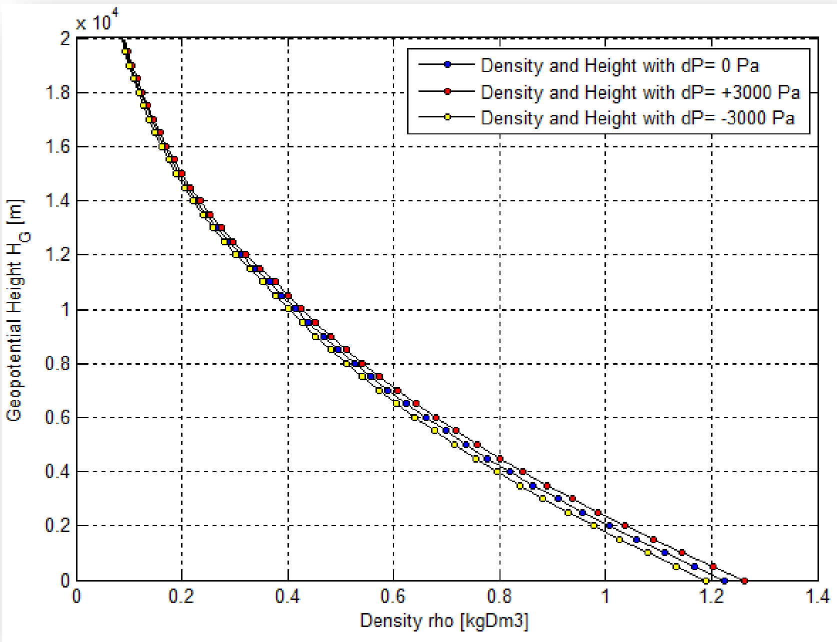


Figure 2-30 Calculation of the Density with dP = $\pm 3000\ Pa$ over Geopotential Height

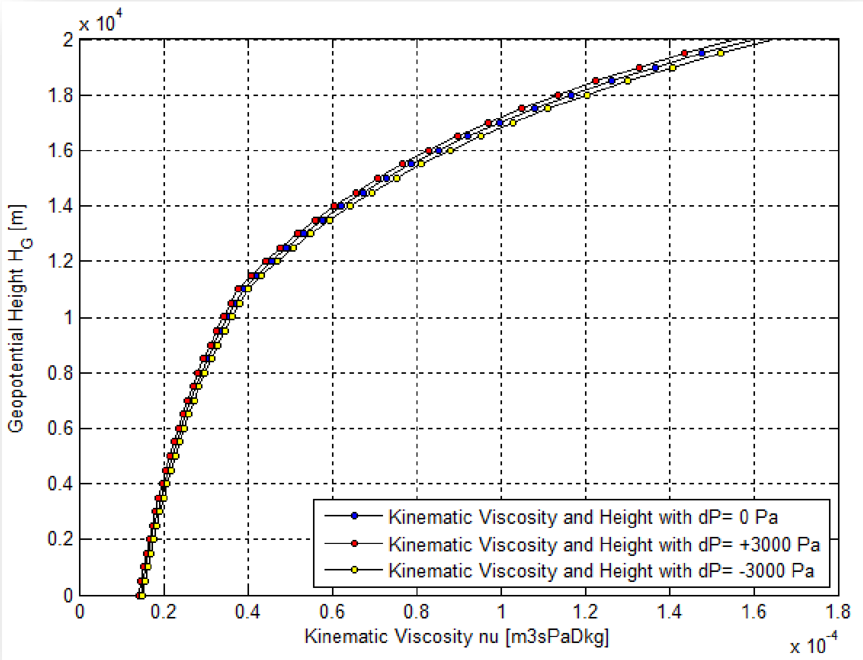


Figure 2-31 Calculation of the Kinematic Viscosity with dP = $\pm 3000\ Pa$ **over Geopotential Height**

Test Name:	**Equivalence of Implementation and Dissimilar Implementation**
Test ID:	ENV_ISA-Nominal_TC2
Verification Plan:	See section 2.7.2.1
Number of Test Points:	1,000,000
Execution Time:	84.16s
Rel. Deviation Threshold:	1.00e-13

All Deviations below Threshold — YES ☒ NO ☐

Absolute Deviations		**Relative Deviations**	
Average Absolute Deviation:	6.84e-13	*Average Relative Deviation:*	2.80e-17
Maximum Absolute Deviation:	7.28e-11	*Maximum Relative Deviation:*	1.80e-15
Absolute Standard Deviation:	6.37e-12	*Relative Standard Deviation:*	2.88e-16

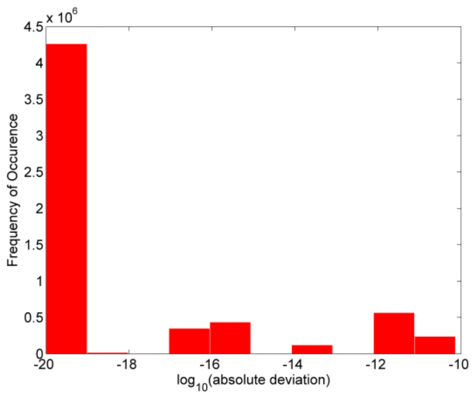


Figure 2-32 ENV_ISA-Nominal_TC2 - 1

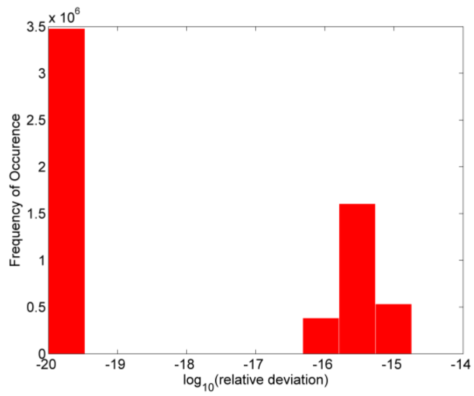


Figure 2-33 ENV_ISA-Nominal_TC2 - 2

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 2-22 ENV_ISA-Nominal_TC2

Equivalence of Implemented Model and Dissimilar Implementation?	**YES ☒**	**NO ☐**

Correct Nominal Behavior?	**YES ☒**	**NO ☐**

2.8.3 Verification of Operational Requirements

2.8.3.1 Results for Point Execution Reproducibility and Determinism Testing

Test Name:	**Point Execution Reproducibility and Determinism Test**
Test ID:	ENV_ISA-Operational_TC1
Verification Plan:	See section 2.7.3.1
Number of Test Points:	1,000,000
Execution Time:	171.03s
Rel. Deviation Threshold:	1.00e-13

Comparison of Forward Sweep to Backward Sweep:			
All Deviations below Threshold		YES ☒	NO ☐
Absolute Deviations		**Relative Deviations**	
Average Absolute Deviation:	0.0	*Average Relative Deviation:*	0.0
Maximum Absolute Deviation:	0.0	*Maximum Relative Deviation:*	0.0
Absolute Standard Deviation:	0.0	*Relative Standard Deviation:*	0.0

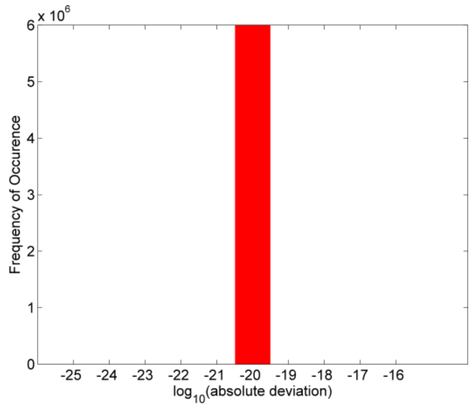


Figure 2-34 ENV_ISA-Operational_TC1 - 1

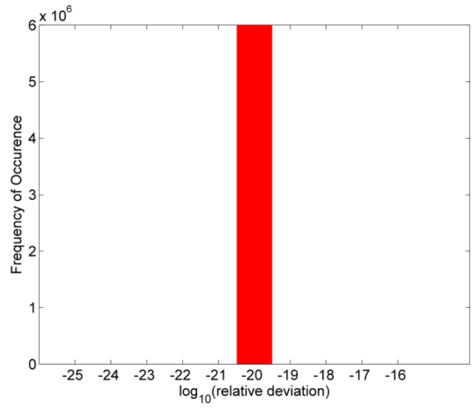


Figure 2-35 ENV_ISA-Operational_TC1 - 2

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 2-23 ENV_ISA-Operational_TC1 - 1

Comparison of Multiple Step Size Sweeps:			
All Deviations below Threshold		YES ☒	NO ☐
Absolute Deviations		**Relative Deviations**	
Average Absolute Deviation:	0.0	*Average Relative Deviation:*	0.0
Maximum Absolute Deviation:	0.0	*Maximum Relative Deviation:*	0.0
Absolute Standard Deviation:	0.0	*Relative Standard Deviation:*	0.0

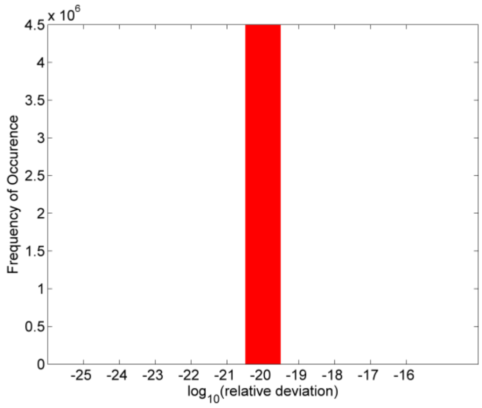
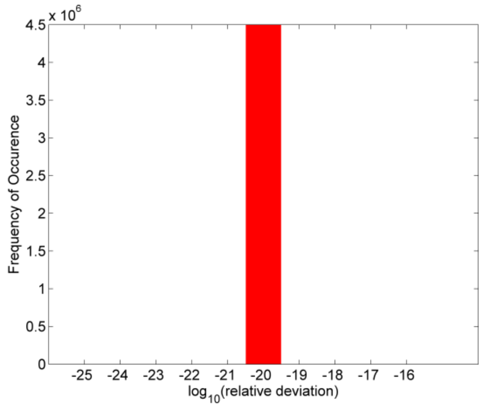

		

Figure 2-36 ENV_ISA-Operational_TC1 - 3		**Figure 2-37 ENV_ISA-Operational_TC1 - 4**	
Description of deviation exceeding the pre-specified threshold and assessment of possible causes			

Table 2-24 ENV_ISA-Operational_TC1 - 2

Run-Time Errors or Warnings?	YES ☐	NO ☒
Description of Error / Warning Messages		
Deficiencies in Operational Robustness?	YES ☐	NO ☒
Description of Detected Deficiencies		

Single Point Execution reproducible and deterministic?	**YES ☒**	**NO ☐**

2.8.3.2 Results for Code Generation and Equivalence Testing

Test Name:	**Code Generation and Equivalence Testing**
Test ID:	ENV_ISA-Operational_TC2
Verification Plan:	See section 2.7.3.2
Number of Test Points:	1,000,000
Execution Time:	85.47s
Rel. Deviation Threshold:	1.00e-13

Equivalence of S-function and Simulation Model	
All Deviations below Threshold	YES ☒ NO ☐
Absolute Deviations	**Relative Deviations**
Average Absolute Deviation: 4.83e-16	*Average Relative Deviation:* 1.62e-20
Maximum Absolute Deviation: 1.45e-10	*Maximum Relative Deviation:* 9.49e-16
Absolute Standard Deviation: 9.58e-13	*Relative Standard Deviation:* 5.66e-17

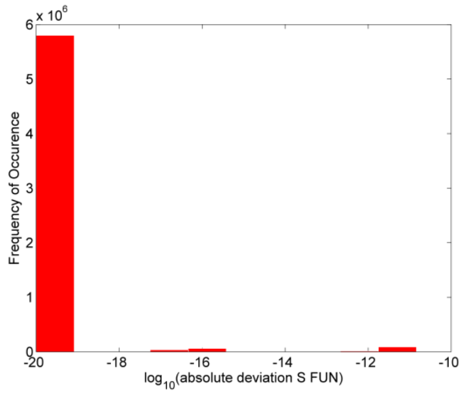
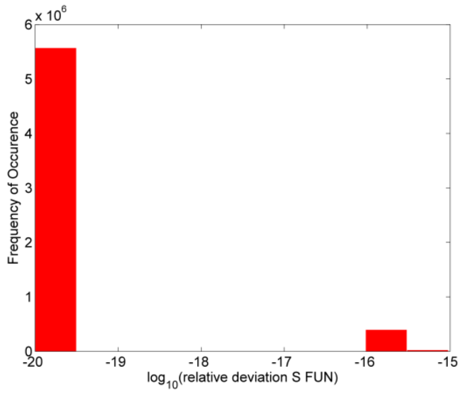

	

Figure 2-38 ENV_ISA-Operational_TC2 - 1	**Figure 2-39 ENV_ISA-Operational_TC2 - 2**
Description of deviation exceeding the pre-specified threshold and assessment of possible causes	

Table 2-25 ENV_ISA-Operational_TC2 - 1

Equivalence of Software-in-the-Loop (SIL) and Simulation Model			
All Deviations below Threshold		YES ☒	NO ☐
Absolute Deviations		**Relative Deviations**	
Average Absolute Deviation:	0.0	*Average Relative Deviation:*	0.0
Maximum Absolute Deviation:	0.0	*Maximum Relative Deviation:*	0.0
Absolute Standard Deviation:	0.0	*Relative Standard Deviation:*	0.0

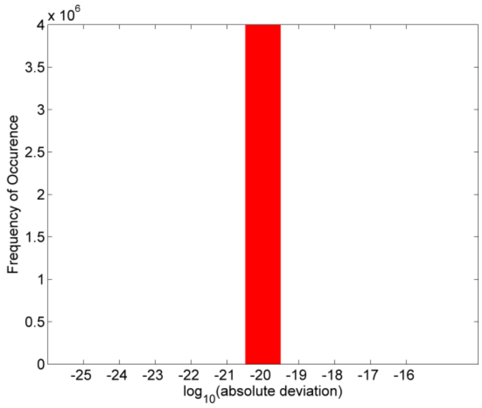


Figure 2-40 ENV_ISA-Operational_TC2 - 3

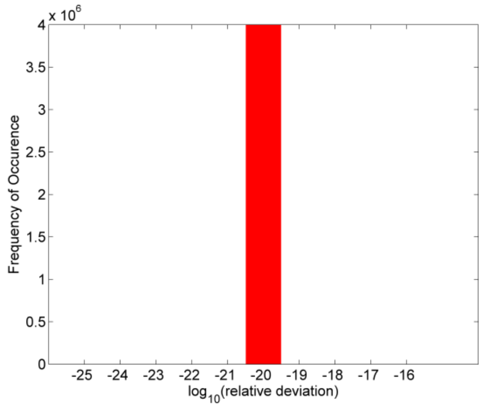


Figure 2-41 ENV_ISA-Operational_TC2 - 4

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 2-26 ENV_ISA-Operational_TC2 - 2

Compilation of S-function successful?	YES ☒	NO ☐
Compilation of standalone executable successful?	YES ☒	NO ☐
Description of Compilation Errors		
Warnings during Compilation?	YES ☐	NO ☒
Description of Compilation Warnings		

Run-Time Errors or Warnings?	YES ☐	NO ☒
Description of Error / Warning Messages		

Code Generation successful and Coded Version equivalent to Simulation Model?	**YES ☒**	**NO ☐**

3 Wind Model

3.1 Introduction

The wind is the result of the motion of air masses in the atmosphere.

Wind is caused by differences in pressure. When a difference in pressure exists, the air is accelerated from higher to lower pressure then the wind is the movement of an air mass from an area of the terrestrial surface with high pressure (anticyclonic) to an area with low pressure (cyclonic).

In the 1970s and 1980s, an alarming number of fatal accidents were attributed to the phenomenon known as wind shear (see Figure 3-1).

Figure 3-1 Wind Shear

Wind shear, sometimes referred to as wind gradient, is a difference in wind speed and direction over a relatively short distance in the atmosphere.

3.2 Description of the Functional and Operational Intent

The system is intended to compute the Wind Velocity $(\vec{\boldsymbol{v}}_W^G)_O^E$ and the Wind Direction χ_W^G, taking into account the effects due to wind shear. In particular, the system calculates the Wind Velocity at an altitude of 20 ft $(\boldsymbol{v}_W^{20ft})_W^E$, necessary for the operation of the Turbulence Model.

3.3 Requirements

Requirements that the model must satisfy, are essentially of three types: functional, operational and implementation requirements. These requirements are summarized in the following tables and are named by the acronyms that allow a more rapid identification.

3.3.1 Functional Requirements

Requirement Name **Computation the Wind Velocity $(\vec{v}_W^G)_O^E$**	Requirement ID R-FUN-ENV_WIND_01
Derived from Purpose of the system.	
Requirement Definition The Wind Velocity of the center of gravity *G* defined with respect to the Earth Centered Fixed (E) frame, components written in O frame, shall be computed.	

Table 3-1 R-FUN-ENV_WIND_01

Requirement Name **Computation of Wind Direction χ_W^G**	Requirement ID R-FUN-ENV_WIND_02
Derived from Purpose of the system.	
Requirement Definition The system shall compute the Wind Direction χ_W^G (defined relative to the direction of the North)	

Table 3-2 R-FUN-ENV_WIND_02

Requirement Name **Computation the Wind Velocity $(v_W^{20ft})_W^E$**	Requirement ID R-FUN-ENV_WIND_03
Derived from Purpose of the system.	
Requirement Definition The Wind Velocity of the center of gravity *G* at an altitude of 20 ft defined with respect to the Earth Centered Fixed (E) frame, components written in W frame, shall be computed.	

Table 3-3 R-FUN-ENV_WIND_03

3.3.2 Operational Requirements

Requirement Name	Requirement ID
Incorporation into Flight Simulator Simulation Model	R-OPS-ENV_WIND
Derived from Usage intents	
Requirement Definition The model shall be incorporated as a child system into the simulation model of an (atmospheric) flight simulation device. Therefore it is necessary that all components support code generation.	

Table 3-4 R-OPS-ENV_WIND

3.3.3 Implementation Requirements

Requirement Name	Requirement ID
Numeric Efficiency	R-NUM-ENV_WIND
Derived from Global Implementation Guidelines	
Requirement Definition The coded algorithm must not contain any of the numerical inefficient programming techniques listed below unless detailed justification substantiates indispensability.	
Programming Techniques to be Avoided:	
Unused / Dead Code Branches	
Computational Redundancies	
Matrix Inversions	
Scalar Expansion of Vector / Matrix Math	
Circle Computations	
Inefficient Lookup Table Programming	
Algebraic Loops	

Table 3-5 R-NUM-ENV_WIND

Requirement Name	Requirement ID
Input / Output Interface Compliance to Parent System and Child Systems	R-IOC-ENV_WIND
Derived from Global Implementation Guidelines	
Requirement Definition Input and Output interface must comply with parent system.	
Compliance required to:	
Global bus object definitions	
I/O signal name matching to parent system	
I/O signal unit matching to parent system	
I/O signal data type matching to parent system	
I/O signal data range compatibility matching to parent system	

Table 3-6 R-IOC-ENV_WIND

Requirement Name	Requirement ID
Implementation Compliance to FSD Style Guidelines	R-SGC-ENV_WIND
Derived from Global Implementation Guidelines	
Requirement Definition Only a subset of SIMULINK blocks is allowed to be implemented.	
Allowed Libraries and Toolboxes:	
FSD Compliant Base	
Use of other Libraries and Toolboxes is Forbidden!	

Table 3-7 R-SGC-ENV_WIND

Requirement Name **Implementation Standards Compliance**	Requirement ID R-ISC-ENV_WIND
Derived from Global Implementation Guidelines	
Requirement Definition The coded algorithm must not contain any programming technique listed below unless detailed justification substantiates indispensability.	
Forbidden Programming Techniques:	
Discrete Switches *	
Memory Blocks	
Time Delays	
Time Dependent / Non-Autonomous Elements	
In-lined Integrations	
Hysteresis and Quantized Elements	
Stochastic / Random Elements	
Normal atan Blocks	
Operations with Sign Loss	
Value Flipping and Range Limiting	
Math Function out of Range	
Division by Zero	
Finite State Transition	

Table 3-8 R-ISC-ENV_WIND

* The switches in the system do not affect the correct operation

3.4 Function Specification

3.4.1 Algorithm Abstract

The system is intended to compute the Wind Velocity $(\vec{\boldsymbol{v}}_W^G)_O^E$ and the Wind Direction χ_W^G in the aircraft's center of gravity. The system allows also to calculate the Wind Velocity $(\boldsymbol{v}_W^{20ft})_W^E$ at an altitude of 20 ft.

3.4.2 Modeling Assumptions, Scope of Validity & Limitations

- Wind Shear intensity is assigned by external inputs:
  It is assumed, that the Wind Shear intensity is externally assigned;
- Initial Wind Orientation is assigned by external inputs:
  It is assumed, that the initial Wind Orientation is externally assigned;
- The change in Wind Direction is linear with altitude:
  It is assumed, that the change in Wind Direction is linear with the altitude, up to an altitude value assigned.

3.4.3 Detailed Algorithm Description

3.4.3.1 Wind Velocity

The wind speed over the altitude is defined through the interpolation of five pairs of values assigned externally:

- five values of velocity $(v_W^{array})_W^E$
- five values of altitude h^{array}

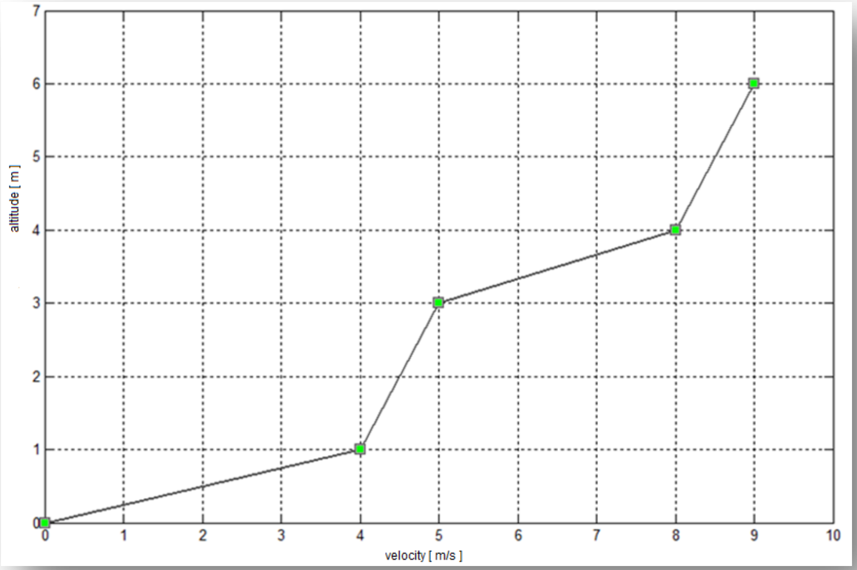


Figure 3-2 Interpolation: velocities and altitudes

The elements of h^{array} are distances from the sea level.

The Wind Shear intensity is assigned by external inputs of size [1,1] :

- $DV_{WindShear}$
- $H_{WindShear}$

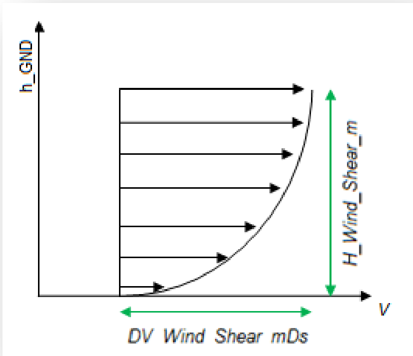


Figure 3-3 Wind Shear Intensity

where the $H_{WindShear}$ is measured starting from the ground.

If h_{GND} is the altitude respect to the ground and h_{TE} is the distance of the ground from the sea:

$$h_{GND} + h_{TE} = h^{G} \qquad \text{3-1}$$

It is possible to find from the interpolation, the value of the velocity corresponding to the $H_{WindShear}$ (point A):

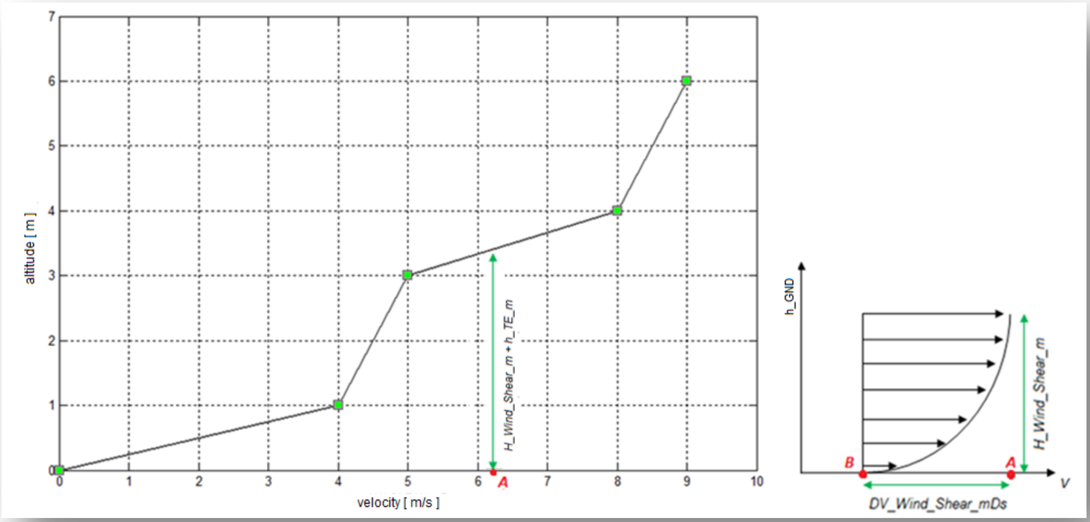


Figure 3-4 Interpolation and Wind Shear Intensity

Knowing the velocity in point A and the value of $DV_{WindShear}$, it is easy to obtain the value of the velocity in point B, called $v_{0.15}$. This value guarantees the continuity between the graphs of Figure 3-2 and Figure 3-3

The velocity profile is defined in reference [3] and here was adapted to meet the requirements set out above:

$$\left(v_{W}^{G}\right)_{W}^{E} = v_{0.15} + \frac{ln\left(\frac{h_{GND}}{z_0}\right)}{ln\left(\frac{H_{WindShear}}{z_0}\right)} \cdot DV_{WindShear} \qquad \text{3-2}$$

where z_0 corresponds to 0.15 ft.

3.4.3.2 Limitations for Wind Velocity

In order to avoid negative values of $\left(v_W^G\right)_W^E$, have been added some limitations for the parameters used:

- $H_{WindShear}$:
  This value must be positive;

- $DV_{WindShear}$:
  This value must be positive. If it is set to a negative value, the system changes it automatically in zero. In this way, from equation 3-2:

$$\left(v_W^G\right)_W^E = v_{0.15} = constant \quad \text{3-3}$$

the velocity remains constant up to $\mathrm{H_{WindShear}}$. To show this problem, an error signal $Error_{Negative_{DV}}$ has been inserted:

$$Error_{Negative_{DV}} = \begin{cases} 0, & DV_{WindShear} > 0 \\ 1, & DV_{WindShear} < 0 \end{cases} \quad \text{3-4}$$

In this case, the value of $\left(v_W^{20ft}\right)_W^E$ could be wrong. To show this problem, a second error signal $Error_{Wrong_{vel_{20ft}}}$ has been inserted:

$$Error_{Wrong_{vel_{20ft}}} = \begin{cases} 0, & H_{WindShear} < 20\,ft \\ 1, & H_{WindShear} > 20\,ft \end{cases} \quad \text{3-5}$$

In fact, if $H_{WindShear} < 20\,ft$, the value of $\left(v_W^{20ft}\right)_W^E$ is determined through interpolation and then it is correct, but if $H_{WindShear} > 20\,ft$, the $\left(v_W^{20ft}\right)_W^E$ is in the range where the velocity remains constant and it is therefore incorrect.
The maximum value of $DV_{WindShear}$ is that corresponding to $H_{WindShear}$, called $vel_{H_{WindShear}}$; in this regard, it should be noted that the $Error_{Negative_{DV}} = 1$ indicates also the case in which it was given in input a value of $DV_{WindShear} > vel_{H_{WindShear}}$ (see Figure 3-17);

- $v_{0.15}$:
  This value must be positive. It is compute by

$$v_{0.15} = vel_{H_{WindShear}} - DV_{WindShear} \tag{3-6}$$

  If $DV_{WindShear} > vel_{H_{WindShear}}$ (in order to have negative values of $v_{0.15}$) the system change automatically the value of $DV_{WindShear}$ in $vel_{H_{WindShear}}$. In this way the minimum value of $v_{0.15}$ is zero.

- h_{GND} :
  This value must be positive. Looking at equation 3-2, it is obvious that a negative value of h_{GND} would make it impossible the calculation of the velocity in the range in which the equation is used. To avoid this problem, at the term of the logarithm has been assigned a lower limit of 1.1; the minimum allowed value for h_{GND} is then:

$$h_{GND} = 1.1 \cdot z_0 \cong 0.0502\, m \tag{3-7}$$

3.4.3.3 Wind Direction

In order to take into account the change of the direction of the wind according to its intensity, a new interpolation has been defined between five pairs of values assigned externally:

- five values of wind direction χ_W^{array}
- five values of altitude h^{array}

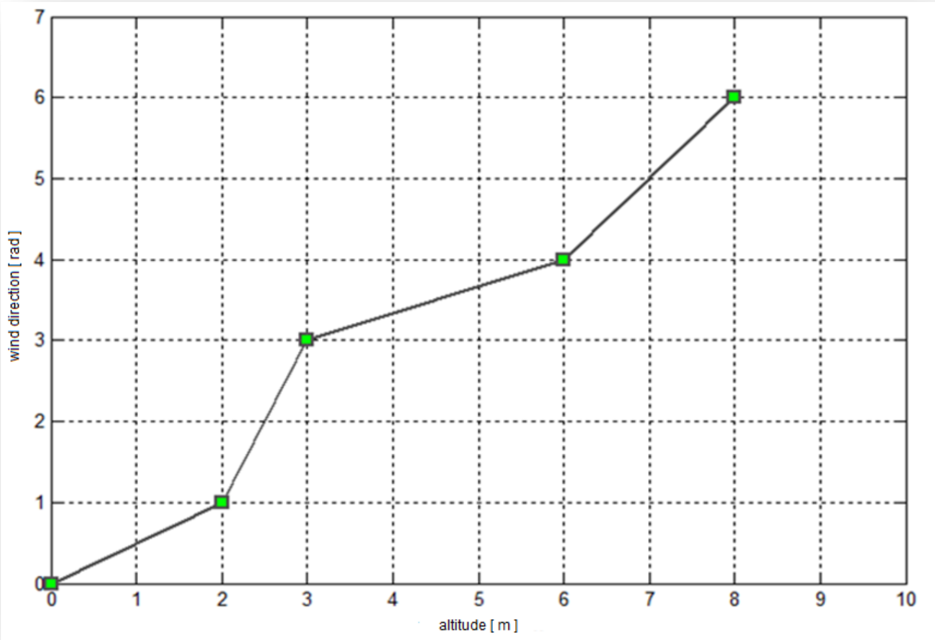


Figure 3-5 Interpolation: wind orientations and altitudes

The Initial Wind Direction is assigned by external inputs of size [1,1]:

- $H_{VectorShear}$
- $D\chi_{VectorShear}$

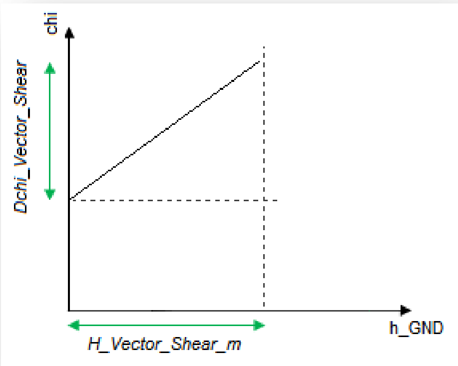


Figure 3-6 Initial Wind Direction

where $H_{VectorShear}$ is measured starting from the ground.

It is possible to find from the interpolation, the value of the wind direction corresponding to the $H_{VectorShear}$ (point A) and is easy to obtain the value of the angle in point B, called $\chi_{0.15}$

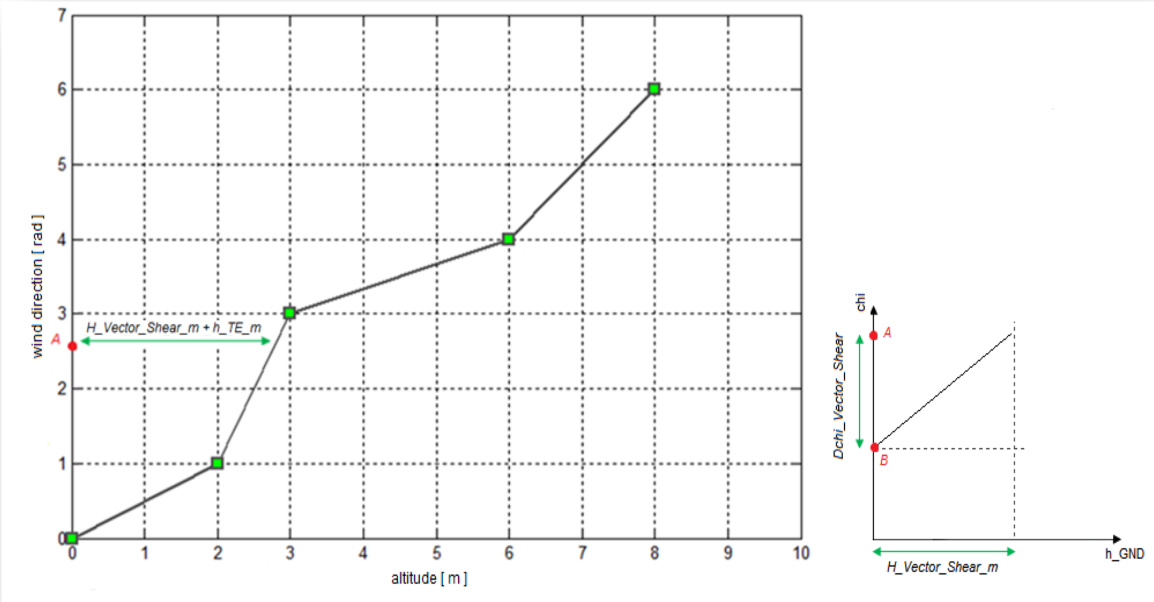


Figure 3-7 Interpolation and Initial Wind Orientation

It is then assumed that the change in wind direction is linear with altitude:

$$\chi_W^G = \chi_{0.15} + \frac{D\chi_{VectorShear}}{(H_{VectorShear})} \cdot h_{GND} \qquad \text{3-8}$$

3.4.3.4 Limitations for Wind Direction

The value of angle χ_W^G may be positive or negative. For this reason, there are not particular limitations for the parameters: the sign of $D\chi_{VectorShear}$ determines the slope of the line and the value of $\chi_{0.15}$ may be positive or negative.

The value of $H_{VectorShear}$ must be positive.

3.4.3.5 Wind Velocity at an altitude of 20 ft

The Wind Velocity at an altitude of 20 ft, necessary for the operation of the Turbulence Model, is computed by the method shown above, assuming $h_{GND}^{20ft} = 6\,m$.

3.4.3.6 Wind Velocity in NED (O) frame

Once calculated the wind velocity $\left(v_W^G\right)_W^E$ and the wind direction χ_W^G, it is easy to change the frame obtaining the wind velocity in the *O* frame:

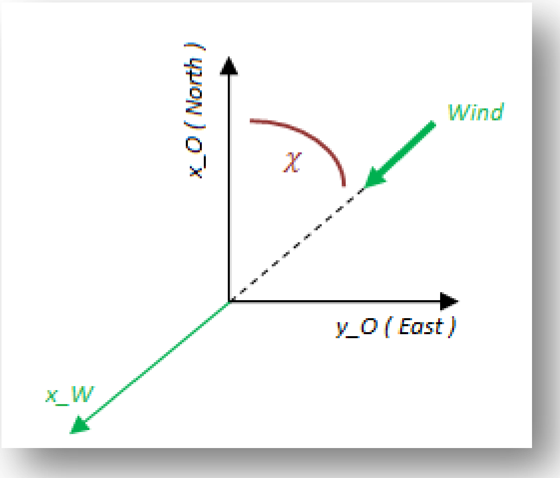


Figure 3-8 Wind Frame and NED Frame

$$\left(\vec{v}_W^G\right)_O^E = \begin{cases} \left(v_W^G\right)_W^E \cdot \left(-\cos \chi_W^G\right) \\ \left(v_W^G\right)_W^E \cdot \left(-\sin \chi_W^G\right) \\ 0 \end{cases} \qquad 3\text{-}9$$

3.4.3.7 Algorithm for Implementation

In the implemented system, all equations 3-1 to 3-9 are used.

3.5 Architecture Specification

3.5.1 Parent / Child Systems

3.5.1.1 Parent System

The system will be embedded into the system “Atmosphere”, which will be embedded into the parent system "Environment", whose purpose it is to simulate all processes regarding the environment of the aircraft (atmosphere, terrain model, earth model, etc.).

3.5.1.2 Child Systems

This system does not contain any child systems.

3.5.2 Signal Definitions

3.5.2.1 Inputs

Inputs							
Symbol	**Name**	**Size**	**Components**	**Data Type**	**Min**	**Max**	**Description**
h_{GND}	h_GND_m	[1 1]	-	double	0	2e4	Height of the aircraft's center of gravity above the ground.
h^G	h_G_WGS84_m	[1 1]	-	double	-500	2e4	Height of the aircraft's center of gravity above the WGS84 reference ellipsoid.
$DV_{WindShear}$	DV_Wind_Shear_mDs	[1 1]	-	double	0	Inf	External input for Wind Intensity: velocity variation
$H_{WindShear}$	H_Wind_Shear_m	[1 1]	-	double	0	2e4	External input for Wind Intensity: height variation
$D\chi_{VectorShear}$	Dchi_Vector_Shear_rad	[1 1]	-	double	-pi	+pi	External input for Initial Wind Orientation: direction variations
$H_{VectorShear}$	H_Vector_Shear_m	[1 1]	-	double	0	2e4	External input for Initial Wind Orientation: height variations
h^{array}	h_array_WGS84_m	[5 1]	-	double	-500	2e4	External input for interpolation: heights
$\left(v_W^{array}\right)_W^E$	vel_W_array_E_W_mDs	[5 1]	-	double	0	Inf	External input for interpolation: velocities
χ_W^{array}	chi_W_array_rad	[5 1]	-	double	-pi	-pi	External input for interpolation: : wind Direction referred to *O* frame (North)

Table 3-9 Inputs

3.5.2.2 Outputs

Outputs							
Symbol	**Name**	**Size**	**Components**	**Data Type**	**Min**	**Max**	**Description**
$\left(\vec{v}_{W}^{G}\right)_{O}^{E}$	vel_W_G_E_O_mDs	[3 1]	u_W_G_E_O_mDs [1 1] v_W_G_E_O_mDs [1 1] 0	double	-Inf -Inf 0	Inf Inf 0	Wind velocity of the center of gravity *G* defined with respect to the Earth Centered Fixed (E) frame, components written in *O* frame
χ_{W}^{G}	chi_W_G_rad	[1 1]	-	double	-pi	+pi	Wind Direction with respect to the *O* frame (North)
$\left(\boldsymbol{v}_{W}^{20ft}\right)_{W}^{E}$	vel_W_20ft_E_W_mDs	[1 1]	-	double	-Inf	+Inf	Wind velocity at an altitude of 20 ft, defined with respect to the Earth Centered Fixed (E) frame, components written in *W* frame
$Error_{Negative_{DV}}$	ERROR_Negative_DV_Wind_Shear_mDs	[1 1]	-	double	0	1	Error signal: $DV_{WindShear}$ negative
$Error_{Wrong_{vel_{20ft}}}$	ERROR_Wrong_vel_20ft_mDs	[1 1]	-	double	0	1	Error signal: wrong value of $\left(v_{W}^{20ft}\right)_{W}^{E}$

Table 3-10 Outputs

3.5.2.3 Bus Structure

To facilitate the transport of signals, were often created the buses.

In the following tables, buses created for input and output signals are then collected.

Inputs

L	Bus Name	Elements	Element Types
0	*pos_G_WGS84_Bus*	lambda_G_WGS84_rad phi_G_WGS84_rad h_G_WGS84_m	double double double
0	*External_Inputs_Wind_Shear_Bus*	DV_Wind_Shear_mDs H_Wind_Shear_m	double double
0	*External_Inputs_Vector_Shear_Bus*	Dchi_Vector_Shear_rad H_Vector_Shear_m	double double
0	*Wind_External_Inputs_Bus*	h_array_WGS84_m vel_W_array_E_W_mDs chi_W_array_rad	double double double

Table 3-11 Inputs Bus Structure

Outputs

L	Bus Name	Elements	Element Types
0	*vel_W_G_E_O_Bus*	u_W_G_E_O_mDs v_W_G_E_O_mDs w_W_G_E_O_mDs	double double double

Table 3-12 Outputs Bus Structure

3.6 Structural Layout

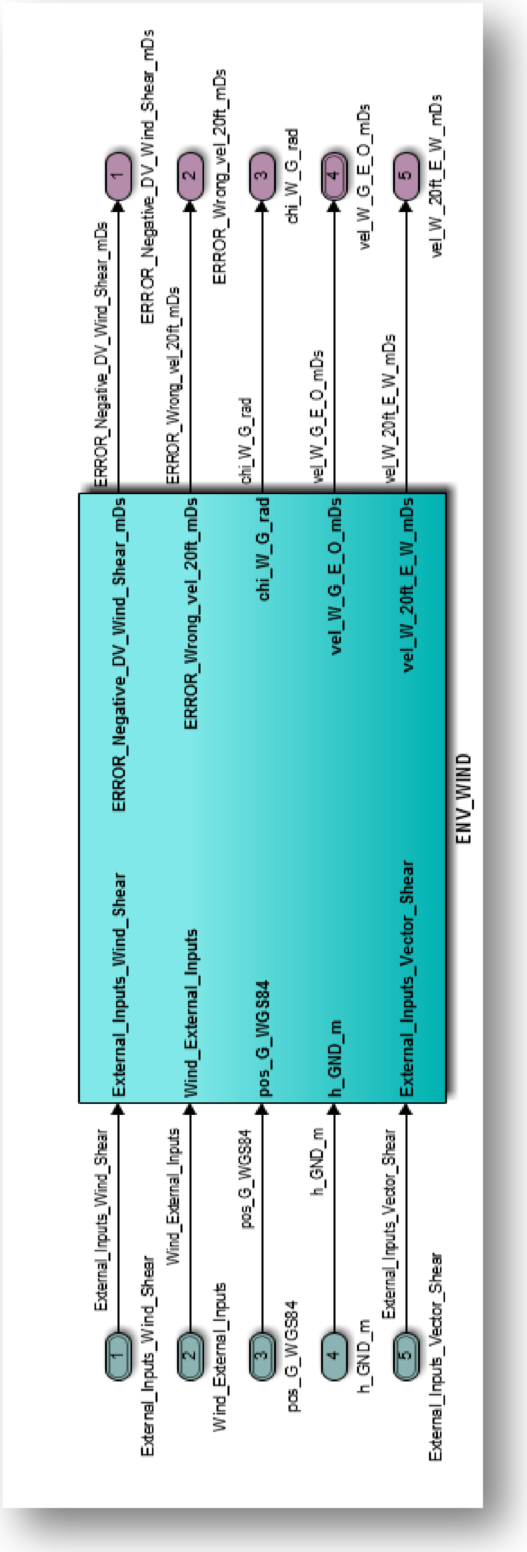


Figure 3-9 L0: ENV_WIND

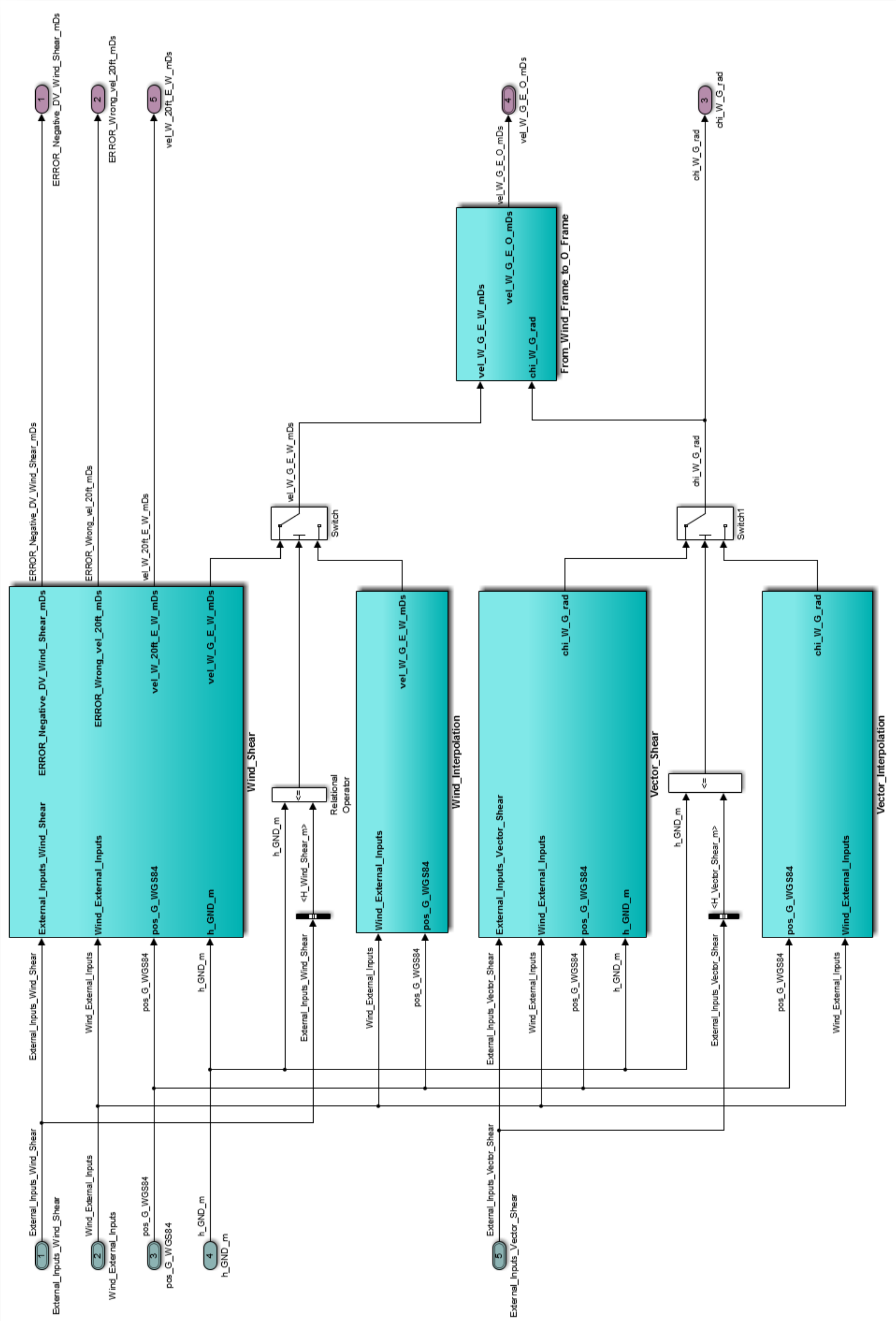


Figure 3-10 L1: WIND_Implementation

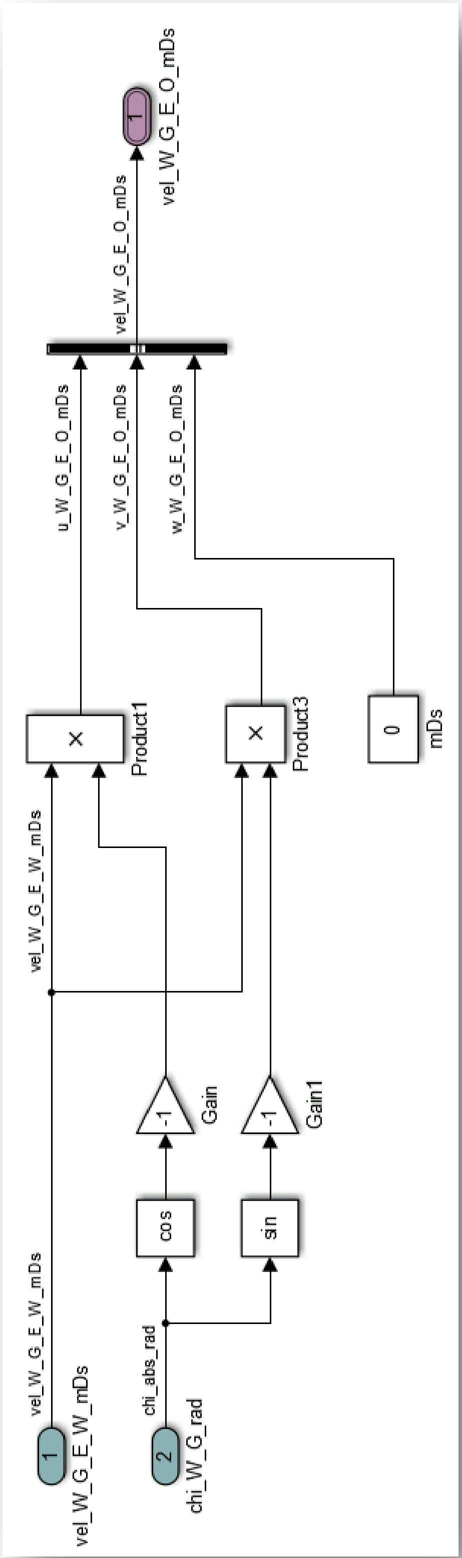


Figure 3-11 L2: From_Wind_Fame_to_O_Frame

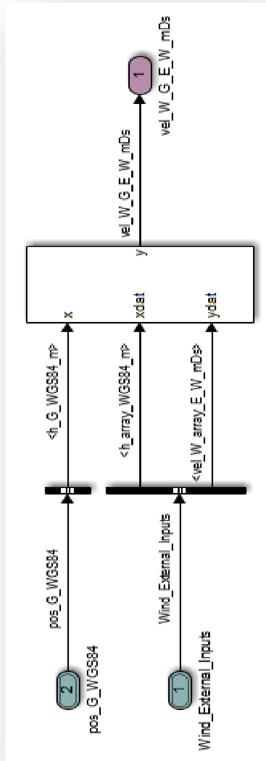


Figure 3-12 L2: Wind_Interpolation

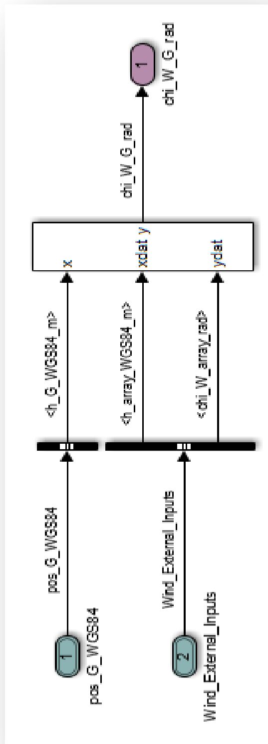


Figure 3-13 L2: Vector _Interpolation

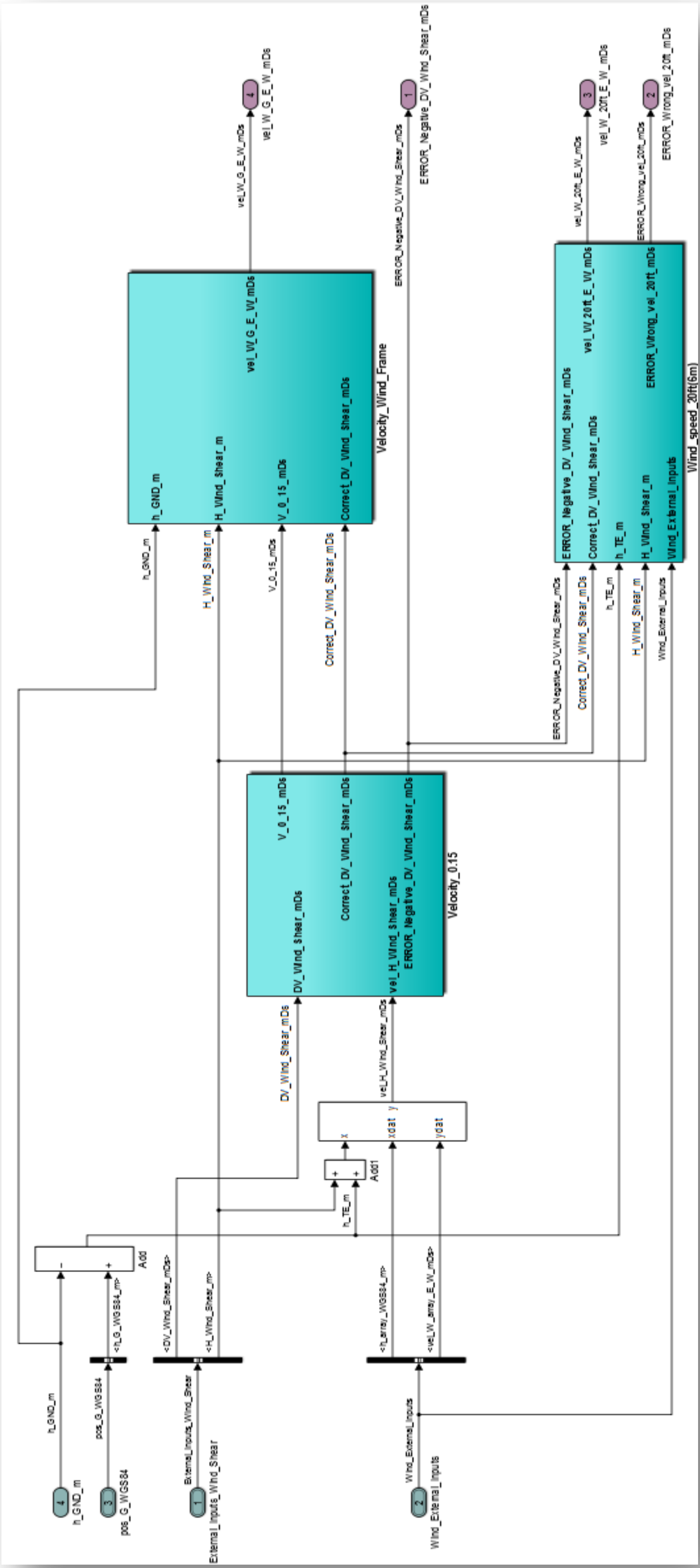


Figure 3-14 L2: Wind_Shear

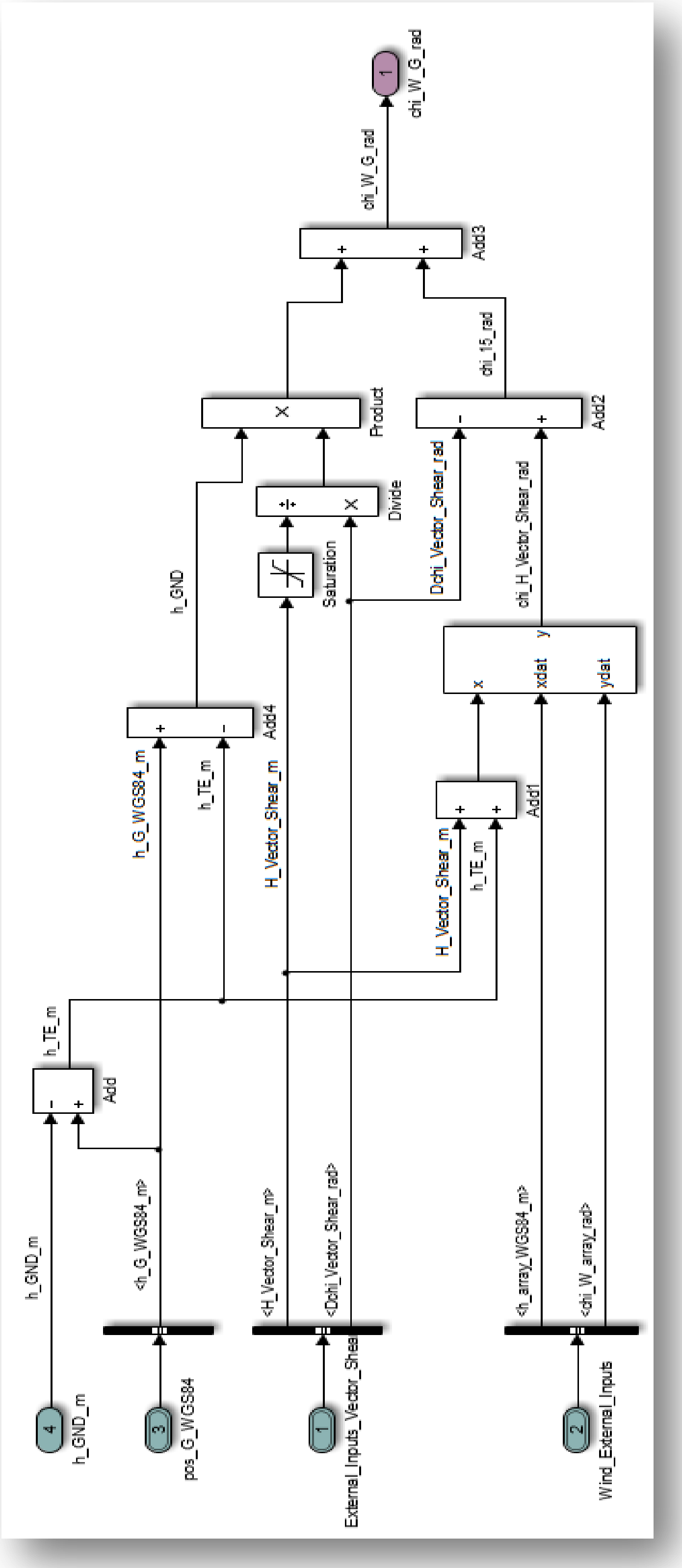


Figure 3-15 L2: Vector_Shear

Figure 3-16 L3: Velocity_Wind_frame

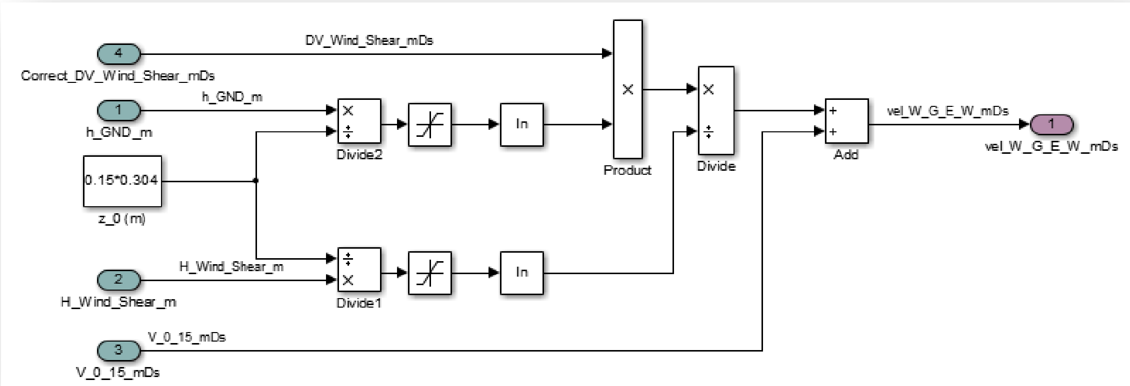


Figure 3-17 L3: Velocity_0.15

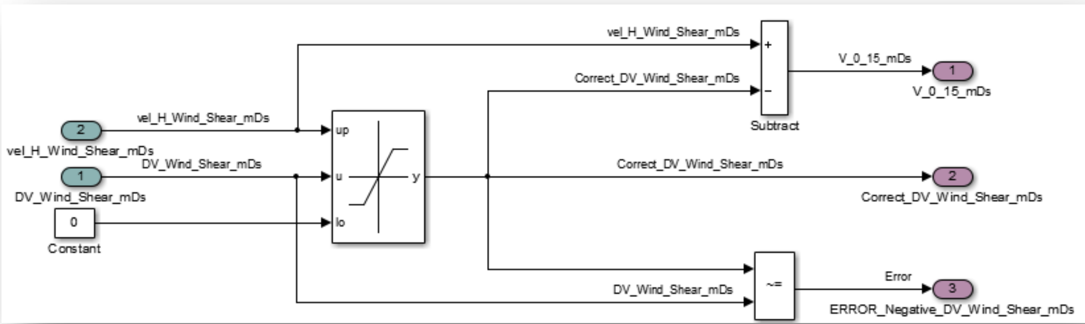
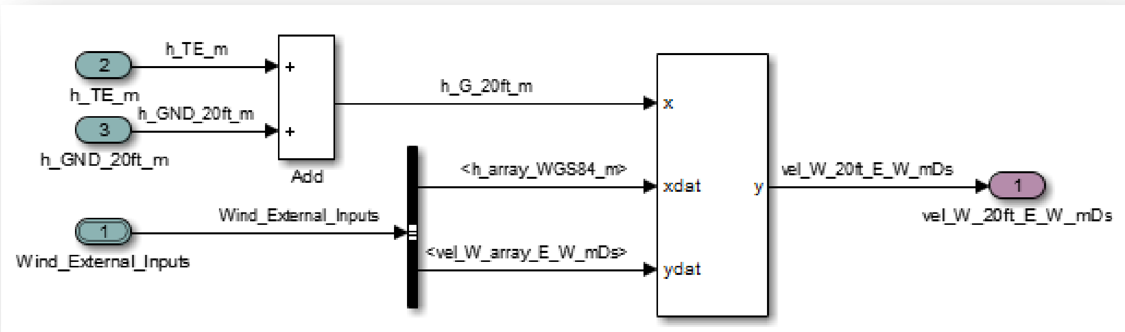


Figure 3-18 L4: Wind_Implementation_20ft



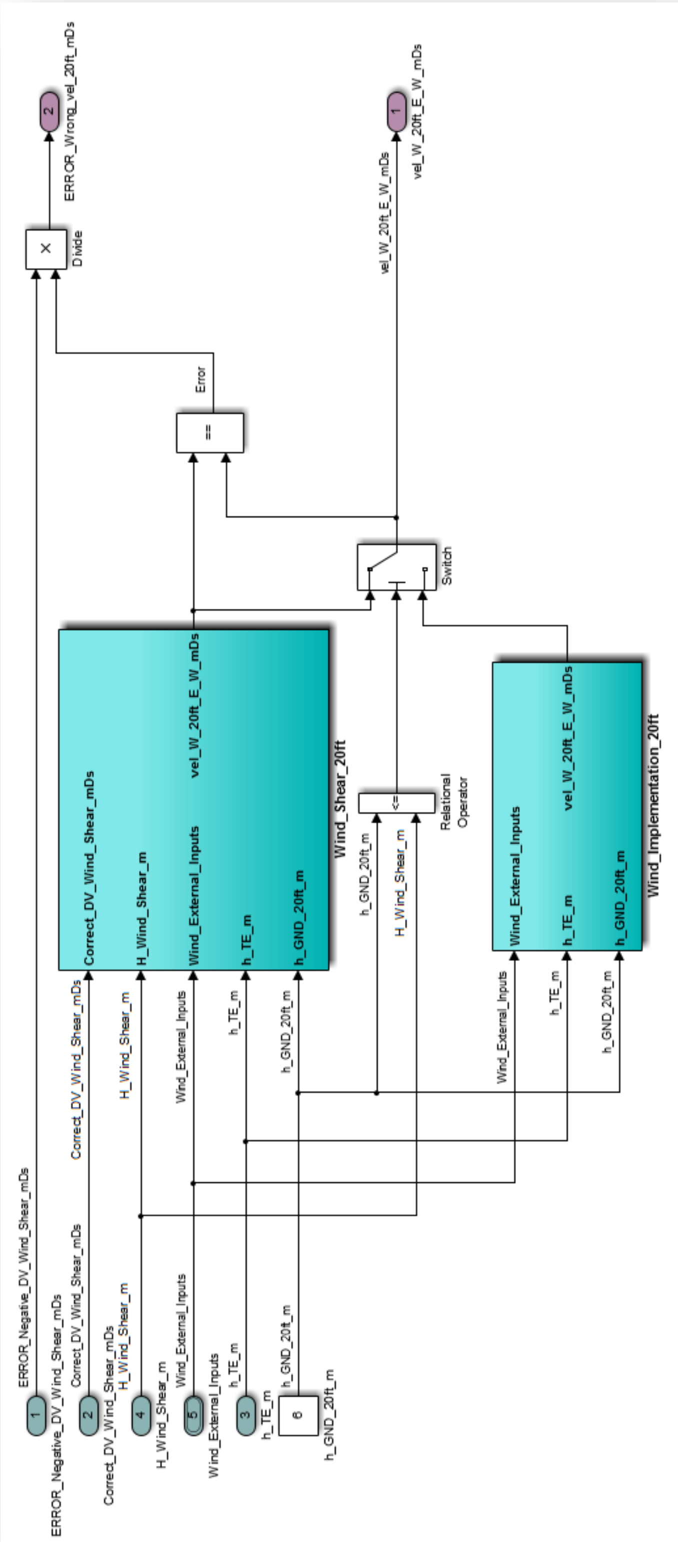


Figure 3-19 L3: Wind_speed_20ft(6m)

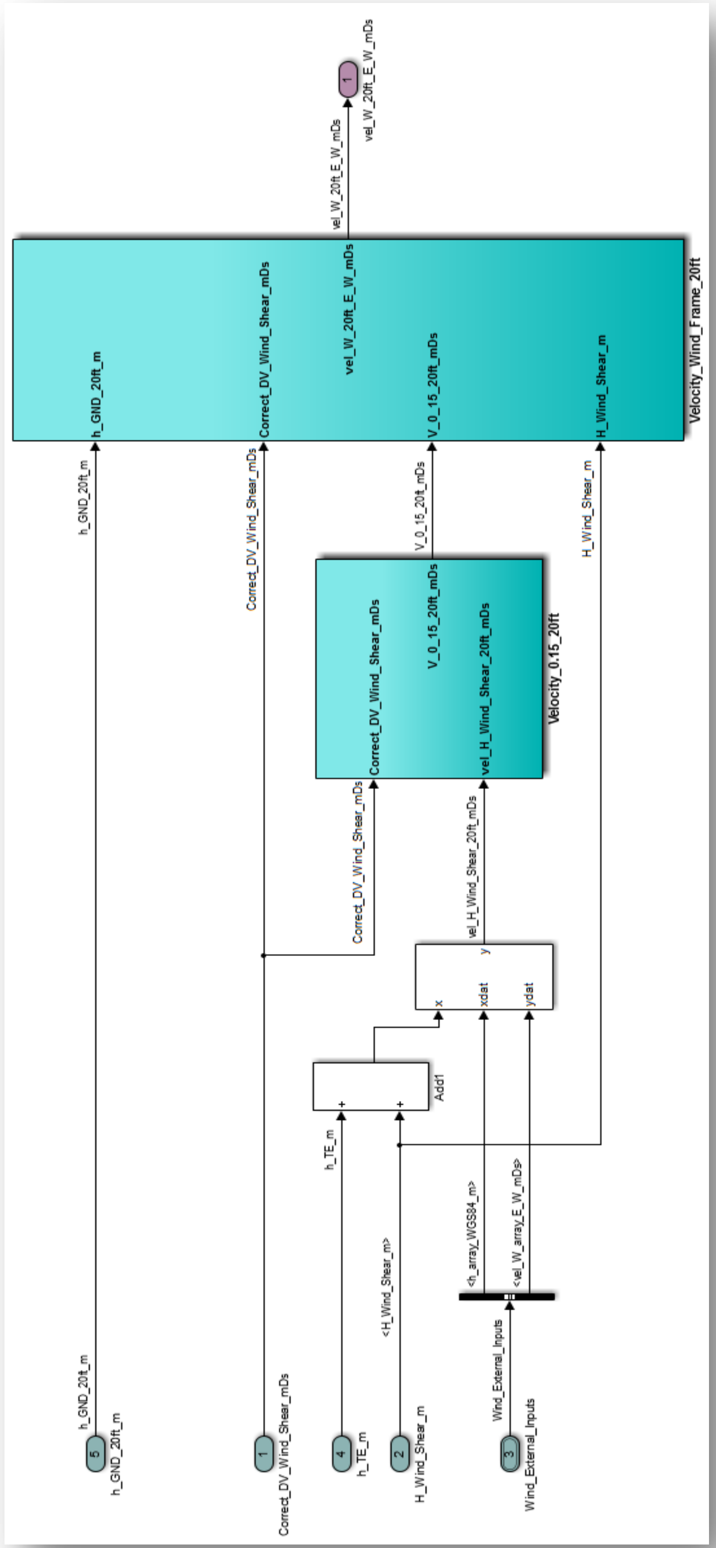


Figure 3-20 L4: Wind_Shear_ 20ft

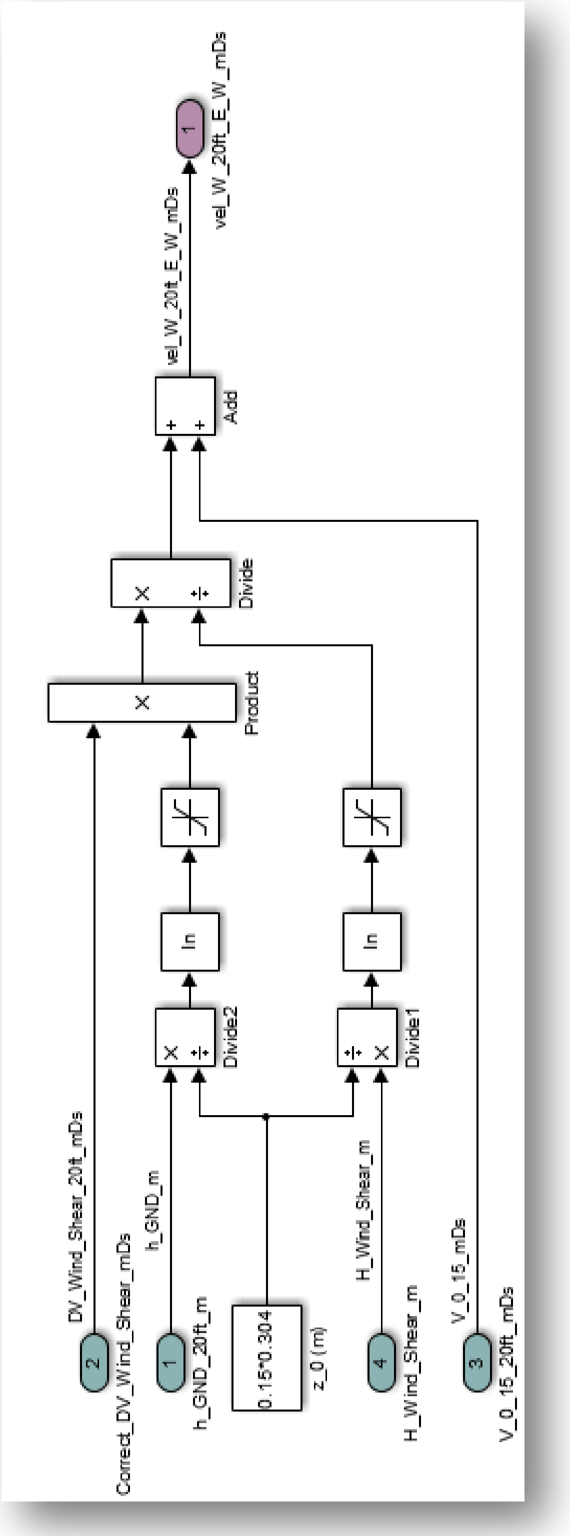


Figure 3-21 L5: Velocity_Wind_Frame_20ft

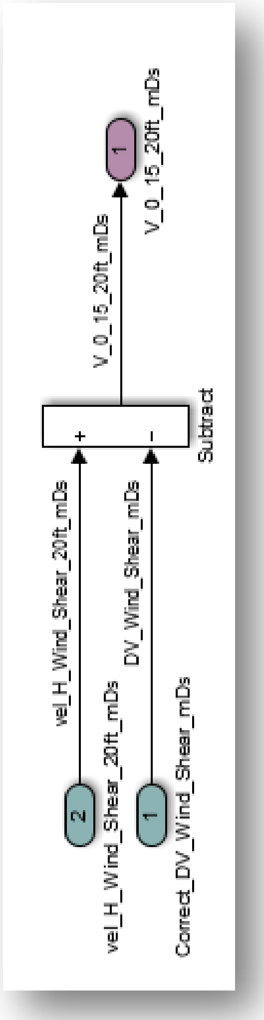


Figure 3-22 L5: Velocity_0.15_20ft

3.7 Verification Plan

3.7.1 Methods Used for Verification

3.7.1.1 Methods for Testing Functional Requirements

Requirement Name and ID	Description of Verification Method
R-FUN-ENV_WIND_01 Computation of Wind Velocity of the center of gravity *G* defined with respect to the Earth Centered Fixed (E) frame, components written in *O* frame: $\left(\vec{\boldsymbol{v}}_W^G\right)_O^E$	Correct computation demonstrated in ENV_WIND-Nominal_TC1, comparison to dissimilar implementation in ENV_WIND-Nominal_TC2 (see section 3.7.2.1).
R-FUN-ENV_WIND_02 Computation of Wind Direction χ_W^G respect to *O* frame	Correct computation in ENV_WIND-Nominal_TC1, comparison to dissimilar implementation in ENV_WIND-Nominal_TC2 (see section 3.7.2.1).
R-FUN-ENV_WIND_03 Computation of Wind velocity of the center of gravity *G* at an altitude of 20 ft defined with respect to the Earth Centered Fixed (E) frame, components written in *W* frame: $\left(\boldsymbol{v}_W^{20ft}\right)_W^E$	Correct computation in ENV_WIND-Nominal_TC1, comparison to dissimilar implementation in ENV_WIND-Nominal_TC2 (see section 3.7.2.1).

Table 3-13 Methods for Testing Functional Requirements

3.7.1.2 Methods for Testing Implementation Requirements

Requirement Name and ID	Description of Verification Method
R-NUM-ENV_WIND Numeric Efficiency	Manual review of the implemented model. Records according to section 3.8.1.1
R-IOC-ENV_WIND Input / Output Interface Compliance to Parent System	Compliance to parent system will be verified at integration with parent system.
R-SGC-ENV_WIND Implementation Compliance to FSD Style Guides	Manual review of the implemented model. Records according to section 3.8.1.2
R-ISC-ENV_WIND Implementation Standards Compliance	Manual review of the implemented model. Records according to section 3.8.1.3

Table 3-14 Methods for Testing Implementation Requirements

3.7.1.3 Methods for Testing Operational Requirements

Derived Standard Requirement Matrix:		
SIMULINK Modes	Simulink Offline Simulation	Demonstrated during test ENV_WIND-Nominal_TC1, ENV_WIND-Nominal_TC2 and ENV_WIND-Operational_TC1 (see sections 3.7.2.1 and 3.7.3.1
	Simulink Pseudo Real Time Simulation	Will be demonstrated on a higher level of the simulation model.
External Control for SIMULINK Execution	Bypassing of Non-Autonomous Elements	Not applicable as there are no non-autonomous elements present in the model.
	Single Point Execution	Demonstrated during test ENV_WIND-Nominal_TC1, ENV_WIND-Nominal_TC2 and ENV_WIND-Operational_TC1 (see sections 3.7.2.1 and 3.7.3.1).
	Online Integration Freeze and Reset	Not applicable as there are no integrators present in the model.
	Workspace Initialization	Not applicable as there are no variables present to be initialized in the workspace.
	Runtime Parameter Tuning	Not applicable as there are no tunable parameters present in the model
RTW Code Generation	S-function	Generation of S-function demonstrated during test ENV_WIND-Operational_TC2 (see section 3.7.3.2)
Code Modes	Stand-alone Batch Simulation	Will be demonstrated on a higher level of the simulation model.
	Stand-alone Real Time Simulation	Will be demonstrated on a higher level of the simulation model.

Table 3-15 Methods for Testing Operational Requirements

3.7.2 Verification Plan for Functional Requirements

3.7.2.1 Nominal Testing Procedure

Test Name:	**Correct Calculation of Wind Velocity $(\vec{v}_W^G)_O^E$ and Wind Direction χ_W^G. Correct calculation of Wind Velocity at an altitude of 20 ft $\left(v_W^{20ft}\right)_W^E$**	
Test ID:	ENV_WIND-Nominal_TC1	
Related Requirements:		R-FUN-ENV_WIND_01: Computation of $(\vec{v}_W^G)_O^E$ R-FUN-ENV_WIND_02: Computation of χ_W^G R-FUN-ENV_WIND_03: Computation of $\left(v_W^{20ft}\right)_W^E$
Verification Data:		See section 3.8.2.1

The implemented simulation model is excited with different input combinations. Afterwards, the course of the velocities and the wind directions are plotted over the geometrical altitude.

The following figure shows the scheme used for the test:

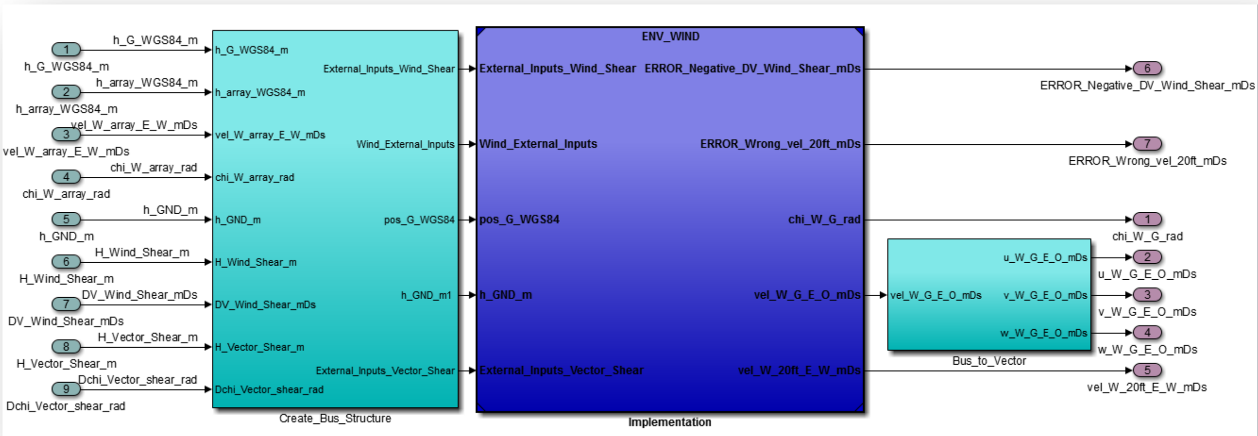


Figure 3-23 ENV_WIND-Nominal_TC1

It was necessary to create, first of all, the buses to generate inputs and then select the individual signals from the output bus. The test is then put into operation using a Matlab script.

To perform the test, the input signals have been made change in the following ranges:

- h^G: from 200 to $20000\ m$
- h_{GND} : $h^G - 100$ m
- $DV_{WindShear}$: $20\ m/s$
- $D\chi_{VectorShear}$: 0.25 rad
- $H_{WindShear}$: $3000\ m$
- $H_{VectorShear}$: $2000\ m$
- h^{array}:$[200,4000,10000,14000,20000]m$
- χ_W^{array}: $[\ 0.5, 1.5, 1.7, 2.0, 2.5\ ]$ rad
- $(v_W^{array})_W^E$: $[\ 60, 80, 85, 95, 100\ ]$ m/s

Test Name: **Equivalence of Implementation and Dissimilar Implementation**	
Test ID: ENV_WIND-Nominal_TC2	
Related Requirements:	R-FUN-ENV_WIND_01: Computation of Wind Velocity (*O* frame) R-FUN-ENV_WIND_02: Computation of Wind Direction R-FUN-ENV_WIND_03: Computation of Wind Velocity at 20 ft
Verification Data:	See section 3.8.2.1

The implemented model and a dissimilar implementation (Embedded Matlab Function) are both excited with the same input signals. Afterwards, the output is checked for deviations. As long as the relative deviations stay below a certain threshold both implementations are considered equivalent and the test is passed.

The following figure is intended to show only the scheme used for the test:

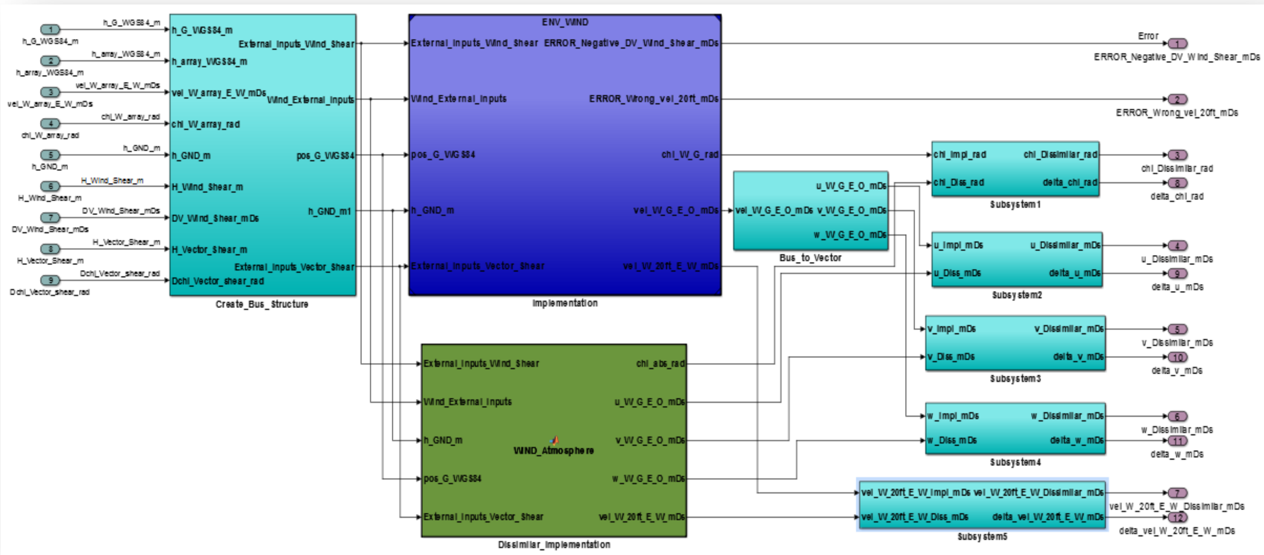


Figure 3-24 ENV_WIND-Nominal_TC2

It was necessary to create, first of all, the buses to generate inputs and then select the individual signals from the output bus. The test is then put into operation using a Matlab script.

To perform the test, the input signals have been made change in the following ranges:

- h^G: from 200 to $20000\ m$
- h_{GND} : $h^G - 100$ m
- $DV_{WindShear}$: $20\ m/s$
- $D\chi_{VectorShear}$: 0.25 rad
- $H_{WindShear}$: $3000\ m$
- $H_{VectorShear}$: $2000\ m$
- h^{array}:$[200,4000,10000,14000,20000]m$
- χ_W^{array}: $[\ 0.5, 1.5, 1.7, 2.0, 2.5\ ]$ rad
- $(v_W^{array})_W^E$: $[\ 60, 80, 85, 95, 100\ ]$ m/s

3.7.3 Verification Plan for Operational Requirements

3.7.3.1 Point Execution Reproducibility and Determinism Testing

Test Name:	**Point Execution Reproducibility and Determinism Test**
Test ID:	ENV_WIND-Operational_TC1
Related Requirements:	R-OPS-ENV_WIND: Operation Standard Requirements Matrix
Verification Data:	See section 3.8.3.1

The implemented simulation model is excited with different input combinations. Afterwards, the calculation is repeated with the inputs in reverse order and with varying step sizes.

The purpose of this test is to check for hidden non-autonomous elements in the model.

As long as the relative deviations between the forward and backward runs and the multistep runs stay below a certain threshold the implementations are considered as being equivalent.

The following figure shows the scheme used for the test:

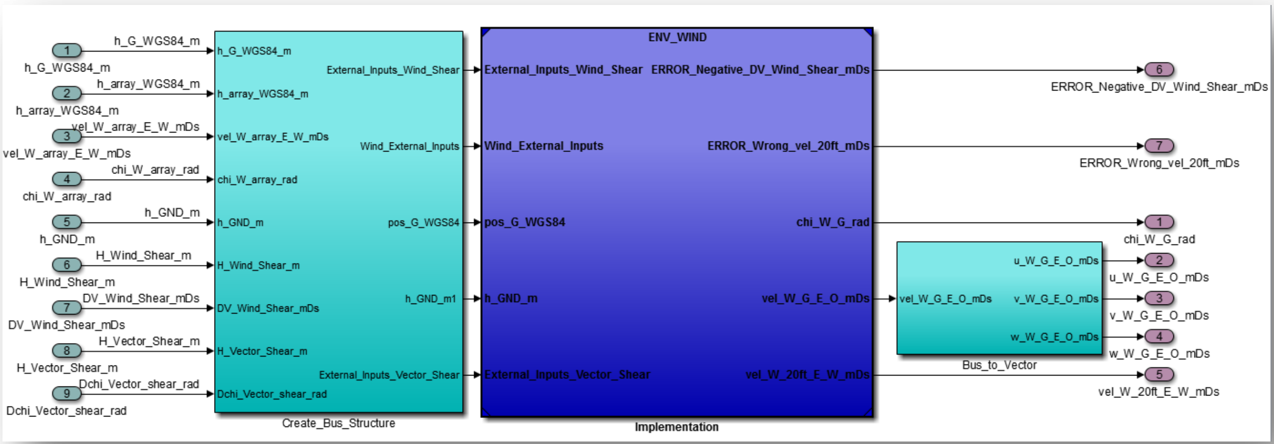


Figure 3-25 ENV_WIND-Operational_TC1

To perform the test, the input signals have been made change in the following ranges:

- h^G: from 2 to $20000\ m$
- h_{GND} : $h^G - 1\ m$

Other parameters are:

- $DV_{WindShear}$: $20\ m/s$
- $D\chi_{VectorShear}$: 0.25 rad
- $H_{WindShear}$: $3000\ m$
- $H_{VectorShear}$: $2000\ m$
- h^{array}:$[0,4000,10000,14000,20000]m$
- χ_W^{array}: [0.5, 1.5, 1.7, 2.0, 2.5] rad
- $(v_W^{array})_W^E$: [60, 80, 85, 95, 100] m/s

3.7.3.2 Code Generation and Equivalence Testing

Test Name:	**Code Generation and Equivalence Testing**
Test ID:	ENV_WIND-Operational_TC2
Related Requirements:	R-OPS-ENV_WIND: Operation Standard Requirements Matrix
Verification Data:	See section 3.8.3.2

The implemented simulation model running in normal mode as well as a compiled version (SIL) and a S-function are excited with random inputs signals.

Afterwards the outputs are checked for deviations. As long as the deviations stay below a certain threshold the test is passed.

The following figure is intended to show only the scheme used for the test:

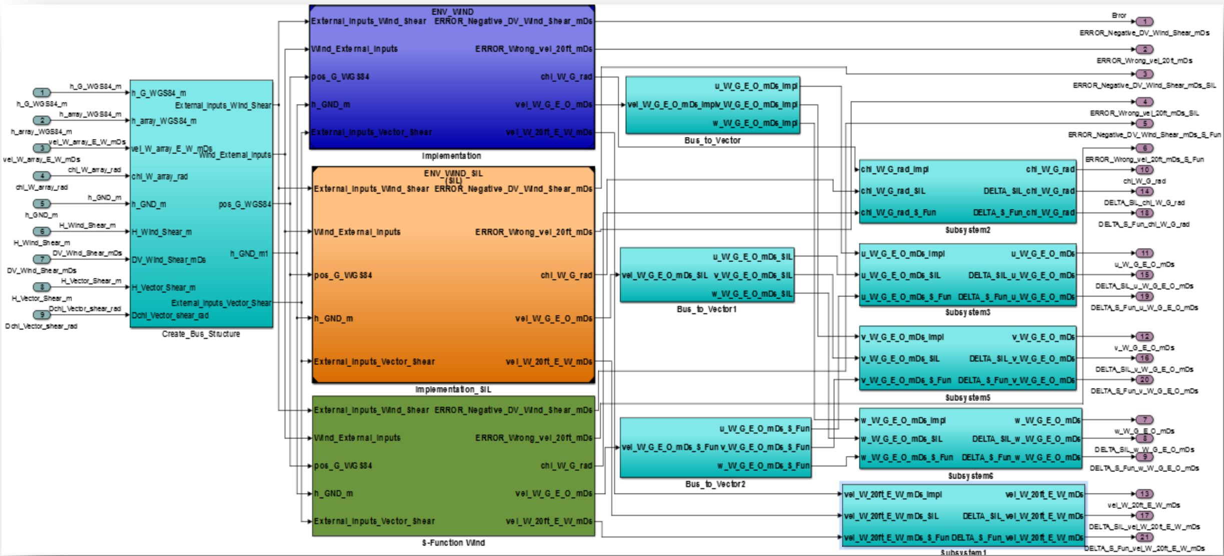


Figure 3-26 ENV_WIND-Operational_TC2

The inputs to the system are varied randomly between the following bounds:

– h^G: from 2 to 20000 m

– h_{GND} : $h^G - 1\ m$

Other parameters are:

- $DV_{WindShear}$: $20\ m/s$
- $D\chi_{VectorShear}$: 0.25 rad
- $H_{WindShear}$: $3000\ m$
- $H_{VectorShear}$: $2000\ m$
- h^{array}:$[0,4000,10000,14000,20000]m$
- χ_W^{array}: [0.5, 1.5, 1.7, 2.0, 2.5] rad
- $(v_W^{array})_W^E$: [60, 80, 85, 95, 100] m/s

3.8 Verification Data

3.8.1 Verification of Implementation Requirements

3.8.1.1 Numeric Efficiency

Requirement Name **Numeric Efficiency**		Requirement ID R-NUM-ENV_WIND
Requirement is violated if the model contains one or more of the following items:		
Unused / Dead Code Branches Description	YES ☐	NO ☒
Computational Redundancies Description	YES ☐	NO ☒
Matrix Inversions Description	YES ☐	NO ☒
Scalar Expansions of Vector/Matrix Math Description	YES ☐	NO ☒
Circle Computations Description	YES ☐	NO ☒
Inefficient Lookup Table Programming Description	YES ☐	NO ☒
Algebraic Loops Description	YES ☐	NO ☒
Numeric Efficiency met?	**YES ☒**	**NO ☐**

Table 3-16 R-NUM-ENV_WIND

3.8.1.2 Implementation Compliance to FSD Style Guidelines

Requirement Name **Implementation Compliance to FSD Style Guidelines**	Requirement ID R-SGC-ENV_WIND	
Requirement is violated if the model contains other blocks than the specified ones.		
Non-Specified Blocks within the Model? Description	YES ☐	NO ☒
Style Guide Compliance met?	**YES ☒**	**NO ☐**

Table 3-17 R-SGC-ENV_WIND

3.8.1.3 Implementation Standards Compliance

Requirement Name **Implementation Standards Compliance**	Requirement ID R-ISC-ENV_WIND	
The coded algorithm must not contain any programming technique listed below unless detailed justification substantiates indispensability.		
*Discrete Switches **	YES ☒	NO ☐
Memory Blocks	YES ☐	NO ☒
Time Delays	YES ☐	NO ☒
Time Dependent / Non-Autonomous Elements	YES ☐	NO ☒
In-lined Integrations	YES ☐	NO ☒
Hysteresis and Quantized Elements	YES ☐	NO ☒
Stochastic / Random Elements	YES ☐	NO ☒
Normal atan Blocks	YES ☐	NO ☒
Operations with Sign Loss	YES ☐	NO ☒
Value Flipping and Range Limiting	YES ☐	NO ☒
Math Function out of Range	YES ☐	NO ☒
Division by Zero	YES ☐	NO ☒
Finite State Transition	YES ☐	NO ☒
Implementation Standards Compliance met?	**YES ☒**	**NO ☐**

Table 3-18 R-ISC-ENV_WIND

* The switches in the system do not affect the correct operation

3.8.2 Verification of Functional Requirements

3.8.2.1 Results for Nominal Testing

Test Name:	**Correct Calculation of Wind Velocity $(\vec{v}_W^G)_O^E$ and Wind Direction χ_W^G. Correct calculation of Wind Velocity at an altitude of 20 ft $\left(v_W^{20ft}\right)_W^E$**
Test ID:	ENV_WIND-Nominal_TC1
Verification Plan:	See section 3.7.2.1

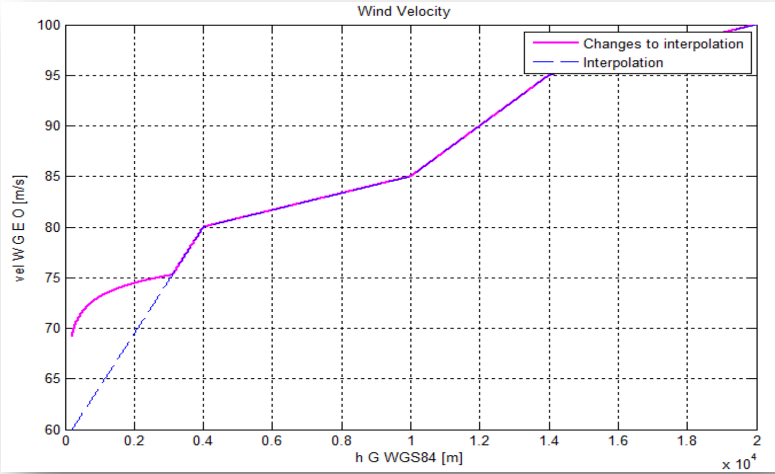


Figure 3-27 Correct Calculation of Wind Velocity

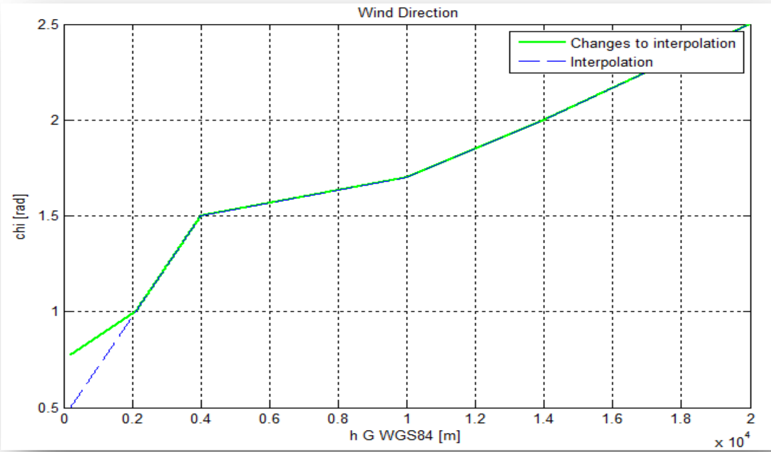


Figure 3-28 Correct Calculation of Wind Direction

The following figures are included only for the purpose of showing some exceptional cases. The Figure 3-29 shows the behavior if $DV_{WindShear} < 0$.

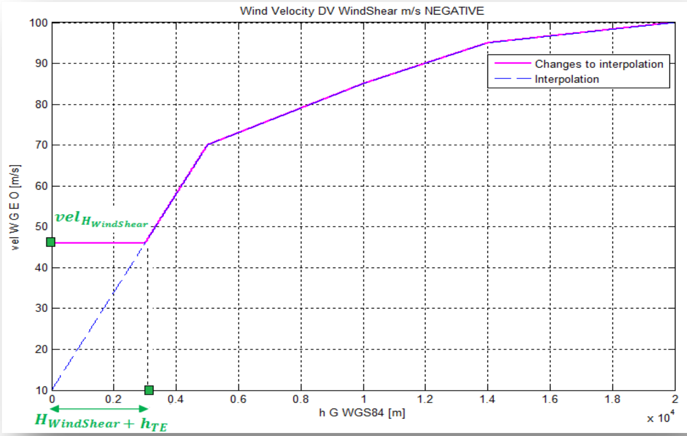


Figure 3-29 Limitations: correct behavior of the Wind Velocity if $DV_{WindShear}$ is negative

The problem is indicated by $Error_{Negative_{DV}} = 1$. Furthermore, since in this case is also $h_{GND}^{20ft} < H_{WindShear}$ (where h_{GND}^{20ft} is the $h_{GND} = 20\ ft$) the value of $\left(v_W^{20ft}\right)_W^E$ computed in this way is wrong: this value, in fact, is located in the range where the velocity remains constant. The problem is indicated by $Error_{Wrong_{vel_{20ft}}} = 1$.

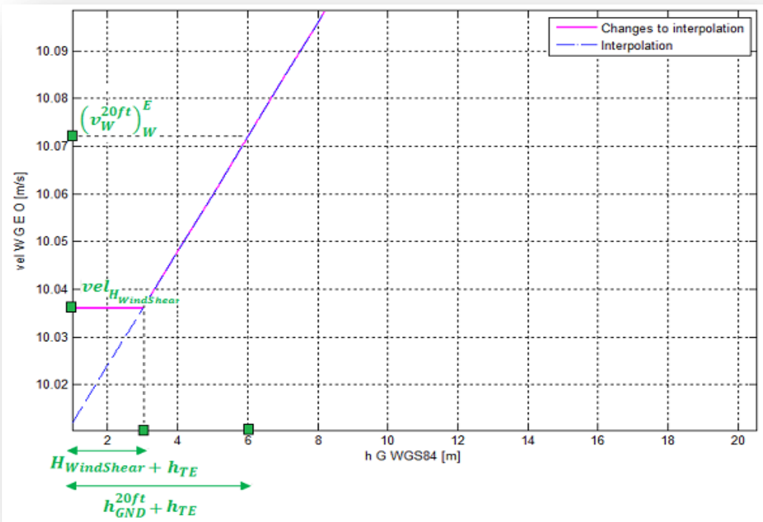


Figure 3-30 Limitations: correct value of $\left(v_W^{20ft}\right)_W^E$ with $h_{TE} = 0\ m$

The above figure shows the case in which $h_{GND}^{20ft} > H_{WindShear}$: the value of the $\left(v_W^{20ft}\right)_W^E$ is calculated correctly and $Error_{Wrong_{vel_{20ft}}} = 0$.

In the following figures is shown the correct operation of the system for negative values of altitude : h^G from -500 to $20000\,m$ in step of $1\,m$ and $h_{GND} > 0$

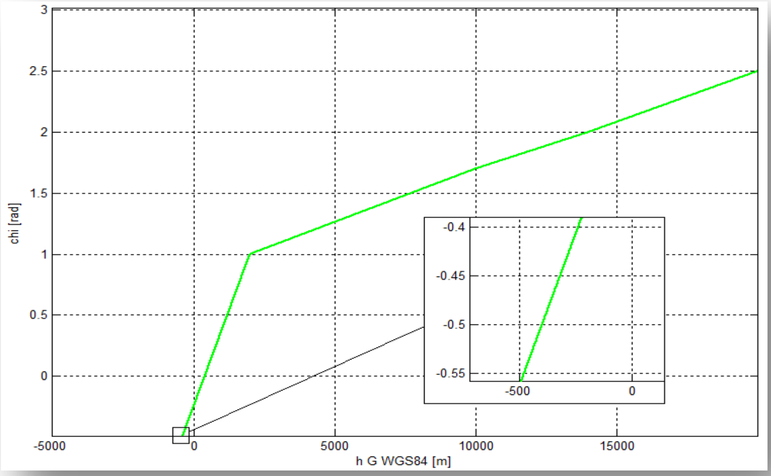


Figure 3-31 Correct behavior of the system with negative values of altitude and negative initial values of the Wind Direction

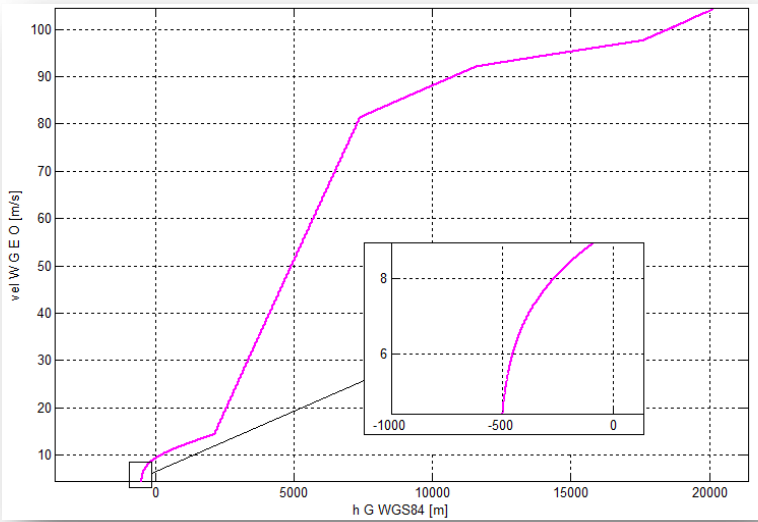


Figure 3-32 Correct behavior of the system with negative values of altitude

The following Figure 3-33 shows what happens with a negative, and then incorrect, input h_{GND}:

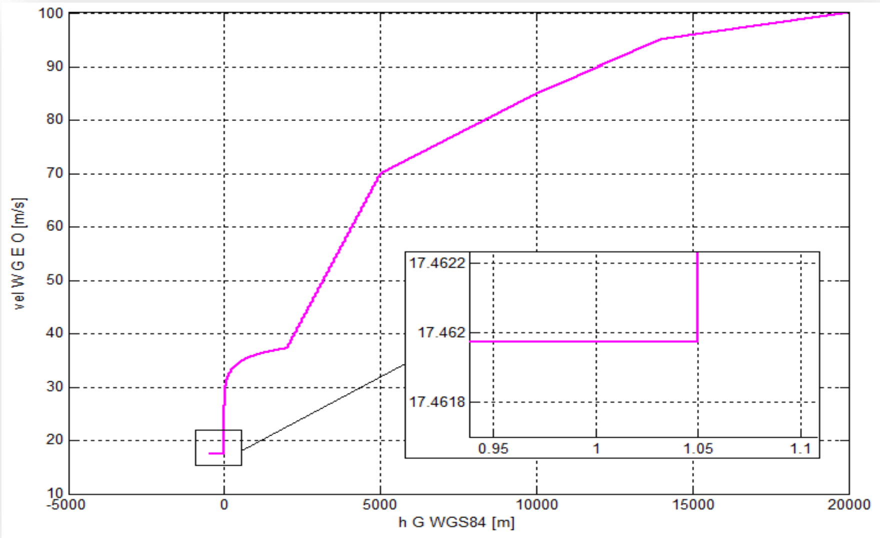


Figure 3-33 Negative value of input h_{GND}

The figure is made with the following values:

- h^G : from -500 to $20000\ m$ in steps of $0.01\ m$
- $h_{GND} = h^G - 1\ m$

In this case, the value of h_{TE} remains constant and equal to 1 m.

The system maintains the velocity constant until h_{GND} becomes greater than $0.0502\ m$ then h^G greater than $1.0502\ m$, as shown in Figure.

Course of Wind Velocity and Wind Direction over geometrical altitude realistic?	**YES ☒**	**NO ☐**

Test Name:	**Equivalence of Implementation and Dissimilar Implementation**
Test ID:	ENV_WIND-Nominal_TC2
Verification Plan:	See section 3.7.2.1
Number of Test Points:	1,000,000
Execution Time:	80.42s
Rel. Deviation Threshold:	1.00e-13

All Deviations below Threshold YES ☒ NO ☐

Absolute Deviations		**Relative Deviations**	
Average Absolute Deviation:	0.0	*Average Relative Deviation:*	0.0
Maximum Absolute Deviation:	0.0	*Maximum Relative Deviation:*	0.0
Absolute Standard Deviation:	0.0	*Relative Standard Deviation:*	0.0

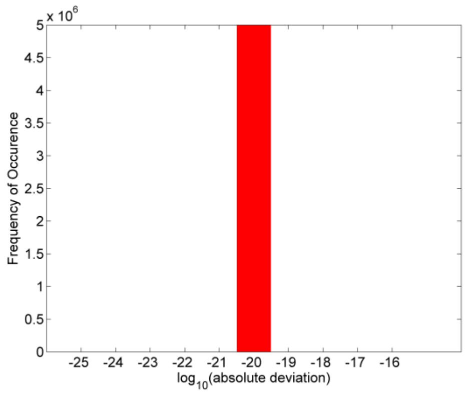


Figure 3-34 ENV_WIND-Nominal_TC2 - 1

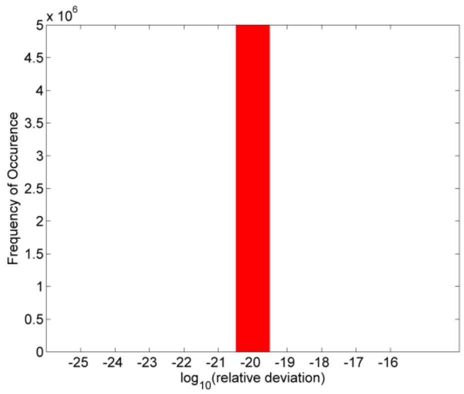


Figure 3-35 ENV_WIND-Nominal_TC2 - 2

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 3-19 ENV_WIND-Nominal_TC2

Equivalence of Implemented Model and Dissimilar Implementation?	**YES ☒**	**NO ☐**
Correct Nominal Behavior?	**YES ☒**	**NO ☐**

3.8.3 Verification of Operational Requirements

3.8.3.1 Results for Point Execution Reproducibility and Determinism Testing

Test Name:	**Point Execution Reproducibility and Determinism Test**
Test ID:	ENV_WIND-Operational_TC1
Verification Plan:	See section 3.7.3.1
Number of Test Points:	1,000,000
Execution Time:	189.71s
Rel. Deviation Threshold:	1.00e-13

Comparison of Forward Sweep to Backward Sweep:

All Deviations below Threshold YES ☒ NO ☐

Absolute Deviations		**Relative Deviations**	
Average Absolute Deviation:	0.0	*Average Relative Deviation:*	0.0
Maximum Absolute Deviation:	0.0	*Maximum Relative Deviation:*	0.0
Absolute Standard Deviation:	0.0	*Relative Standard Deviation:*	0.0

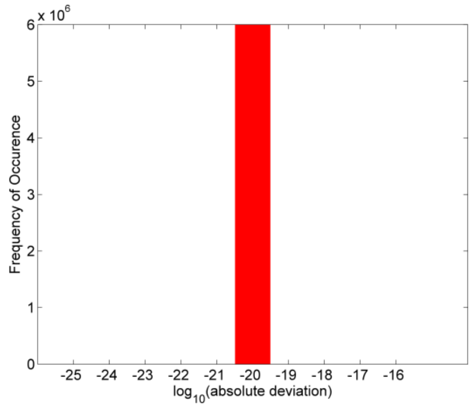


Figure 3-36 ENV_WIND-Operational_TC1 - 1

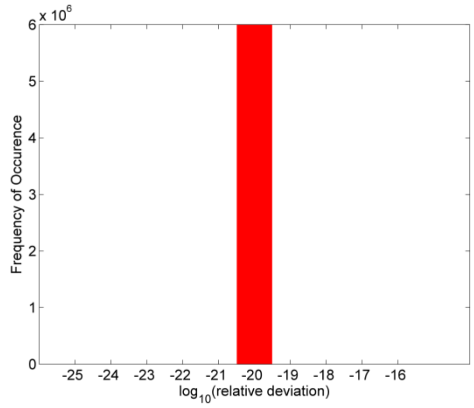


Figure 3-37 ENV_WIND-Operational_TC1 - 2

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 3-20 ENV_WIND-Operational_TC1 - 1

Comparison of Multiple Step Size Sweeps:			
All Deviations below Threshold		YES ☒	NO ☐
Absolute Deviations		**Relative Deviations**	
Average Absolute Deviation:	0.0	*Average Relative Deviation:*	0.0
Maximum Absolute Deviation:	0.0	*Maximum Relative Deviation:*	0.0
Absolute Standard Deviation:	0.0	*Relative Standard Deviation:*	0.0

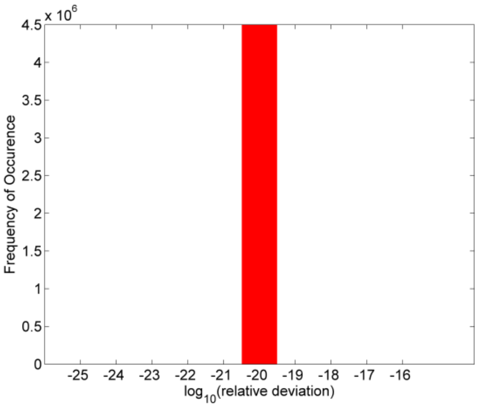
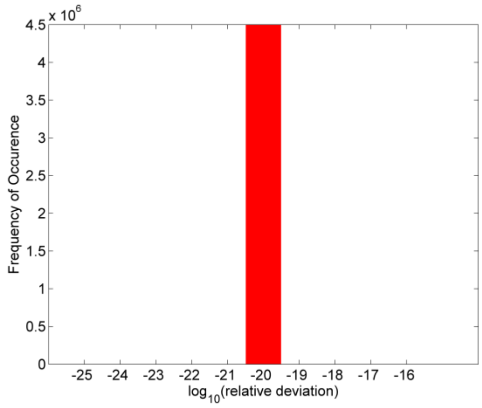

		

Figure 3-38 ENV_WIND-Operational_TC1 - 3		**Figure 3-39 ENV_WIND-Operational_TC1 - 4**	
Description of deviation exceeding the pre-specified threshold and assessment of possible causes			

Table 3-21 ENV_WIND-Operational_TC1 - 2

Run-Time Errors or Warnings?	YES ☐	NO ☒
Description of Error / Warning Messages		
Deficiencies in Operational Robustness?	YES ☐	NO ☒
Description of Detected Deficiencies		

Single Point Execution reproducible and deterministic?	**YES ☒**	**NO ☐**

3.8.3.2 Results for Code Generation and Equivalence Testing

Test Name:	**Code Generation and Equivalence Testing**
Test ID:	ENV_WIND-Operational_TC2
Verification Plan:	See section 3.7.3.2
Number of Test Points:	1,000,000
Execution Time:	33.59s
Rel. Deviation Threshold:	1.00e-13

Compilation of S-function successful?	YES ☒	NO ☐
Compilation of standalone executable successful?	YES ☒	NO ☐
Description of Compilation Errors		
Warnings during Compilation?	YES ☐	NO ☒
Description of Compilation Warnings		

Equivalence of S-function and Simulation Model			
All Deviations below Threshold		YES ☐	NO ☒
Absolute Deviations		**Relative Deviations**	
Average Absolute Deviation:	-1.08e-17	*Average Relative Deviation:*	-2.30e-17
Maximum Absolute Deviation:	9.24e-14	*Maximum Relative Deviation:*	1.18e-10
Absolute Standard Deviation:	6.88e-15	*Relative Standard Deviation:*	2.04e-13

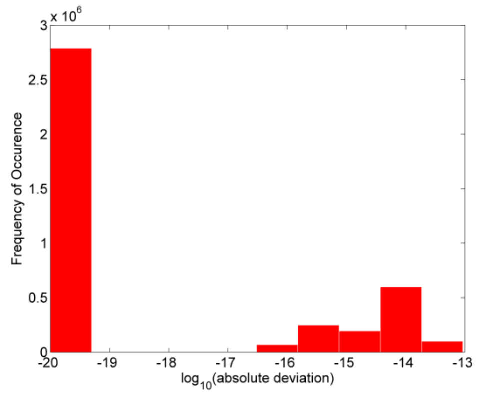


Figure 3-40 ENV_WIND-Operational_TC2 - 1

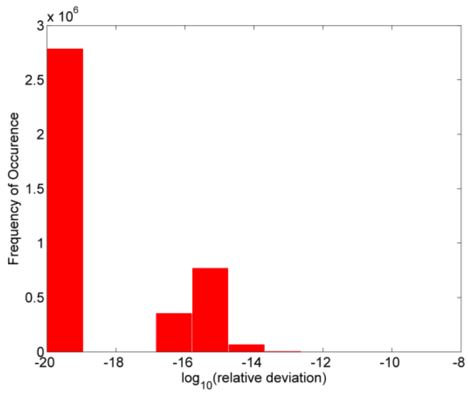


Figure 3-41 ENV_WIND-Operational_TC2 - 2

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

The computation involves numbers of very small and very huge magnitude, therefore numeric errors are assumed to cause the differences between the models

Table 3-22 ENV_WIND-Operational_TC2 - 1

Equivalence of Software-in-the-Loop (SIL) and Simulation Model			
All Deviations below Threshold		YES ☐	NO ☒
Absolute Deviations		**Relative Deviations**	
Average Absolute Deviation:	6.79e-18	*Average Relative Deviation:*	-2.24e-17
Maximum Absolute Deviation:	9.24e-14	*Maximum Relative Deviation:*	1.18e-10
Absolute Standard Deviation:	6.82e-15	*Relative Standard Deviation:*	2.04e-13

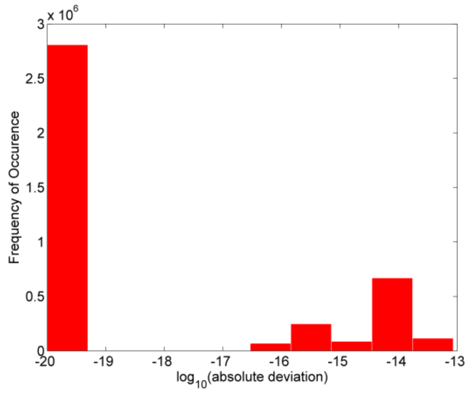
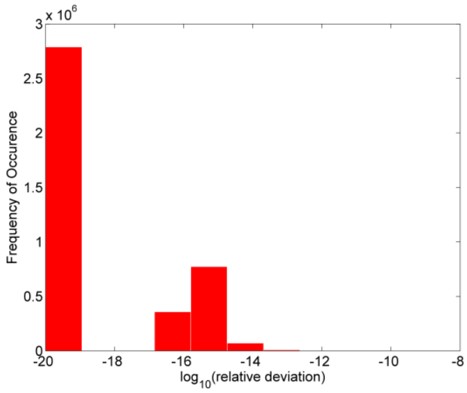

		

Figure 3-42 ENV_WIND-Operational_TC2 - 3		**Figure 3-43 ENV_WIND-Operational_TC2 - 4**	
Description of deviation exceeding the pre-specified threshold and assessment of possible causes			

Table 3-23 ENV_WIND-Operational_TC2 - 2

Run-Time Errors or Warnings?	YES ☐	NO ☒
Description of Error / Warning Messages The computation involves numbers of very small and very huge magnitude, therefore numeric errors are assumed to cause the differences between the models		

Code Generation successful and Coded Version equivalent to Simulation Model?	**YES ☒**	**NO ☐**

4 Turbulence Model

4.1 Introduction

The local air velocities are continuous and random in nature and definable only in a statistical sense.

Consequently the responses of the airplane can only be known statistically.

Of the methods of response calculations available, the spectral density approach is perhaps the best.

This approach provides statistical descriptions of the dynamic responses from a combination of a power spectral description of the turbulent velocities and solutions of linear equations of motion of the airplane. [4]

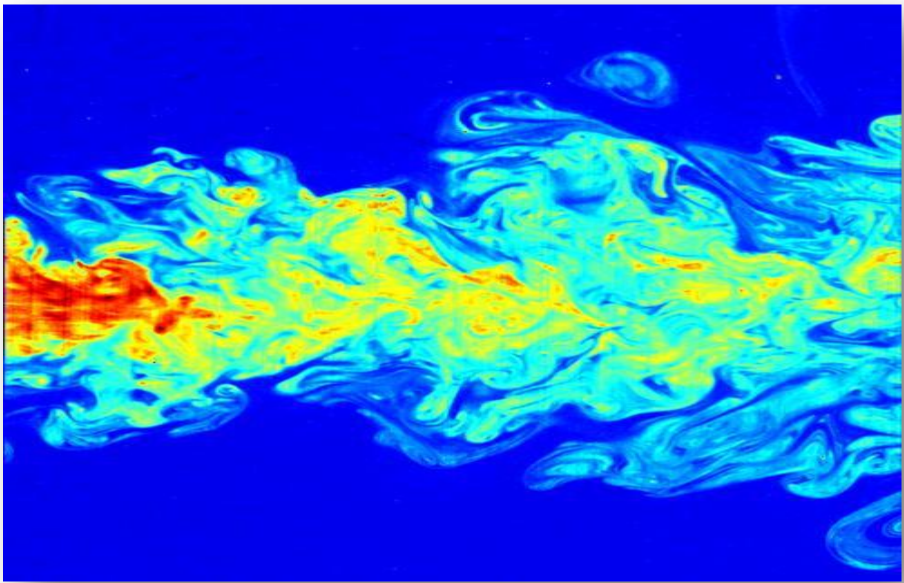


Figure 4-1 Far-field of a turbulent jet

4.2 Description of the Functional and Operational Intent

The system is intended to compute the effects of turbulence, allowing to calculate the velocity and the angular velocity of the air, according to Dryden Model of Turbulence [3].

In particular, in order to take into consideration the case in which there are a maximum of twenty aircraft flying in a limited area, the signals used by this turbulence model are not completely random but they take into account the correlation between the velocities perceived by each aircraft [5].

The final model will be used as part of a simulation model, which shall be incorporated into the simulation framework of a flight simulation device. Therefore it is necessary that all subsystems are compliant to code generation requirements.

4.3 Requirements

Requirements that the model must satisfy, are essentially of three types: functional, operational and implementation requirements. These requirements are summarized in the following tables and are named by the acronyms that allow a more rapid identification.

4.3.1 Functional Requirements

Requirement Name	Requirement ID
Computation of Air Velocity	R-FUN-ENV_TURB_01
Derived from Purpose of the system.	
Requirement Definition The system shall compute the Wind Velocity of the center of gravity *G* defined with respect to the Earth Centered Fixed (E) frame, components written in Body frame (B) .	

Table 4-1 R-FUN-ENV_TURB_01

Requirement Name	Requirement ID
Computation of Angular Velocity of the Air	R-FUN-ENV_TURB_02
Derived from Purpose of the system.	
Requirement Definition The system shall compute the Wind angular rate of the Wind frame (W) relative to the Earth Centered Fixed (E) frame, components written in Body frame (B) .	

Table 4-2 R-FUN-ENV_TURB_02

4.3.2 Operational Requirements

Requirement Name **Incorporation into Flight Simulator Simulation Model**	Requirement ID R-OPS-ENV_TURB
Derived from Usage intents	
Requirement Definition The model shall be incorporated as a child system into the simulation model of an (atmospheric) flight simulation device. Therefore it is necessary that all components support code generation.	

Table 4-3 R-OPS-ENV_TURB

4.3.3 Implementation Requirements

Requirement Name **Numeric Efficiency**	Requirement ID R-NUM-ENV_TURB
Derived from Global Implementation Guidelines	
Requirement Definition The coded algorithm must not contain any of the numerical inefficient programming techniques listed below unless detailed justification substantiates indispensability.	
Programming Techniques to be Avoided:	
Unused / Dead Code Branches	
Computational Redundancies	
Matrix Inversions	
Scalar Expansion of Vector / Matrix Math	
Circle Computations	
Inefficient Lookup Table Programming	
Algebraic Loops	

Table 4-4 R-NUM-ENV_TURB

Requirement Name	Requirement ID
Input / Output Interface Compliance to Parent System and Child Systems	R-IOC-ENV_TURB
Derived from Global Implementation Guidelines	
Requirement Definition Input and Output interface must comply with parent system.	
Compliance required to:	
Global bus object definitions	
I/O signal name matching to parent system	
I/O signal unit matching to parent system	
I/O signal data type matching to parent system	
I/O signal data range compatibility matching to parent system	

Table 4-5 R-IOC-ENV_TURB

Requirement Name	Requirement ID
Implementation Compliance to FSD Style Guidelines	R-SGC-ENV_TURB
Derived from Global Implementation Guidelines	
Requirement Definition Only a subset of SIMULINK blocks is allowed to be implemented.	
Allowed Libraries and Toolboxes:	
FSD Compliant Base	
Use of other Libraries and Toolboxes is Forbidden!	

Table 4-6 R-SGC-ENV_TURB

Requirement Name **Implementation Standards Compliance**	Requirement ID R-ISC-ENV_TURB
Derived from Global Implementation Guidelines	
Requirement Definition The coded algorithm must not contain any programming technique listed below unless detailed justification substantiates indispensability.	
Forbidden Programming Techniques:	
Discrete Switches*	
Memory Blocks	
Time Delays	
Time Dependent / Non-Autonomous Elements	
In-lined Integrations	
Hysteresis and Quantized Elements	
Stochastic / Random Elements	
Normal atan Blocks	
Operations with Sign Loss	
Value Flipping and Range Limiting	
Math Function out of Range	
Division by Zero	
Finite State Transition	

Table 4-7 R-ISC-ENV_TURB

* The switches in the system do not affect the correct operation.

4.4 Function Specification

4.4.1 Algorithm Abstract

The system is intended to compute the Wind velocity of the center of gravity G defined with respect to the Earth Centered Fixed frame, components written in Body frame $\left(\vec{v}^{G}_{W_{turb}}\right)^{E}_{B}$ and the Wind angular rate of the Wind frame relative to the Earth Centered Fixed frame, components written in Body frame $\left(\vec{\omega}^{EW}_{W_{turb}}\right)_{B}$ by the Dryden Model of the Turbulence.

4.4.2 Modeling Assumptions, Scope of Validity & Limitations

- The local air velocities are continuous and random in nature:
  The local air velocities are continuous and random in nature and definable only in a statistical sense. The method used to calculate the response of the aircraft in a statistical sense is the spectral density approach;
- Spectral approach: "one-dimensional analysis":
  The formulation of the spectral approach used is the "one-dimensional analysis": this formulation of the problem is characterized by the assumption that the airplane responds only to variations of the gust velocity along the flight path [4];
- Model used for the description of Turbulence:
  The expressions used for the description of atmospheric turbulence are the Dryden spectral density functions;
- Correlation:
  The correlation between components of the air velocities are given by the correlation tensor:
- The Turbulence field is frozen - Taylor's Hypothesis:
  It is assumed, that the Turbulence field is frozen; the wind turbulence is a stochastic function of position but it is not dependent on time;
- The characteristic length of the atmosphere is assumed to be given by the scale of turbulence:
  this length appears as a parameter in the mathematical description of turbulence and indicates the influence of the turbulence on the response of the aircraft [4].

4.4.3 Detailed Algorithm Description

Experience has shown that the local air velocities are continuous and random in nature and definable only in a statistical sense.

Consequently the motion of the atmosphere must be mathematically described as a random process.

The method used for the calculation of the response of the airplane is the spectral density approach; the use of this approach requires some important assumptions.

Intuitively the atmosphere should be described by time and spatial averages that vary with position and time respectively. In the model developed, it was considered that the atmosphere consists of patches of stationary and homogeneous turbulence.

The spectral density function can give a complete statistical description of the random process because the probability distribution of stationary and homogeneous patches of turbulence is nearly Gaussian.

In addition, for this representation the Taylor's hypothesis must be valid. This hypothesis assumes that the turbulence pattern is frozen until the airplane has passed through it; consequently the time displacements are equivalent to longitudinal space displacements. [4]

4.4.3.1 Mathematical Representation

The atmosphere will be assumed to be a continuous medium. The instantaneous velocity of air respect to the ground $\vec{V}_g(\vec{r},t)$ is, in general, a function of the position $\vec{r} = (\xi_1, \xi_2, \xi_3)$ and the time *t* and is assumed to be the sum of a steady velocity $\vec{V}_{g,s}(\vec{r},t)$ and a zero-mean turbulent fluctuation $\bar{u}(\vec{r},t)$:

$$\vec{V}_g(\vec{r},t) = \vec{V}_{g,s}(\vec{r},t) + \bar{u}(\vec{r},t) \qquad \text{4-1}$$

This turbulent fluctuation $\bar{u}(\vec{r},t)$ is mathematically described by a random vector field which is a function of time and spatial coordinates. The components of the turbulent fluctuation:

$$\bar{u}(\vec{r},t) = \begin{cases} u(\vec{r},t) \\ v(\vec{r},t) \\ w(\vec{r},t) \end{cases} \qquad 4\text{-}2$$

are usually rather small, at least small compared to the speed of sound that justified the assumption of incompressible fluid. A random function as $\bar{u}(\vec{r},t)$ is determined statistically by the complete set of joint-probability distribution of the values of $\bar{u}(\vec{r},t)$ at any *n* value of $\vec{r}$ and *t*. The joint-probability density functions are related to the complete set of mean-value velocity products that consists of components which are the statistical average of the product of *m* components of the velocities at *n* different points:

$$Q_{ij..p}^{(m)}(\vec{r},t) = E[\,u_i(\vec{r}_1,t_1)\cdot u_j(\vec{r}_2,t_2)\cdots u_p(\vec{r}_m,t_m)] \qquad 4\text{-}3$$

where $Q_{ij..p}^{(m)}(\vec{r},t)$ represents the m-order n-point velocity product mean value [4].

It has been shown that $Q_{ij..p}^{(m)}(\vec{r},t)$ for the value of *n=2* is sufficient to describe homogeneous and isotropic turbulence. This is the case of velocity components taken at two points and has the name *correlation tensor*. Using Taylor's hypothesis, the double correlation tensor (*m=2*) is:

$$R_{ij}(\vec{r}) = Q_{ij}^{(2)}(\vec{r}) = E[\,u_i(\vec{r}_1)\cdot u_j(\vec{r}_2)] \qquad 4\text{-}4$$

The statistical description of atmospheric turbulence is also given by a mathematical expression called spectral tensor, whose components are related to the components of the correlation tensor through a Fourier transform:

$$\Phi_{ij}(\Omega) = \frac{1}{\pi}\int_{-\infty}^{+\infty} R_{ij}(\vec{r})\cdot e^{-i\cdot\Omega\cdot\vec{r}}\,d\vec{r} \qquad 4\text{-}5$$

where Ω [rad/m] is the spatial frequency. [4]

The most common expression for the representation of atmospheric turbulence are the Dryden and Von Karman spectral density functions. Here the Dryden representation is used.

To make the formulas easier to read, with the symbols u_{turb}, v_{turb} *and* w_{turb}, the components of $\left(\vec{v}_{W_{turb}}^{G}\right)_{B}^{E}$ have been indicated and with the symbols p_{turb}, q_{turb} *and* r_{turb} the components of $\left(\vec{\omega}_{W_{turb}}^{EW}\right)_{B}$ have been indicated.

According to [3], the formulas are listed below:

$$\begin{cases} \Phi_{u_{turb}}(\Omega) = \sigma_u^2 \cdot 2 \cdot \frac{L_u}{\pi} \cdot \frac{1}{1 + (L_u \cdot \Omega)^2} \\ \Phi_{v_{turb}}(\Omega) = \sigma_v^2 \cdot 2 \cdot \frac{L_v}{\pi} \cdot \frac{1 + 3 \cdot (L_v \cdot \Omega)^2}{[1 + (L_v \cdot \Omega)^2]^2} \\ \Phi_{w_{turb}}(\Omega) = \sigma_w^2 \cdot 2 \cdot \frac{L_w}{\pi} \cdot \frac{1 + 3 \cdot (L_w \cdot \Omega)^2}{[1 + (L_w \cdot \Omega)^2]^2} \end{cases} \quad \text{4-6}$$

where Ω is the spatial frequency, L is the scale of turbulence and σ is the turbulence intensity.

The convention chosen for the angular velocities is as follows:

$$\begin{cases} p_{turb} = \frac{\partial w_{turb}}{\partial y} \\ q_{turb} = -\frac{\partial w_{turb}}{\partial x} \\ r_{turb} = \frac{\partial v_{turb}}{\partial x} \end{cases} \quad \text{4-7}$$

The corresponding spectra functions are:

$$\begin{cases} \Phi_{p_{turb}}(\Omega) = \frac{\sigma_w^2}{L_w} \cdot \frac{0.8\left(\frac{\pi L_w}{4b}\right)^{1/3}}{1 + \left(\frac{4b}{\pi} \cdot \Omega\right)^2} \\ \Phi_{q_{turb}}(\Omega) = \frac{\Omega^2}{1 + \left(\frac{4b}{\pi} \cdot \Omega\right)^2} \cdot \Phi_{w_{turb}} \\ \Phi_{r_{turb}}(\Omega) = \frac{\Omega^2}{1 + \left(\frac{3b}{\pi} \cdot \Omega\right)^2} \cdot \Phi_{v_{turb}} \end{cases} \quad \text{4-8}$$

where b is the wingspan.

If $\Omega = \frac{\omega}{V_{TAS}}$, where $\omega\ [rad/s]$ is the radian frequency, then $\Phi(\omega) = \frac{\Phi(\Omega)}{V_{TAS}}$.

The other parameters are defined according to altitude h^G:

Low Altitude ($h^G < 1000\,ft$)

The scale of turbulence in low altitude is defined by:

$$\begin{cases} L_w = h^G \\ L_u = L_v = \dfrac{h^G}{(0.177 + 0.000823 \cdot h^G)^{1.2}} \end{cases} \qquad 4\text{-}9$$

where h^G is the altitude in feet.

The turbulence intensities are:

$$\begin{cases} \sigma_w = 0.1 \cdot \left(v_W^{20ft}\right)_W^E \\ \dfrac{\sigma_u}{\sigma_w} = \dfrac{\sigma_v}{\sigma_w} = \dfrac{1}{(0.177 + 0.000823 \cdot h)^{0.4}} \end{cases} \qquad 4\text{-}10$$

where $\left(v_W^{20ft}\right)_W^E$ is the wind speed at $h^G = 20\,ft$.

In this region, the longitudinal turbulence velocity u_{turb} is aligned along the horizontal relative mean vector and the vertical turbulence velocity w_{turb} is aligned with the vertical.

High Altitude ($h^G > 2000\,ft$)

The scale of turbulence and the turbulence intensities are based on the assumption that the turbulence is isotropic. The scales of turbulence are:

$$L_u = L_v = L_w = 1750\,ft \qquad 4\text{-}11$$

The turbulence intensities are determined from a lookup table that provides the intensities as a function of altitude and probability of exceedance:

$$\sigma_u = \sigma_v = \sigma_w \qquad 4\text{-}12$$

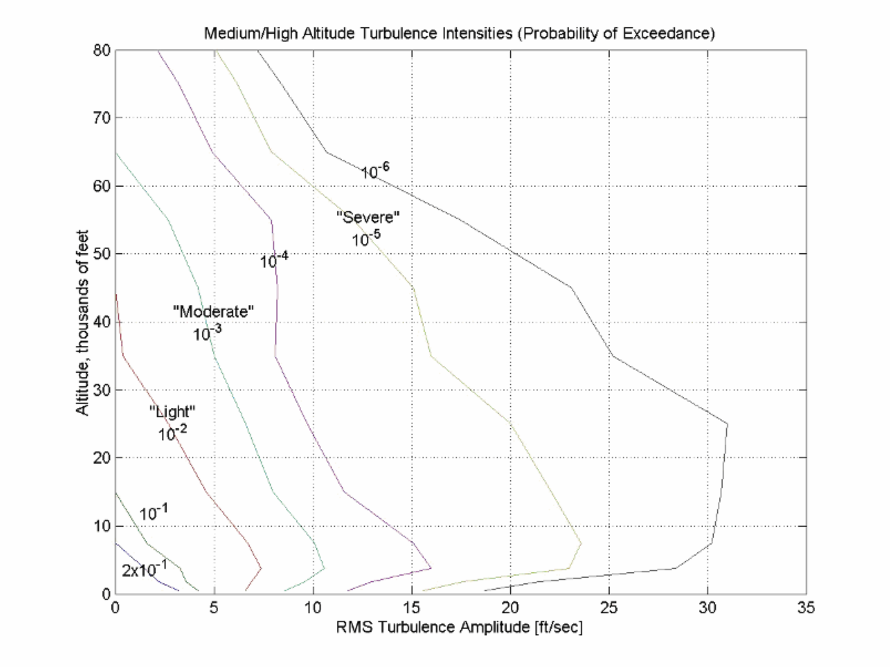


Figure 4-2 Turbulence intensities

As can be seen from the figure above, the probability of exceedance can be selected by numbers from 1 to 7 (from light to severe intensities).

The axes are aligned with the body coordinates.

Medium Altitude ($1000\,ft < h^G < 2000\,ft$)

At altitudes between $1000\,ft$ and $2000\,ft$, the scale of turbulence and the turbulence intensities are determined by a linear interpolation between the values obtained from the low altitude model, transformed from wind coordinates to body coordinates, $\left[\left(\vec{v}^{\mathrm{G}}_{\mathrm{W_{turb}}}\right)^E_B\right]_{\mathrm{LA}}$, and the values obtained from high altitude model in body coordinates, $\left[\left(\vec{v}^{\mathrm{G}}_{\mathrm{W_{turb}}}\right)^E_B\right]_{\mathrm{HA}}$.

The interpolation is realized by the following formulas:

$$\left(\vec{v}^G_{W_{turb}}\right)^E_B = w_{LA} \cdot \left[\left(\vec{v}^G_{W_{turb}}\right)^E_B\right]_{LA} + w_{HA} \cdot \left[\left(\vec{v}^G_{W_{turb}}\right)^E_B\right]_{HA} \qquad 4\text{-}13$$

where w_{HA} and w_{LA} are parameters defined by:

$$w_{HA} = \frac{h^G - 1000}{\Delta h} \quad (4\text{-}14)$$

$$w_{LA} = w_{HA} - 1 \quad (4\text{-}15)$$

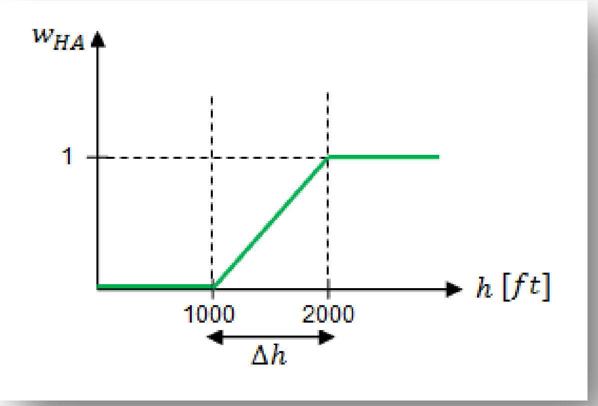


Figure 4-3 W_HA

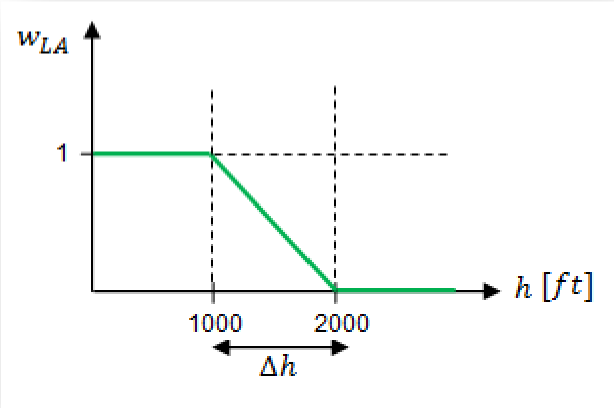


Figure 4-4 W_LA

Both parameters are defined in the range [0,1]. A similar procedure was used for the angular velocity.

It was decided to use for all parameters the SI units.

4.4.3.2 Generating a turbulence signal

In order to build signals representative of atmospheric turbulence, can be considered a process which is filtered through a linear filter with impulsive response $h(t)$. Let $H(j\omega)$ be the Fourier transform of $h(t)$, $n(t)$ be the input process to the filter and $y(t)$ the corresponding output. The main idea to generate a set of turbulence data is to use the equation:

$$\Phi_y(\omega) = \Phi_n(\omega) \cdot |H(j\omega)|^2 \quad \text{4-16}$$

where $\Phi_n(\omega)$ represents the spectra of the input and $\Phi_y(\omega)$ the spectra of the output. The input signal is a Gaussian white noise such that $\Phi_n(\omega) = 1$. [10]

The function $\Phi_y(\omega)$ is known because it is the spectral form that has been chosen, in this case the Dryden form. To generate the turbulence signal with the Gaussian white noise as input, it is necessary to find from equation 4-16 the "forming filter" $H(j\omega)$; since $H(j\omega) = H(s)|_{s=j\omega}$, the functions $y(s)$ are obtained from:

$$H(s) = \frac{y(s)}{n(s)} \quad \text{4-17}$$

Starting from the Dryden spectral density functions (equations 4-6 and 4-8) it is possible to obtain the various functions [7]:

- $\boldsymbol{u_{turb}(t), \dot{u}_{turb}(t)}$

$$H_u(s) = \sigma_u \sqrt{\frac{2L_u}{V_{TAS}\pi}} \frac{1}{\left(1 + \frac{L_u}{V_{TAS}} s\right)} \quad \text{4-18}$$

using the equation 4-17 with the white noise $n_u(t)$ as input, can be obtain the differential equation:

$$\dot{u}_{turb}(t) = \frac{V_{TAS}}{L_u}\left(\sigma_u \sqrt{\frac{2L_u}{V_{TAS}\pi}}\, n_u(t) - u_{turb}(t)\right) \quad \text{4-19}$$

❖ $\boldsymbol{v_{turb}(t), \ddot{v}_{turb}(t)}$

$$H_v(s) = \sigma_v \sqrt{\frac{L_v}{V_{TAS}\pi}} \frac{1 + \sqrt{3}\frac{L_v}{V_{TAS}} s}{\left(1 + \frac{L_v}{V_{TAS}} s\right)^2} \tag{4-20}$$

using the equation 4-17 with the white noise $n_v(t)$ as input, can be obtain the differential equation:

$$\ddot{v}_{turb}(t) + 2\frac{V_{TAS}}{L_v}\dot{v}_{turb}(t) + \left(\frac{V_{TAS}}{L_v}\right)^2 v_{turb}(t) = \\ = \sigma_v \left(\frac{V_{TAS}}{L_v}\right)^{\frac{3}{2}} \sqrt{\frac{1}{\pi}}\, n_v(t) + \sigma_v \sqrt{3\frac{V_{TAS}}{L_v\pi}}\, \dot{n}_v(t) \tag{4-21}$$

❖ $\boldsymbol{w_{turb}(t), \ddot{w}_{turb}(t)}$

$$H_w(s) = \sigma_w \sqrt{\frac{L_w}{V_{TAS}\pi}} \frac{1 + \sqrt{3}\frac{L_w}{V_{TAS}} s}{\left(1 + \frac{L_w}{V_{TAS}} s\right)^2} \tag{4-22}$$

using the equation 4-17 with the white noise $n_w(t)$ as input, can be obtain the differential equation:

$$\ddot{w}_{turb}(t) + 2\frac{V_{TAS}}{L_w}\dot{w}_{turb}(t) + \left(\frac{V_{TAS}}{L_w}\right)^2 w_{turb}(t) = \\ = \sigma_w \left(\frac{V_{TAS}}{L_w}\right)^{\frac{3}{2}} \sqrt{\frac{1}{\pi}}\, n_w(t) + \sigma_w \sqrt{3\frac{V_{TAS}}{L_w\pi}}\, \dot{n}_w(t) \tag{4-23}$$

❖ $\boldsymbol{p_{turb}(t), \dot{p}_{turb}(t)}$

$$H_p(s) = \sigma_w \sqrt{\frac{0.8}{V_{TAS}L_w}} \frac{\left(\frac{\pi L_w}{4b}\right)^{1/6}}{\left(1 + \frac{4b}{\pi V_{TAS}} s\right)} \tag{4-24}$$

using the equation 4-17 with the white noise $n_p(t)$ as input, can be obtain the differential equation:

$$\dot{p}_{turb}(t) = \frac{\pi V_{TAS}}{4b}\left(\sigma_w \sqrt{\frac{0.8}{V_{TAS}L_w}}\left(\frac{\pi L_w}{4b}\right)^{1/6} n_p(t) - p_{turb}(t)\right) \quad 4\text{-}25$$

- $\boldsymbol{q_{turb}(t), \dot{q}_{turb}(t)}$

$$H_q(s) = \frac{-\frac{s}{V_{TAS}}}{\left(1 + \frac{4b}{\pi V_{TAS}}s\right)} \cdot H_w(s) \quad 4\text{-}26$$

using the equation 4-17 can be obtain the differential equation:

$$\dot{q}_{turb}(t) + \frac{\pi V_{TAS}}{4b} q_{turb}(t) = -\frac{\pi}{4b}\dot{w}_{turb}(t) \quad 4\text{-}27$$

- $\boldsymbol{r_{turb}(t), \dot{r}_{turb}(t)}$

$$H_r(s) = \frac{\frac{s}{V_{TAS}}}{\left(1 + \frac{3b}{\pi V_{TAS}}s\right)} \cdot H_v(s) \quad 4\text{-}28$$

using the equation 4-17 can be obtain the differential equation:

$$\dot{r}_{turb}(t) + \frac{\pi V_{TAS}}{3b} r_{turb}(t) = \frac{\pi}{3b}\dot{v}_{turb}(t) \quad 4\text{-}29$$

In the simulation model, the equations have been implemented in the form given by reference [6].

4.4.3.3 Change of frame: sign convention

To get the results in body axes, it is necessary to make some changes of frame. The following is the convention used to switch from Wind Frame to NED Frame:

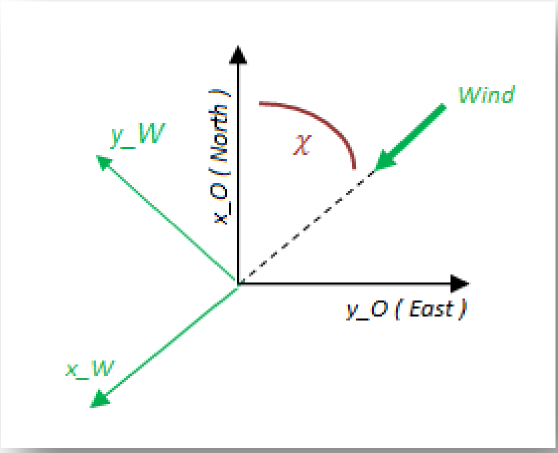


Figure 4-5 From Wind Frame (W) to NED Frame (O)

$$\begin{cases} u_o = -u_W \cdot \cos\chi_W^G + v_W \cdot \sin\chi_W^G \\ v_o = -u_W \cdot \sin\chi_W^G - v_W \cdot \cos\chi_W^G \\ w_o = w_W \end{cases} \qquad 4\text{-}30$$

In the Dryden Continuous Model in Simulink Toolbox, the sign convention is the opposite for the two components u_o and v_o:

$$\begin{cases} u_o = u_W \cdot \cos\chi_W^G - v_W \cdot \sin\chi_W^G \\ v_o = u_W \cdot \sin\chi_W^G + v_W \cdot \cos\chi_W^G \\ w_o = w_W \end{cases} \qquad 4\text{-}31$$

For this reason some differences in the results were found; using in the implementation the same sign convention of equations 4-31, similar results are obtained (see section 4.8.2.1)

4.4.3.4 Algorithm for Implementation

In the implemented system, the equations 4-1 to 4-31 are used.

4.5 Architecture Specification

4.5.1 Parent / Child Systems

4.5.1.1 Parent System

The system will be embedded into the system “Atmosphere”, which will be embedded into the parent system "Environment", whose purpose is to simulate all processes regarding the environment of the aircraft (atmosphere, terrain model, earth model, etc.).

4.5.1.2 Child Systems

This system does not contain any child systems.

4.5.2 Signal Definitions

4.5.2.1 Inputs

Inputs							
Symbol	**Name**	**Size**	**Components**	**Data Type**	**Min**	**Max**	**Description**
Ψ	Psi_rad	[1 1]	-	double	-pi	pi	Yaw angle of the aircraft
Θ	Theta_rad	[1 1]	-	double	-pi/2	pi/2	Pitch angle of the aircraft
Φ	Phi_rad	[1 1]	-	double	-pi	pi	Roll angle of the aircraft
φ^G	phi_G_WGS84_rad	[1 1]	-	double	-pi/2	pi/2	Geodetic Latitude of the aircraft's center of gravity.
λ^G	lambda_G_WGS84_rad	[1 1]	-	double	-pi	pi	Geodetic Longitude of the aircraft's center of gravity.
h^G	h_G_WGS84_m	[1 1]	-	double	-500	20000	Height of the aircraft's center of gravity above the WGS84 reference ellipsoid.
$\left(v_W^{20ft}\right)_W^E$	vel_W_20ft_E_W_mDs	[1 1]	-	double	-Inf	+Inf	Wind velocity at an altitude of 20 ft, defined with respect to the Earth Centered Fixed (E) frame, components written in W frame
$Prob_{Exc}$	Probability_Exceedence	[1 1]	-	double	1	7	Probability of the turbulence intensity being exceeded.
b	b_m	[1 1]	-	double	0	Inf	Wingspan

Table 4-8 Inputs

Inputs							
Symbol	**Name**	**Size**	**Components**	**Data Type**	**Min**	**Max**	**Description**
V_{TAS}	V_TAS_mDs	[1 1]	-	double	-	-	True Air Speed perceived by the aircraft.
n_u^{HA}	n_u_HA	[1 1]	-	double	-Inf	Inf	Random signal required to calculate u_{turb} in the model of high altitude.
n_v^{HA}	n_v_HA	[1 1]	-	double	-Inf	Inf	Random signal required to calculate v_{turb} in the model of high altitude.
n_w^{HA}	n_w_HA	[1 1]	-	double	-Inf	Inf	Random signal required to calculate w_{turb} in the model of high altitude.
n_p^{HA}	n_p_HA	[1 1]	-	double	-Inf	Inf	Random signal required to calculate p_{turb} in the model of high altitude.
n_u^{LA}	n_u_LA	[1 1]	-	double	-Inf	Inf	Random signal required to calculate u_{turb} in the model of low altitude.
n_v^{LA}	n_v_LA	[1 1]	-	double	-Inf	Inf	Random signal required to calculate v_{turb} in the model of low altitude.
n_w^{LA}	n_w_LA	[1 1]	-	double	-Inf	Inf	Random signal required to calculate w_{turb} in the model of low altitude.
n_p^{LA}	n_p_LA	[1 1]	-	double	-Inf	Inf	Random signal required to calculate p_{turb} in the model of low altitude.

Table 4-8 Inputs Part II

4.5.2.2 Outputs

Outputs							
Symbol	**Name**	**Size**	**Components**	**Data Type**	**Min**	**Max**	**Description**
$\left(\vec{v}^{G}_{W_{turb}}\right)^{E}_{B}$	Vel_W_turb_G_E_B_mDs	[3 1]	u_W_turb_G_E_B_mDs v_W_turb_G_E_B_mDs w_W_turb_G_E_B_mDs	double	-Inf -Inf -Inf	Inf Inf Inf	Wind Velocity of the center of gravity *G* defined with respect to the Earth Centered Fixed (E) frame, components written in Body frame (B) .
$\left(\vec{\omega}^{EW}_{W_{turb}}\right)_{B}$	rot_W_turb_EW_B_radDs	[3 1]	p_W_turb_EW_B_radDs q_W_turb_EW_B_radDs r_W_turb_EW_B_radDs	double	-Inf -Inf -Inf	Inf Inf Inf	Wind angular rate of the Wind frame (W) relative to the Earth Centered Fixed (E) frame, components written in Body frame (B) .

Table 4-9 Outputs

4.5.3 Bus Structure

To facilitate the transport of signals, were often created the buses.

In the following tables, are then collected buses created for input and output signals.

<u>Inputs</u>

L	Bus Name	Elements	Element Types
0	*pos_G_WGS84_Bus*	phi_G_WGS84_rad lambda_G_WGS84_rad h_G_WGS84_m	double double double
0	*att_euler_Bus*	Psi_rad Theta_rad Phi_m	double double double
0	*Noise_Bus*	Noise_HA_Bus Noise_LA_Bus	Noise_HA_Bus Noise_LA_Bus
1	*Noise_HA_Bus*	n_u_HA n_v_HA n_w_HA n_p_HA	double double double double
1	*Noise_LA_Bus*	n_u_LA n_v_LA n_w_LA n_p_LA	double double double double

Table 4-10 Inputs Bus Structure

Outputs

L	Bus Name	Elements	Element Types
0	*Vel_W_turb_G_E_B_Bus*	u_W_turb_G_E_B_mDs v_W_turb_G_E_B_mDs w_W_turb_G_E_B_mDs	double double double
0	*rot_W_turb_EW_B_Bus*	p_W_turb_EW_B_radDs q_W_turb_EW_B_radDs r_W_turb_EW_B_radDs	double double double

Table 4-11 Outputs Bus Structure

4.6 Structural Layout

Figure 4-6 L0: ENV_TURB

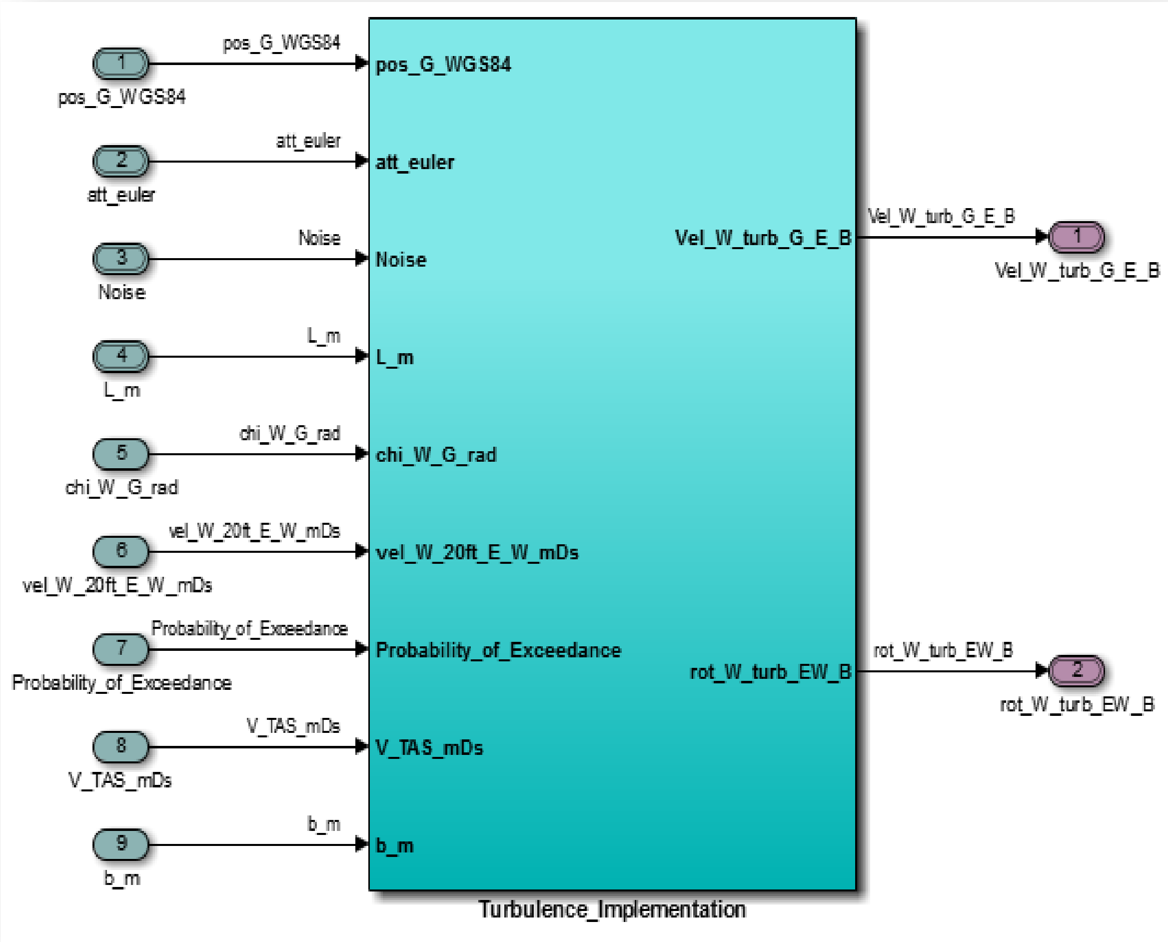


Figure 4-7 L1: Turbulence_Implementation

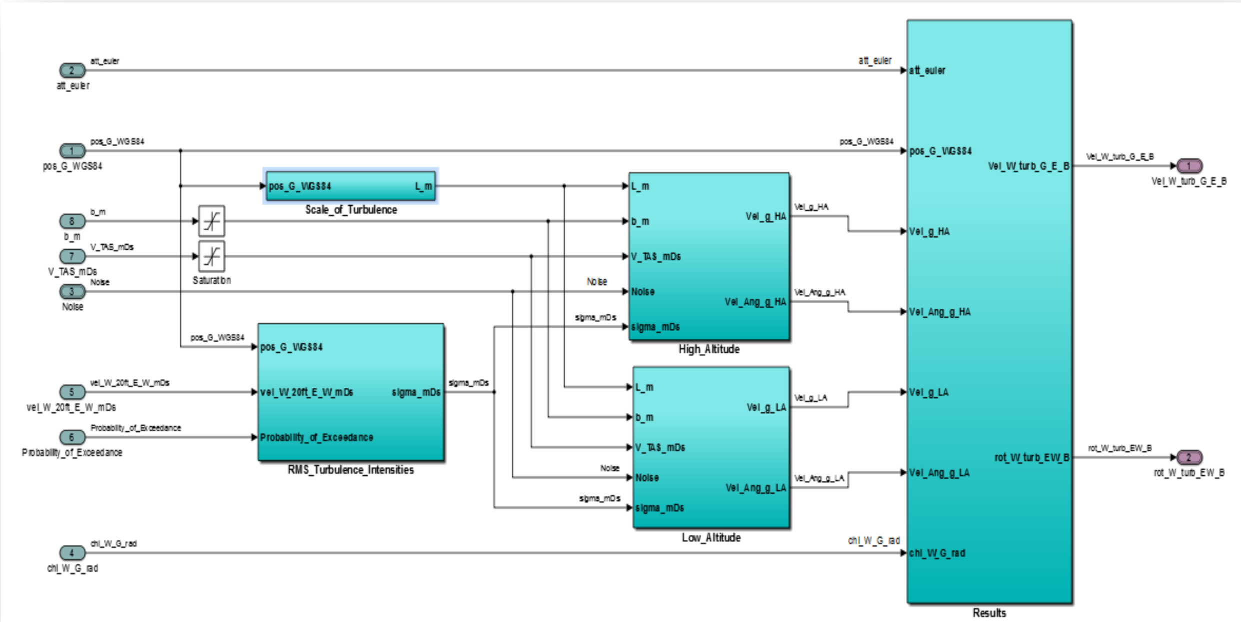


Figure 4-8 L2: High_Altitude

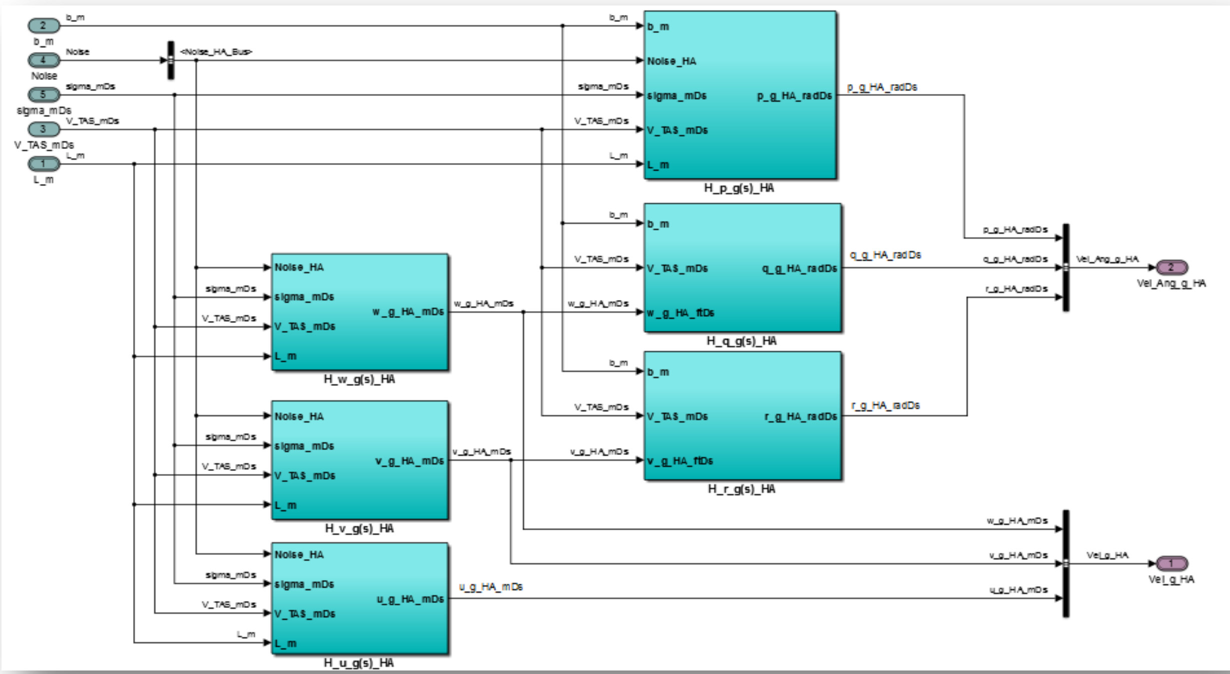


Figure 4-9 L2: Low_Altitude

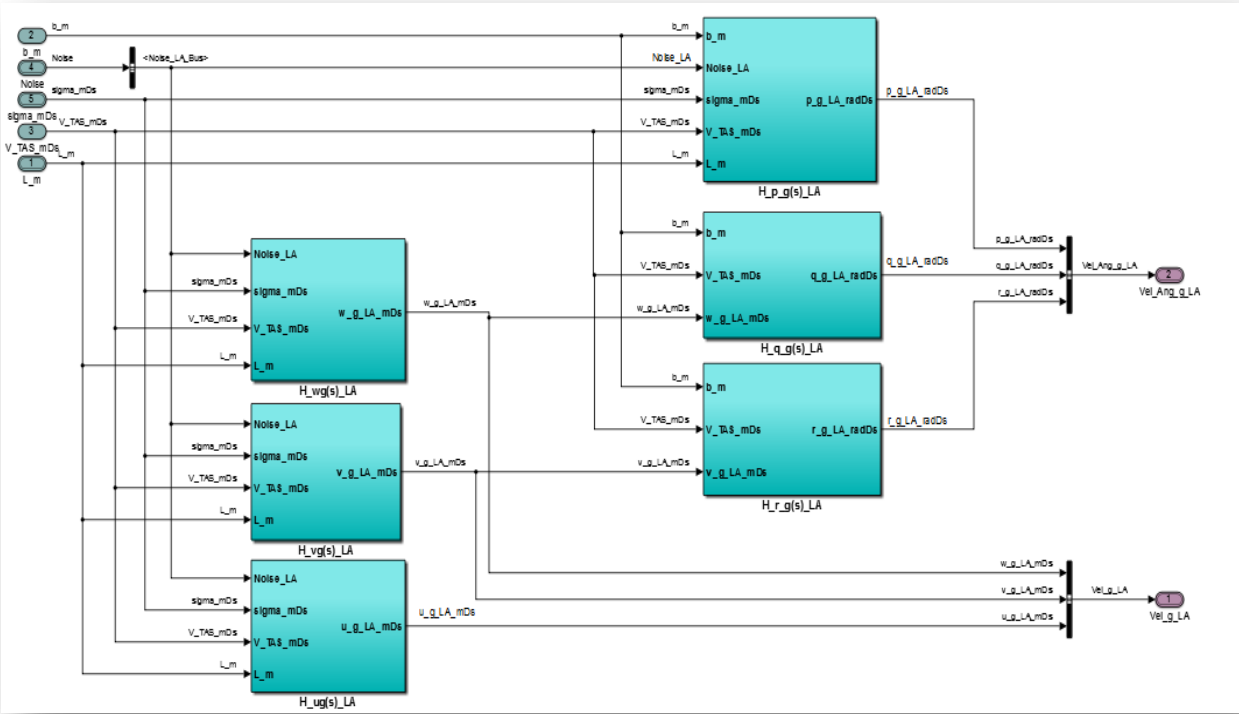


Figure 4-10 L2: Results

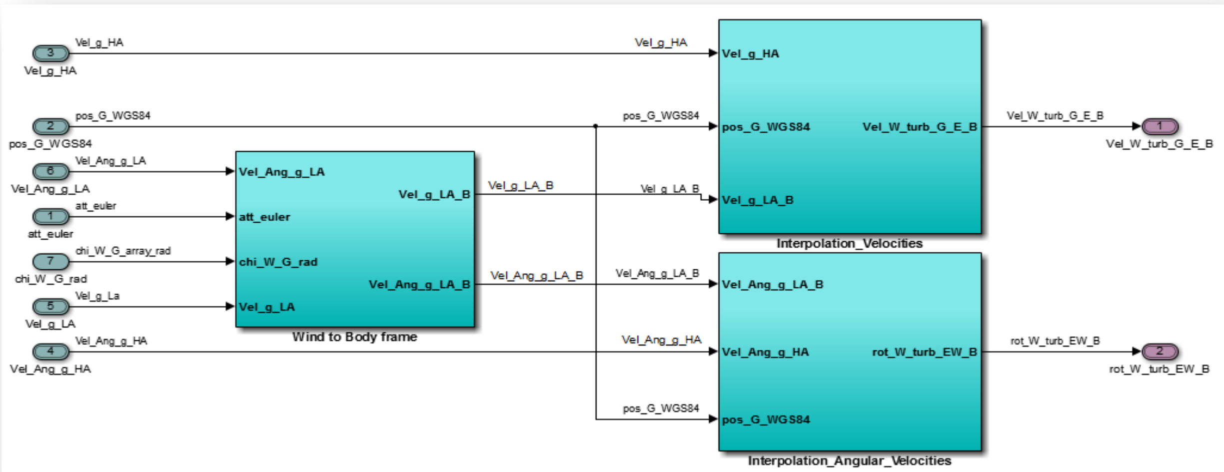


Figure 4-11 L2: RMS_Turbulence_Intensities

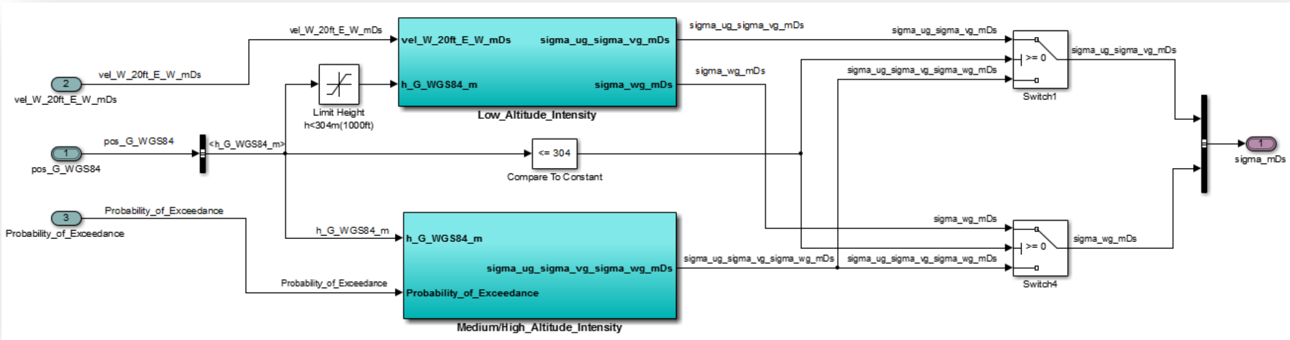


Figure 4-12 L2: Scale_of_Turbulence

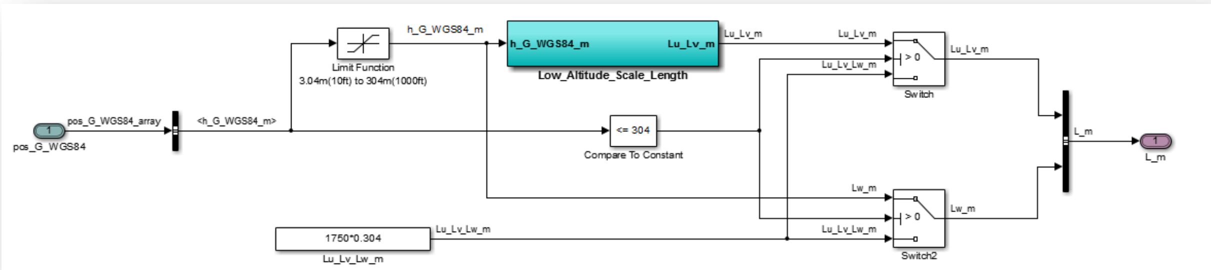


Figure 4-13 L3: H_p_g(s)_HA

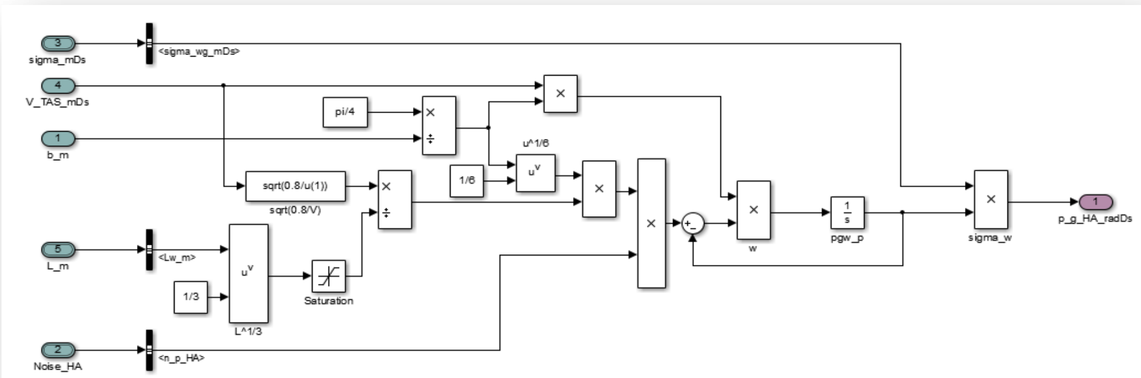


Figure 4-14 L3: H_q_g(s)_HA

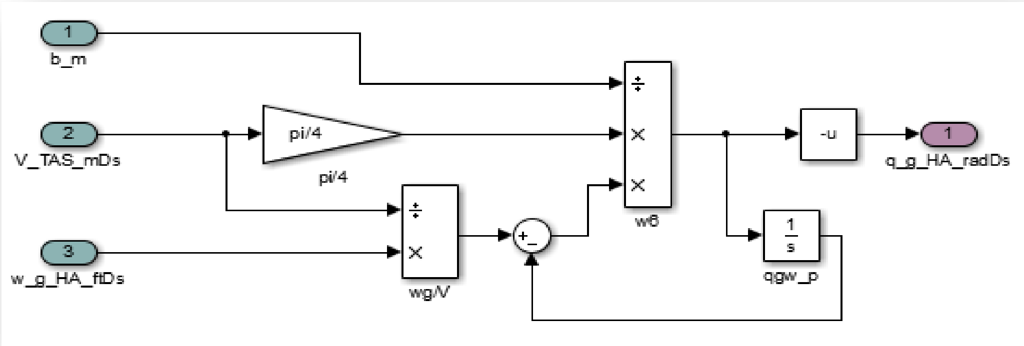


Figure 4-15 L3: H_r_g(s)_HA

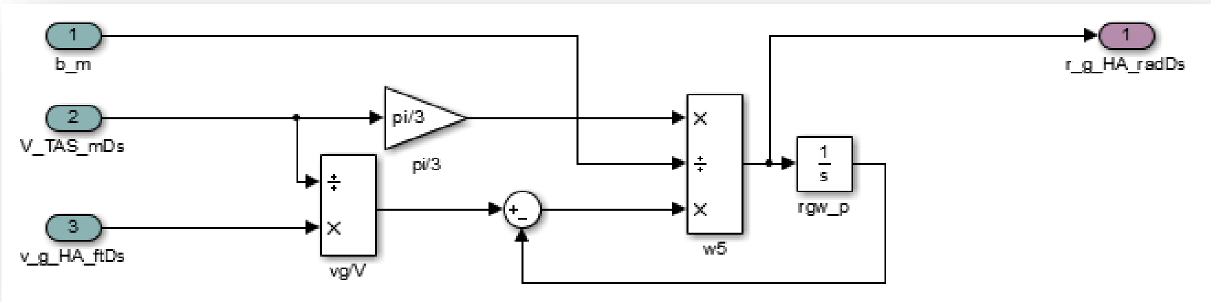


Figure 4-16 L3: H_u_g(s)_HA

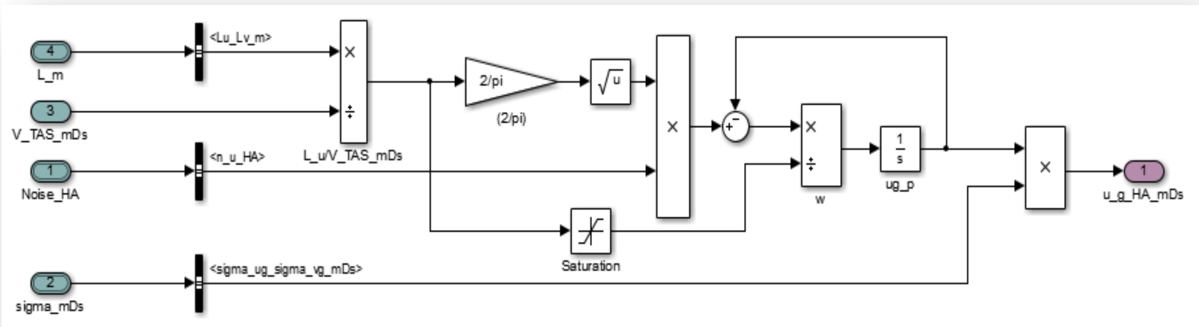


Figure 4-17 L3: H_v_g(s)_HA

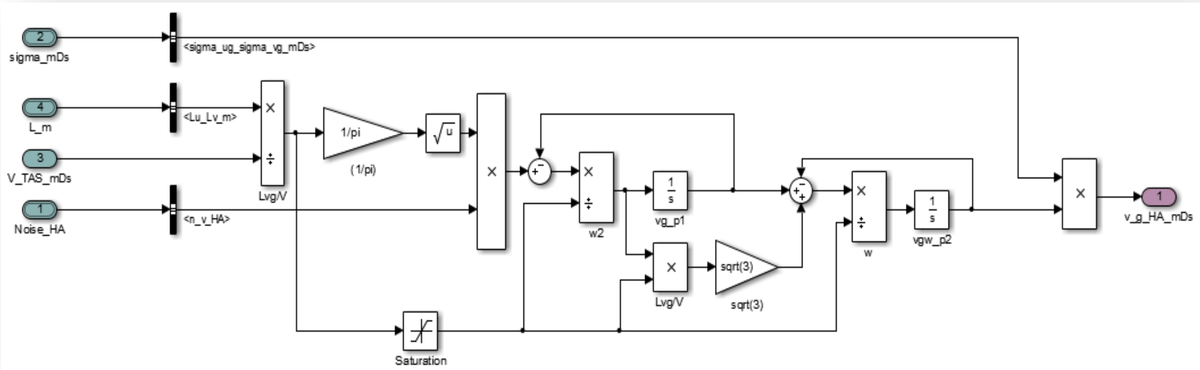


Figure 4-18 L3: H_w_g(s)_HA

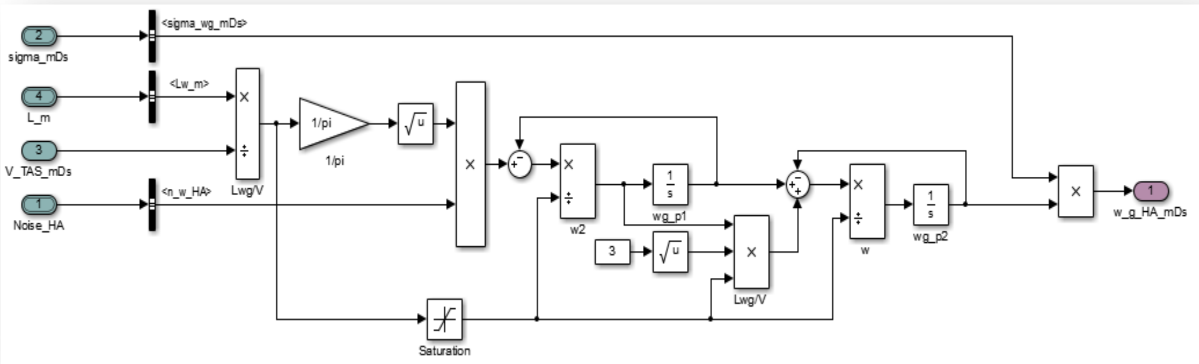


Figure 4-19 L3: H_p_g(s)_LA

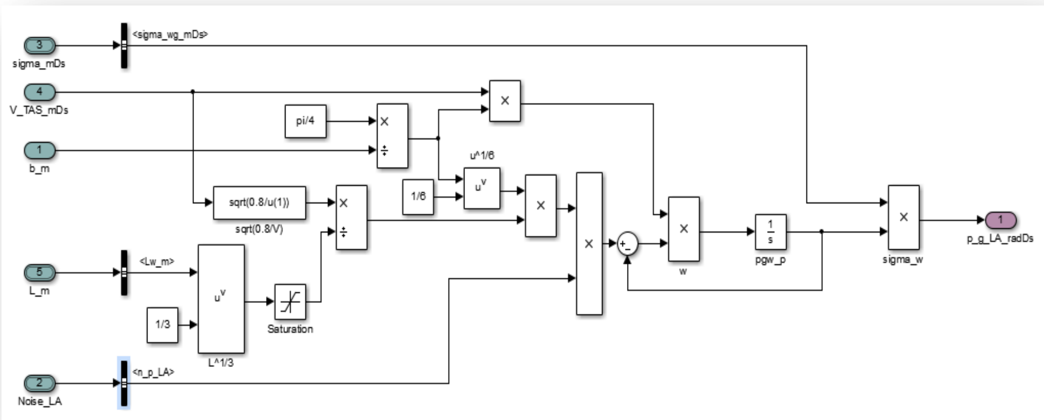


Figure 4-20 L3: H_q_g(s)_LA

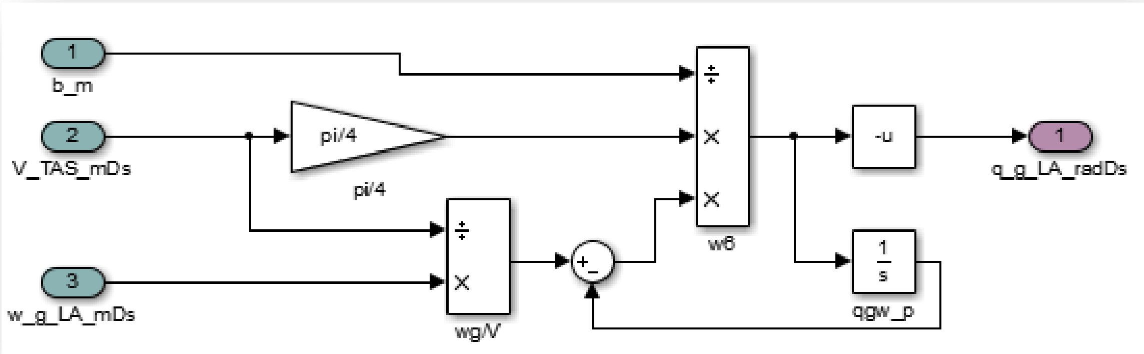


Figure 4-21 L3: H_r_g(s)_LA

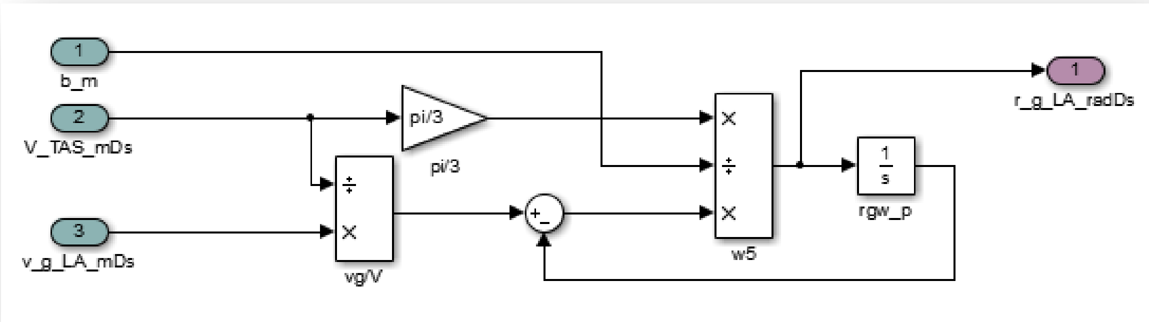


Figure 4-22 L3: H_u_g(s)_LA

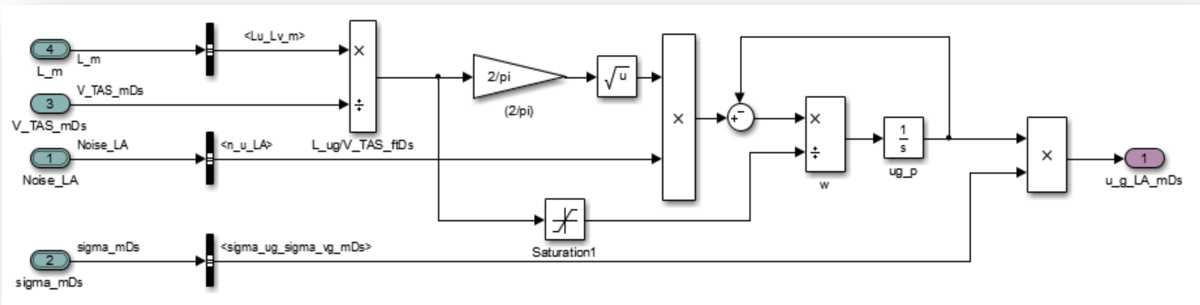


Figure 4-23 L3: H_v_g(s)_LA

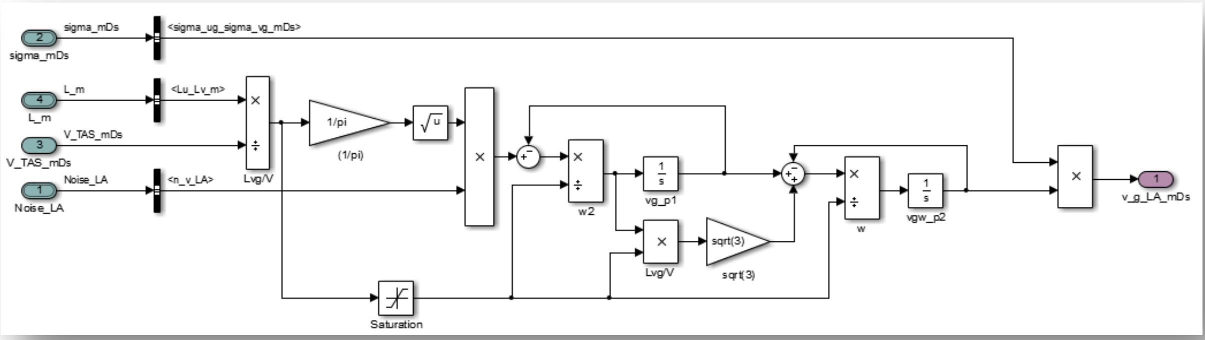


Figure 4-24 L3: H_w_g(s)_LA

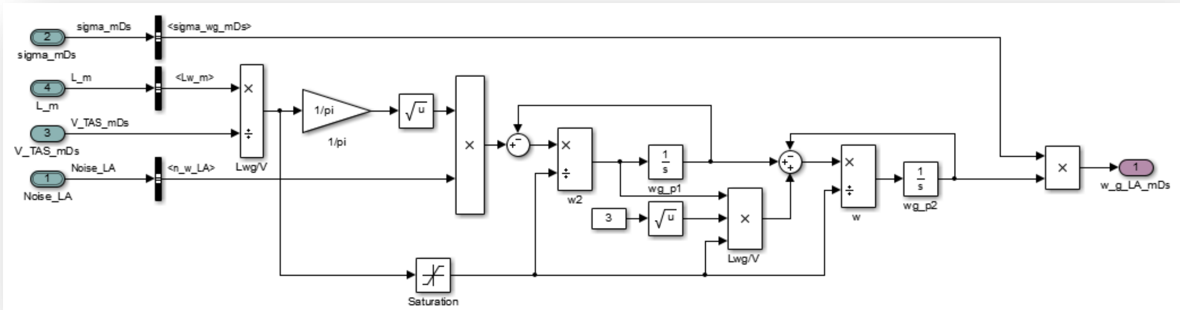


Figure 4-25 L3: Low_Altitude_Scale_Length

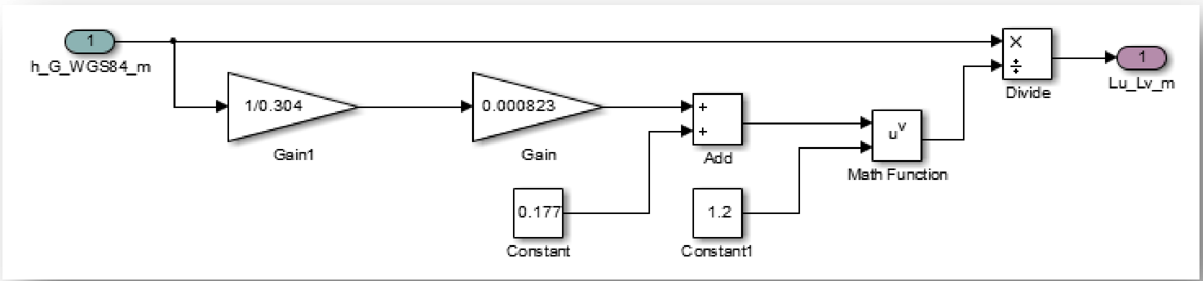


Figure 4-26 L3:Interpolation_Angular_Velocities

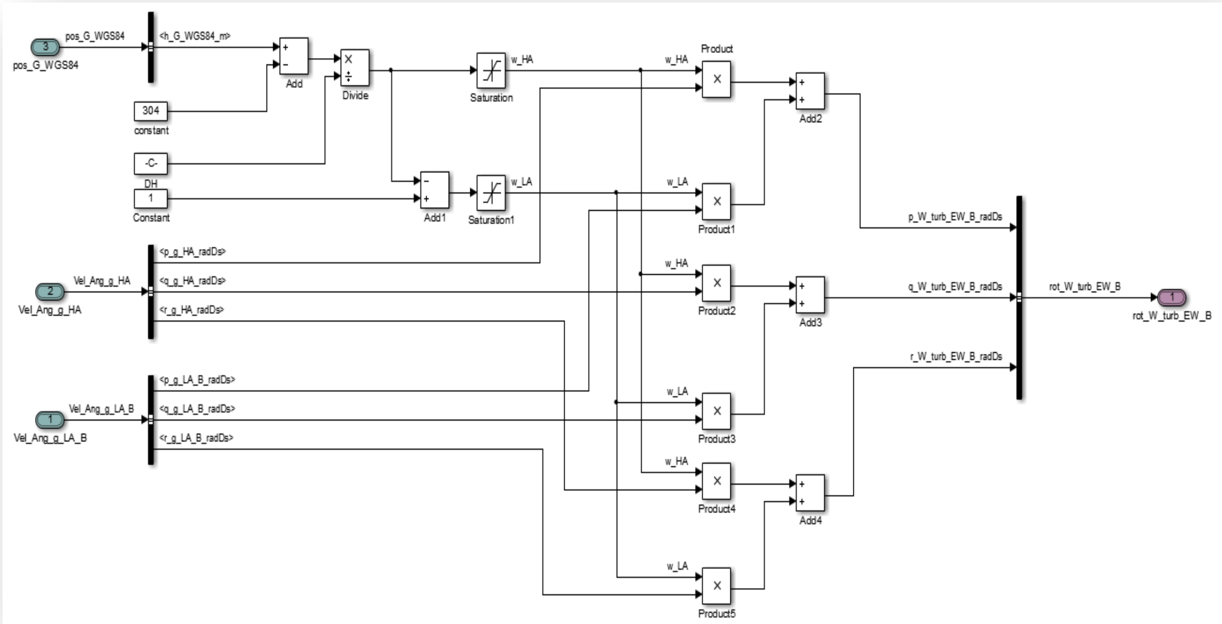


Figure 4-27 L3: Interpolation_Velocities

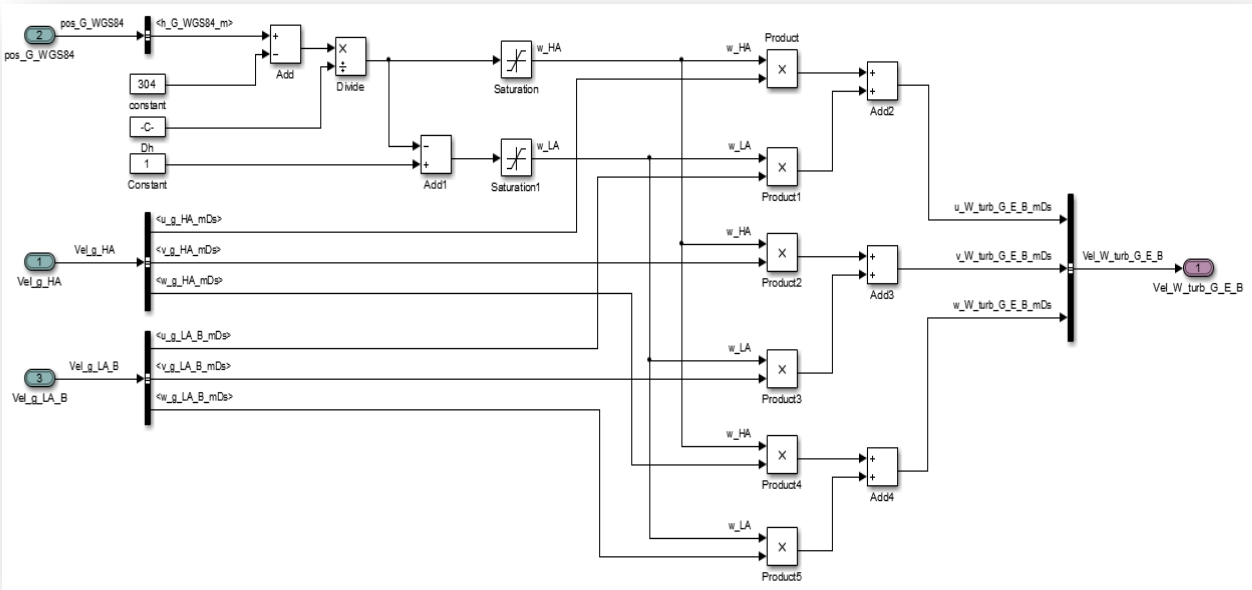


Figure 4-28 L3: Medium/High_Altitude_Intensity

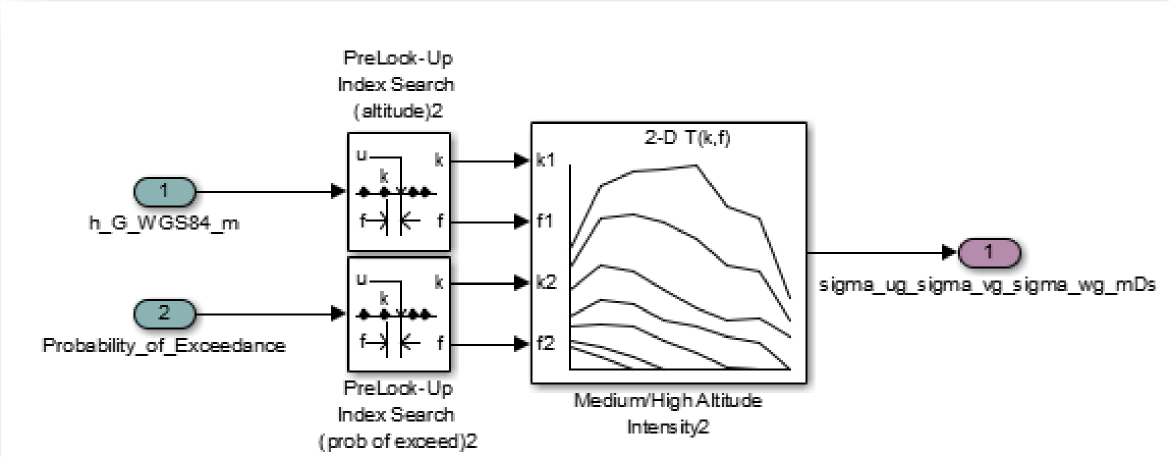


Figure 4-29 L3: Low_Altitude_Intensity

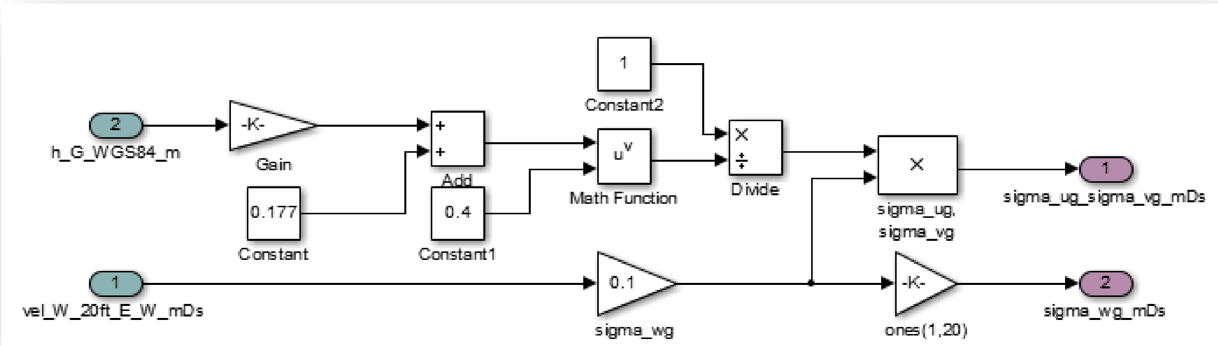


Figure 4-30 L3: Wind_to_Body_Frame

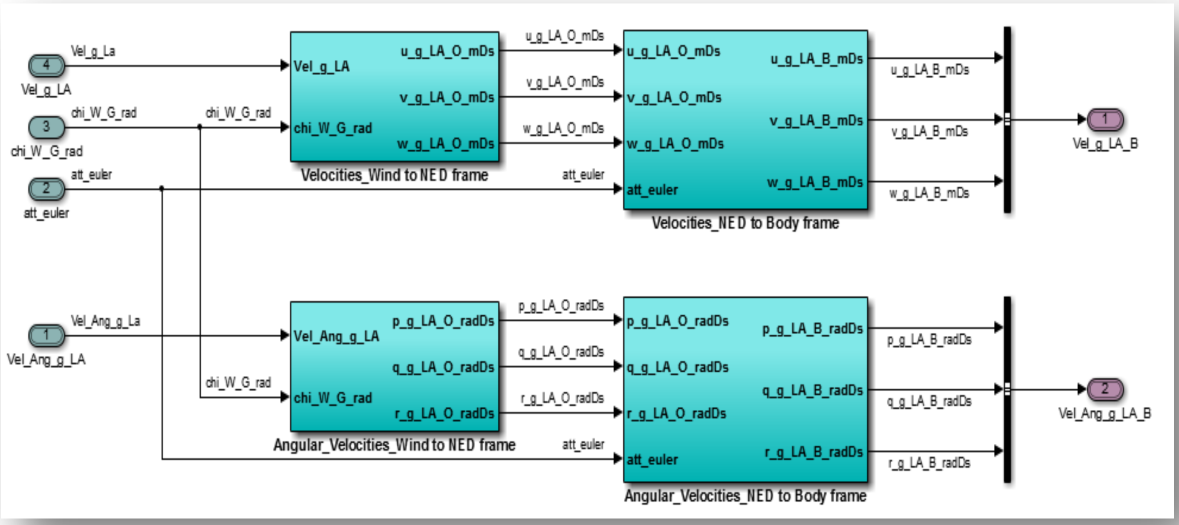


Figure 4-31 L4: Angular_Velocities_NED_to_Body_Frame

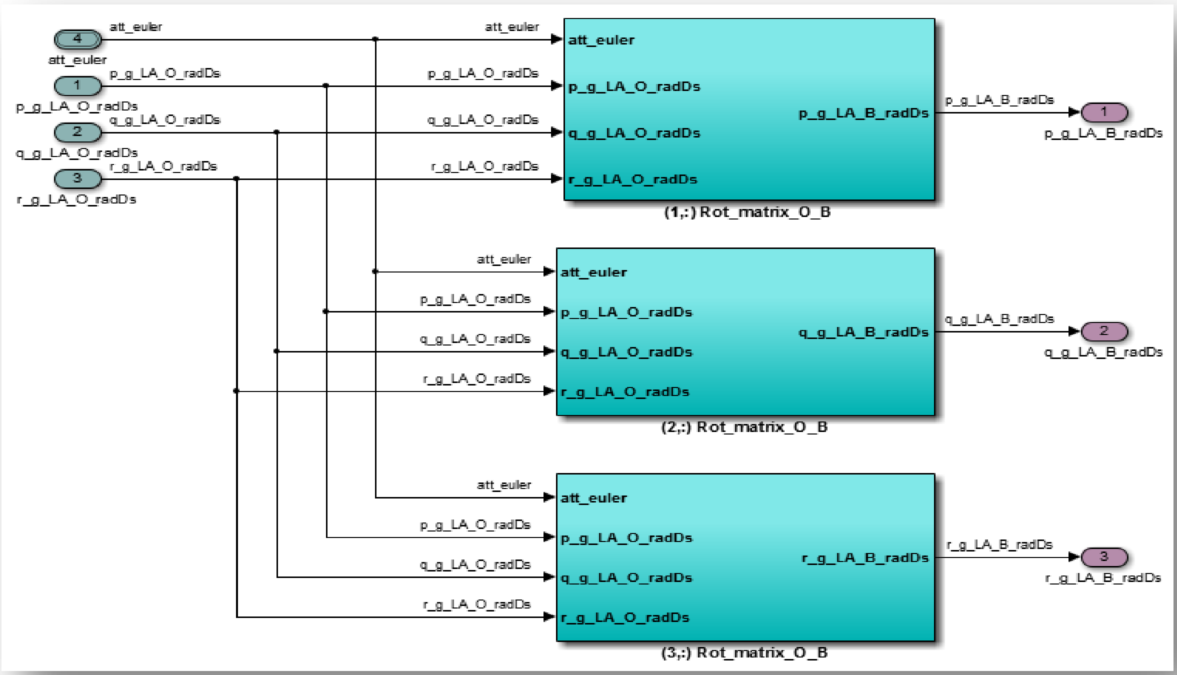


Figure 4-32 L4: Angular_Velocities_Wind_to_NED_Frame

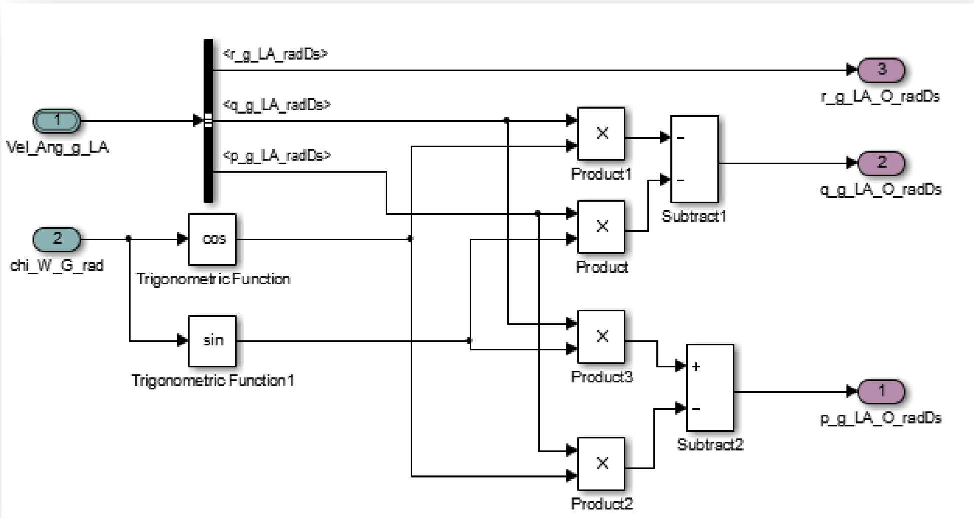


Figure 4-33 L4: Velocities_NED_to_Body_Frame

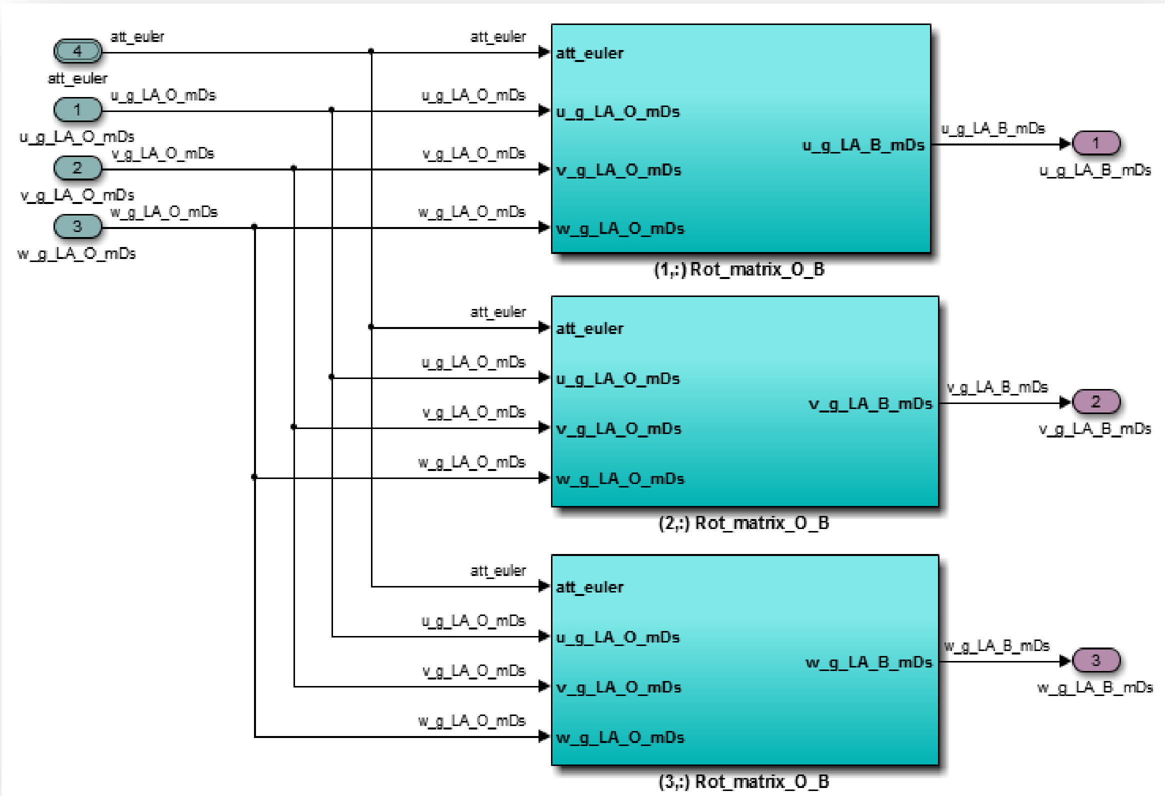


Figure 4-34 L4: Velocities_Wind_to_NED_Frame

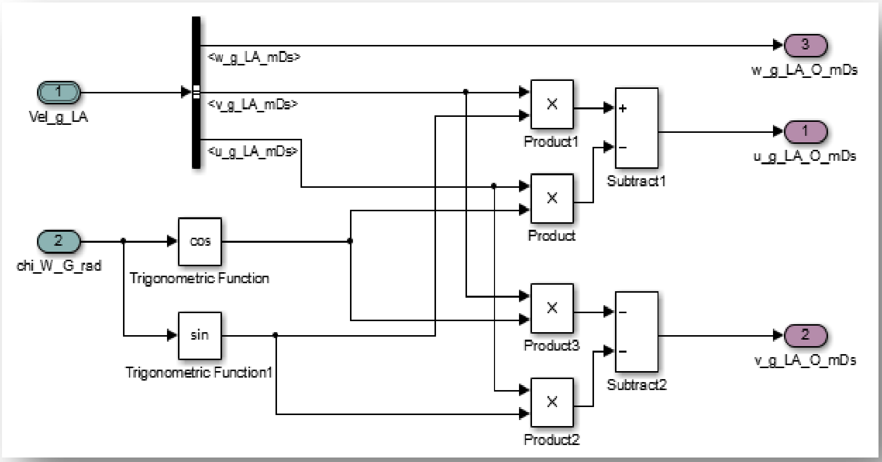


Figure 4-35 L5: (1,:) Rot_matrix_Ang_O_B

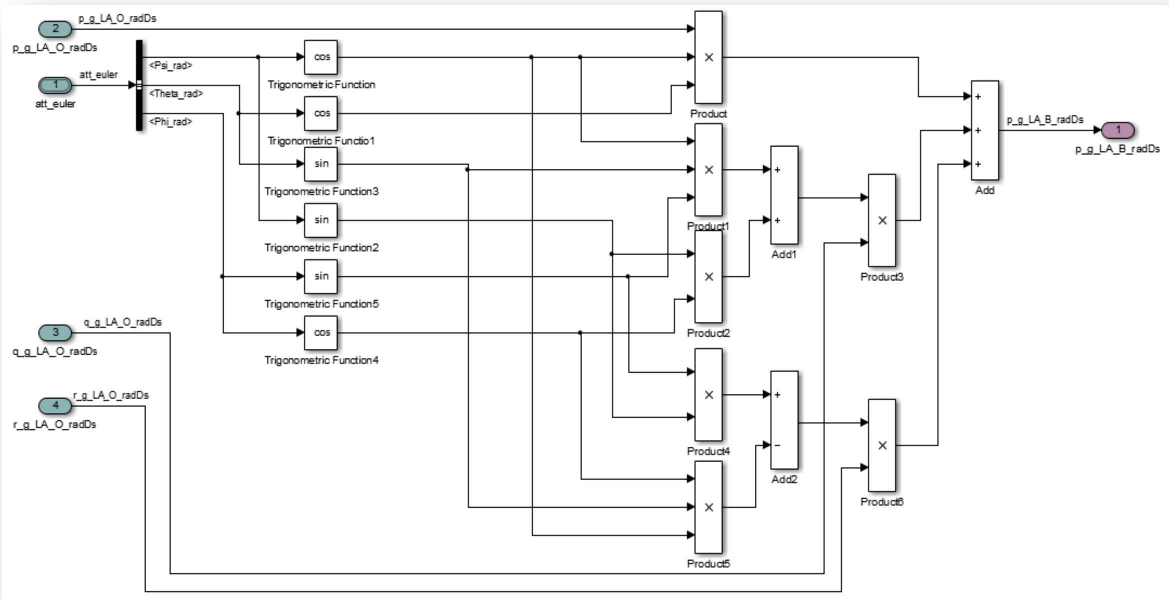


Figure 4-36 L5: (2,:) Rot_matrix_Ang_O_B

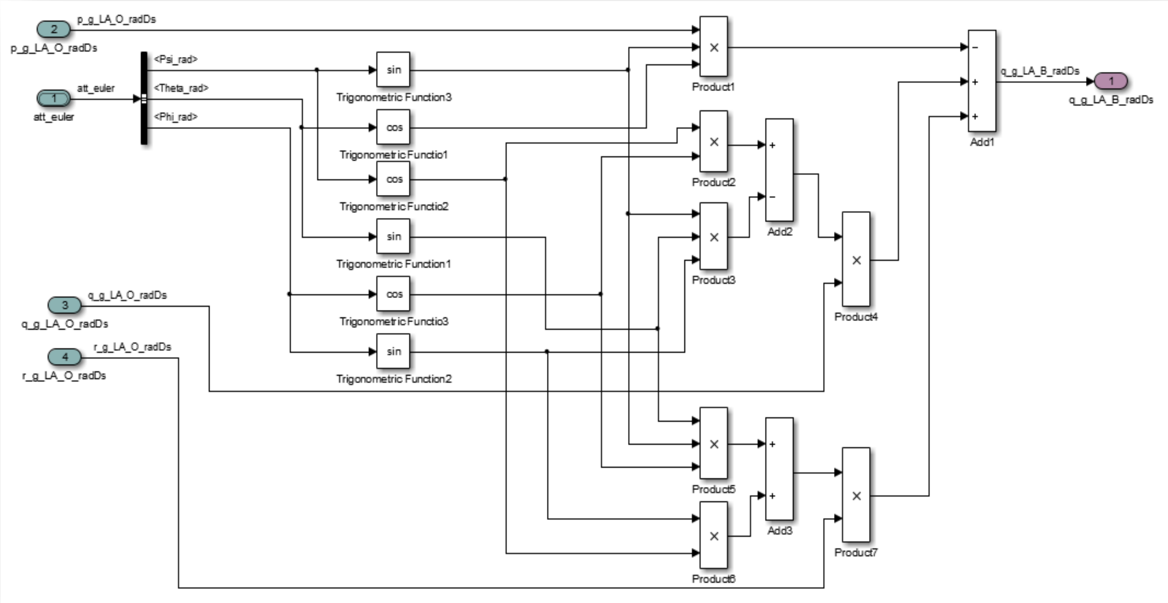


Figure 4-37 L5: (3,:) Rot_matrix_Ang_O_B

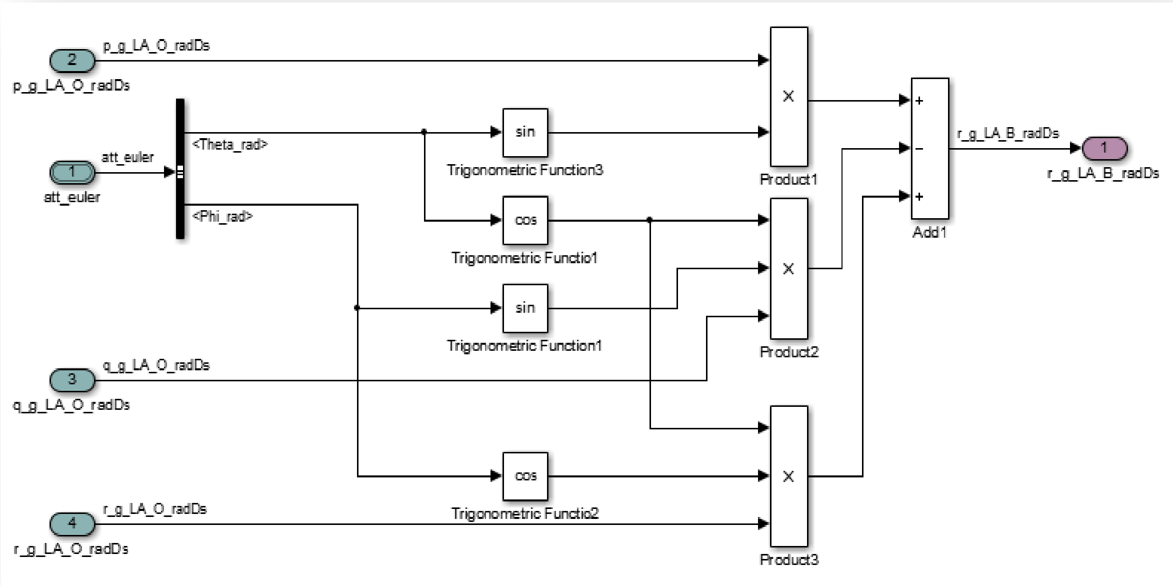


Figure 4-38 L5: (1,:) Rot_matrix_Vel_O_B

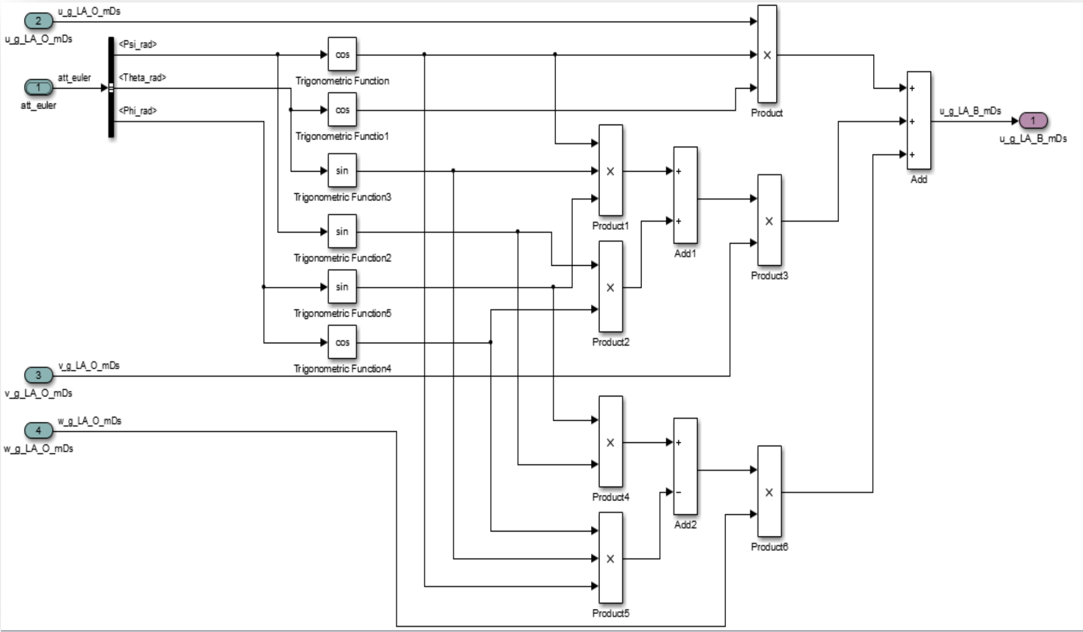


Figure 4-39 L5: (2,:) Rot_matrix_Vel_O_B

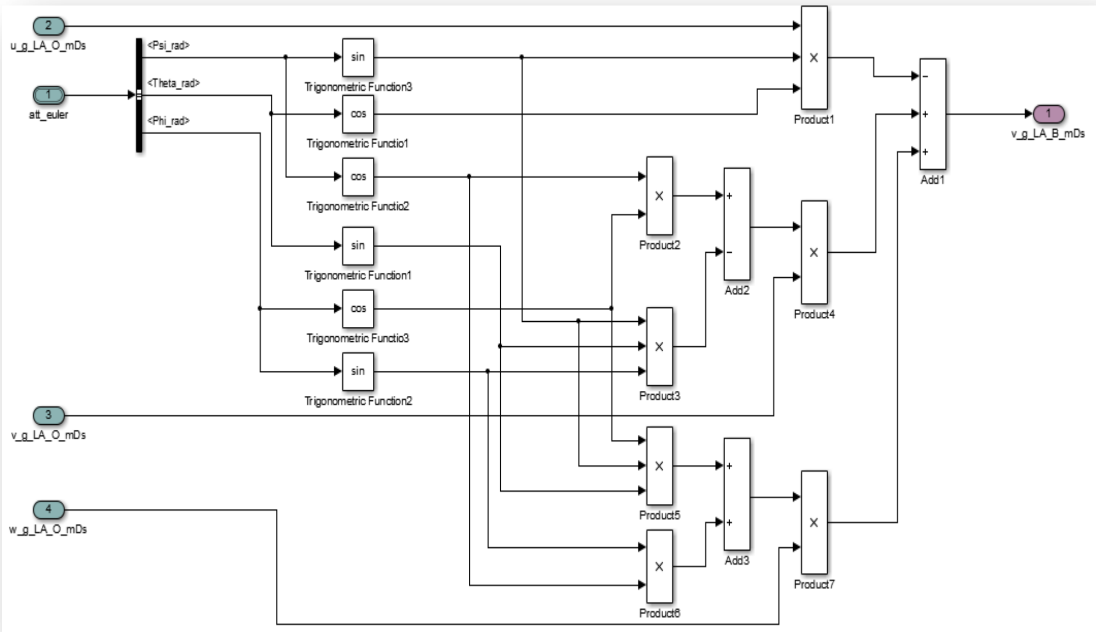
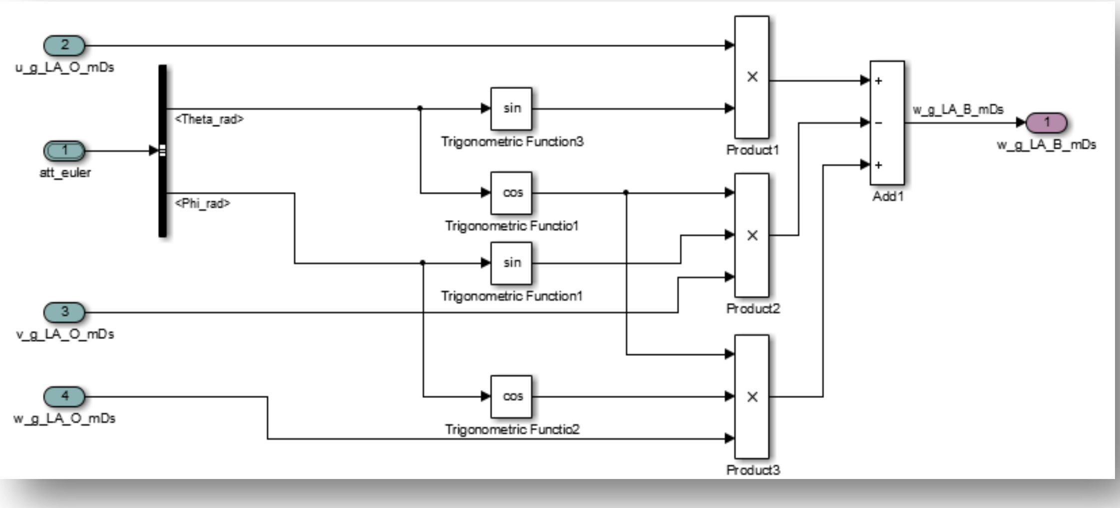


Figure 4-40 L5: (3,:) Rot_matrix_Vel_O_B



4.7 Verification Plan

4.7.1 Methods Used for Verification

4.7.1.1 Methods for Testing Functional Requirements

Requirement Name and ID	Description of Verification Method
R-FUN-ENV_TURB_01 Computation of Wind Velocity	Correct computation of Wind Velocity demonstrated in ENV_TURB-Nominal_TC1, comparison to dissimilar implementation in ENV_TURB-Nominal_TC2 (see section 4.7.2.1)
R-FUN-ENV_TURB_02 Computation of Wind Angular Rate	Correct computation of Wind Angular Rate demonstrated in ENV_TURB-Nominal_TC1, comparison to dissimilar implementation in ENV_TURB-Nominal_TC2 (see section 4.7.2.1)

Table 4-12 Methods for Testing Functional Requirements

4.7.1.2 Methods for Testing Implementation Requirements

Requirement Name and ID	Description of Verification Method
R-NUM-ENV_TURB Numeric Efficiency	Manual review of the implemented model. Records according to section 4.8.1.1
R-IOC-ENV_TURB Input / Output Interface Compliance to Parent System	Compliance to parent system will be verified at integration with parent system.
R-SGC-ENV_TURB Implementation Compliance to FSD Style Guides	Manual review of the implemented model. Records according to section 4.8.1.2
R-ISC-ENV_TURB Implementation Standards Compliance	Manual review of the implemented model. Records according to section 4.8.1.3.

Table 4-13 Methods for Testing Implementation Requirements

4.7.1.3 Methods for Testing Operational Requirements

Derived Standard Requirement Matrix:		
SIMULINK Modes	Simulink Offline Simulation	Demonstrated during test ENV_TURB-Nominal_TC1, ENV_TURB-Nominal_TC2 (see sections 4.7.2.1)
	Simulink Pseudo Real Time Simulation	Will be demonstrated on a higher level of the simulation model.
External Control for SIMULINK Execution	Bypassing of Non-Autonomous Elements	Not applicable as there are no non-autonomous elements present in the model.
	Single Point Execution	Demonstrated during test ENV_TURB-Nominal_TC1, ENV_TURB-Nominal_TC2 (see sections 4.7.2.1).
	Online Integration Freeze and Reset	Not applicable as there are no integrators present in the model.
	Workspace Initialization	Not applicable as there are no variables present to be initialized in the workspace.
	Runtime Parameter Tuning	Not applicable as there are no tunable parameters present in the model
RTW Code Generation	S-function	Generation of S-function demonstrated during test ENV_TURB-Operational_TC1 (see section 4.7.3.1).
Code Modes	Stand-alone Batch Simulation	Will be demonstrated on a higher level of the simulation model.
	Stand-alone Real Time Simulation	Will be demonstrated on a higher level of the simulation model.

Table 4-14 Methods for Testing Operational Requirements

4.7.2 Verification Plan for Functional Requirements

4.7.2.1 Nominal Testing Procedure

Test Name:	**Correct Calculation of Wind Velocity and Wind Angular Rate**
Test ID:	ENV_TURB-Nominal_TC1
Related Requirements:	R-FUN-ENV_TURB_01: Computation of Wind Velocity R-FUN-ENV_TURB_02: Computation of Wind Angular Rate
Verification Data:	See section 4.8.2.1

The implemented simulation model is excited with different input combinations. Afterwards, the output is checked by comparing the results.

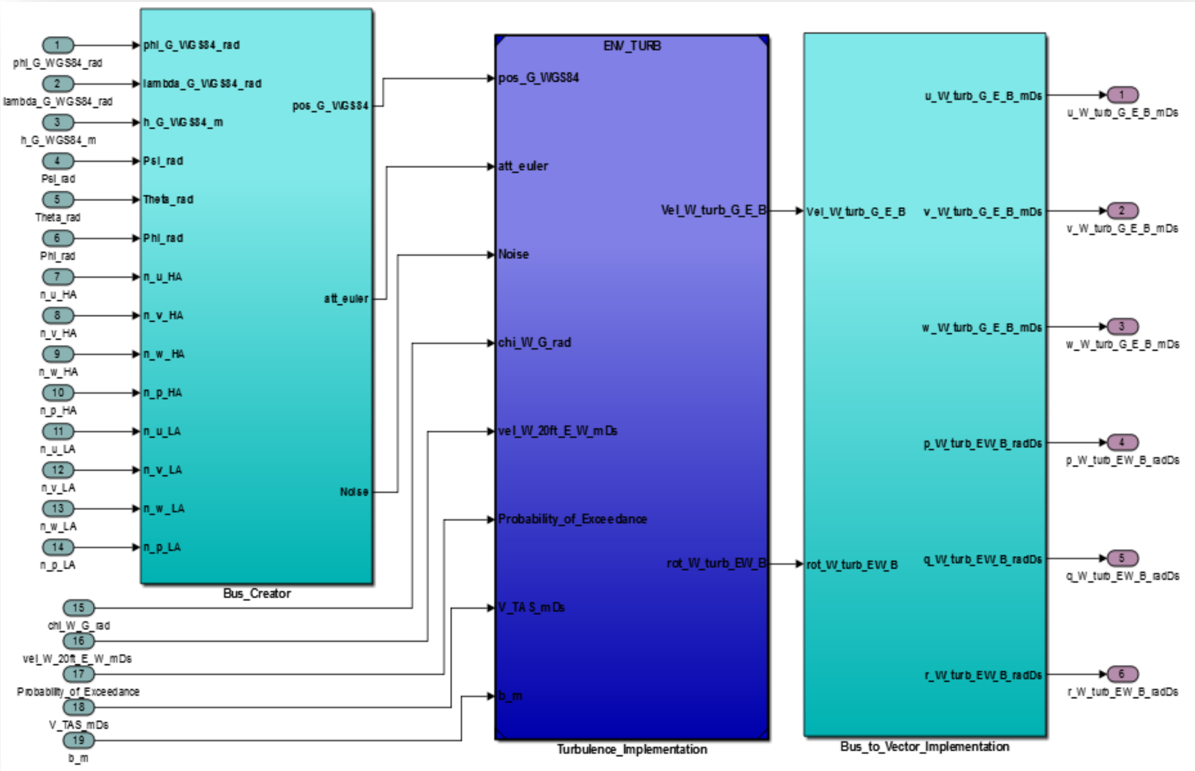


Figure 4-41 ENV_TURB-Nominal_TC1

The verification is carried out for an aircraft of small dimensions, placed in the range of low altitude in condition of Probability of Exceedance = 5.

Test Name:	**Equivalence of Implementation and Dissimilar Implementation**
Test ID:	ENV_TURB-Nominal_TC2
Related Requirements:	R-FUN-ENV_TURB_01: Computation of Wind Velocity R-FUN-ENV_TURB_02: Computation of Wind Angular Rate
Verification Data:	See section 4.8.2.1

The implemented model and a dissimilar implementation (Dryden Continuous Model in SIMULINK Toolbox) are both excited with the same input signals. Afterwards, the output is checked by comparing the results.

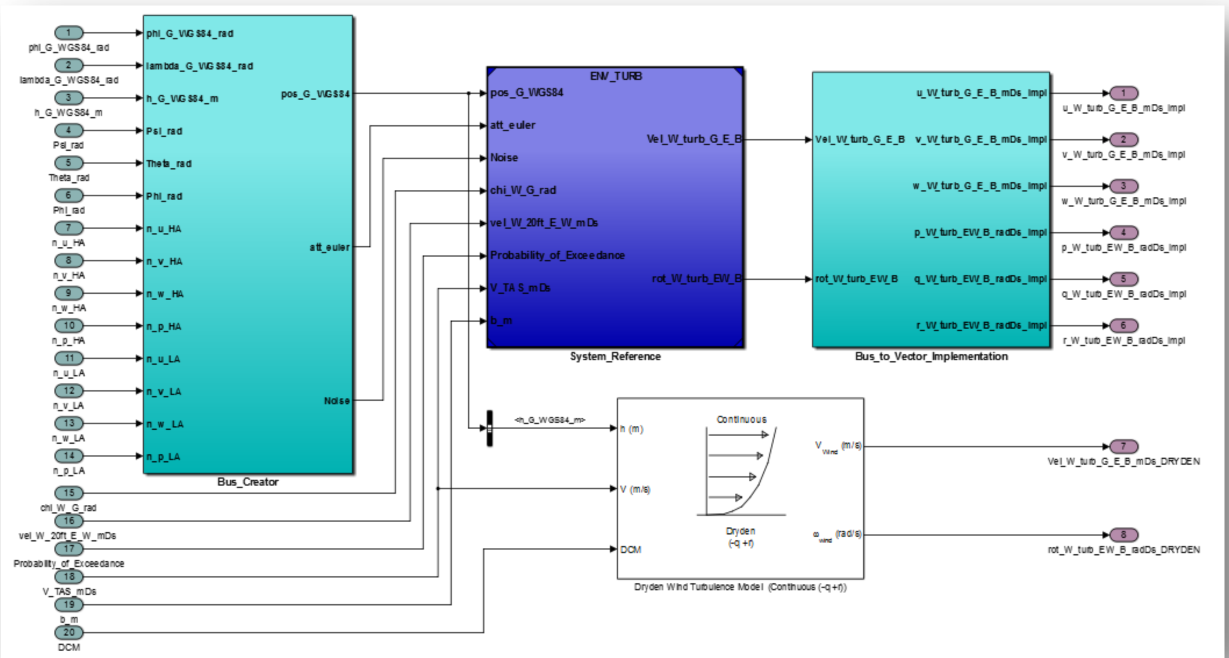


Figure 4-42 ENV_TURB-Nominal_TC2

The verification is carried out for an aircraft of small dimensions, placed in the range of low altitude in condition of Probability of Exceedance = 5.

4.7.3 Verification Plan for Operational Requirements

4.7.3.1 Code Generation and Equivalence Testing

Test Name:	**Code Generation and Equivalence Testing**
Test ID:	ENV_TURB-Operational_TC1
Related Requirements:	R-OPS-ENV_TURB: Operation Standard Requirements Matrix
Verification Data:	See section 4.8.3.1

The implemented simulation model running in normal mode and a S-function are excited with random inputs signals.

Afterwards the outputs are checked by comparing the results.

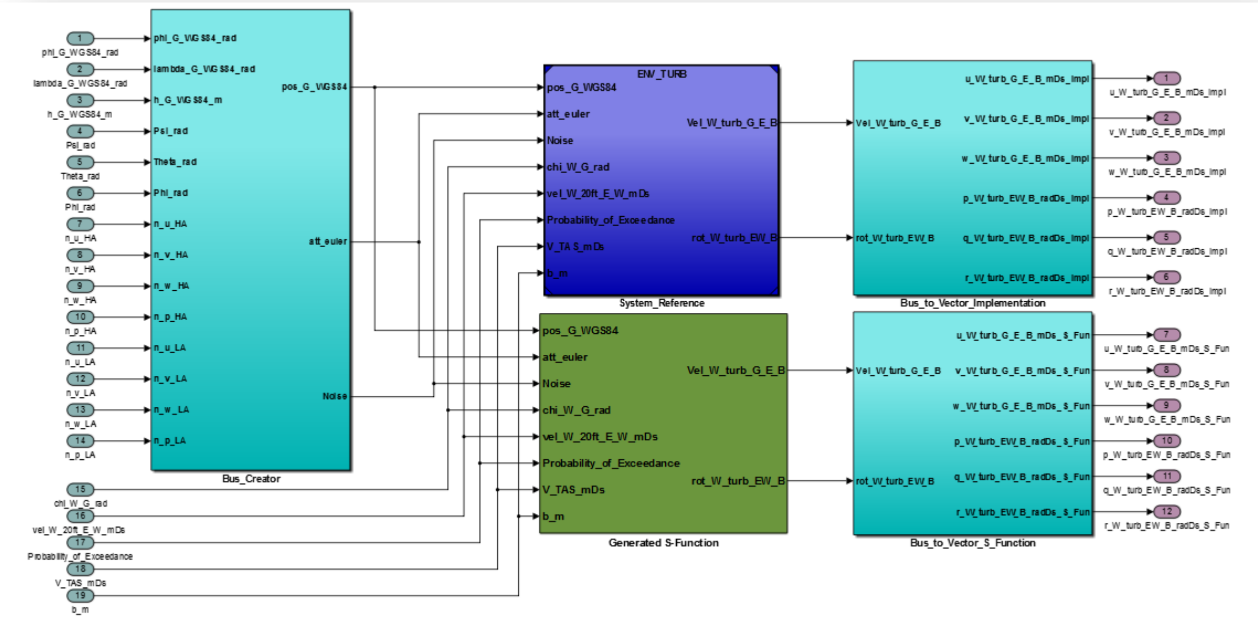


Figure 4-43 ENV_TURB-Operational_TC1

The verification is carried out for an aircraft of small dimensions, placed in the range of low altitude in condition of Probability of Exceedance = 5.

4.8 Verification Data

4.8.1 Verification of Implementation Requirements

4.8.1.1 Numeric Efficiency

Requirement Name **Numeric Efficiency**			Requirement ID R-NUM-ENV_TURB
Requirement is violated if the model contains one or more of the following items:			
Unused / Dead Code Branches Description	YES ☐	NO ☒	
Computational Redundancies Description	YES ☐	NO ☒	
Matrix Inversions Description	YES ☐	NO ☒	
Scalar Expansions of Vector/Matrix Math Description	YES ☐	NO ☒	
Circle Computations Description	YES ☐	NO ☒	
Inefficient Lookup Table Programming Description	YES ☐	NO ☒	
Algebraic Loops Description	YES ☐	NO ☒	
Numeric Efficiency met?	**YES ☒**	**NO ☐**	

Table 4-15 R-NUM-ENV_TURB

4.8.1.2 Implementation Compliance to FSD Style Guidelines

Requirement Name **Implementation Compliance to FSD Style Guidelines**	Requirement ID R-SGC-ENV_TURB	
Requirement is violated if the model contains other blocks than the specified ones.		
Non-Specified Blocks within the Model? Description	YES ☐	NO ☒
Style Guide Compliance met?	**YES ☒**	**NO ☐**

Table 4-16 R-SGC-ENV_TURB

4.8.1.3 Implementation Standards Compliance

Requirement Name **Implementation Standards Compliance**	Requirement ID R-ISC-ENV_TURB	
The coded algorithm must not contain any programming technique listed below unless detailed justification substantiates indispensability.		
Discrete Switches *	YES ☒	NO ☐
Memory Blocks	YES ☐	NO ☒
Time Delays	YES ☐	NO ☒
Time Dependent / Non-Autonomous Elements	YES ☐	NO ☒
In-lined Integrations	YES ☐	NO ☒
Hysteresis and Quantized Elements	YES ☐	NO ☒
Stochastic / Random Elements	YES ☐	NO ☒
Normal atan Blocks	YES ☐	NO ☒
Operations with Sign Loss	YES ☐	NO ☒
Value Flipping and Range Limiting	YES ☐	NO ☒
Math Function out of Range	YES ☐	NO ☒
Division by Zero	YES ☐	NO ☒
Finite State Transition	YES ☐	NO ☒
Implementation Standards Compliance met?	**YES ☒**	**NO ☐**

Table 4-17 R-ISC-ENV_TURB

* The switches in the system do not affect the correct operation.

4.8.2 Verification of Functional Requirements

4.8.2.1 Results for Nominal Testing

Test Name:	**Correct Calculation of Wind Velocity end Wind Angular Rate**
Test ID:	ENV_TURB-Nominal_TC1
Verification Plan:	See section 4.7.2.1

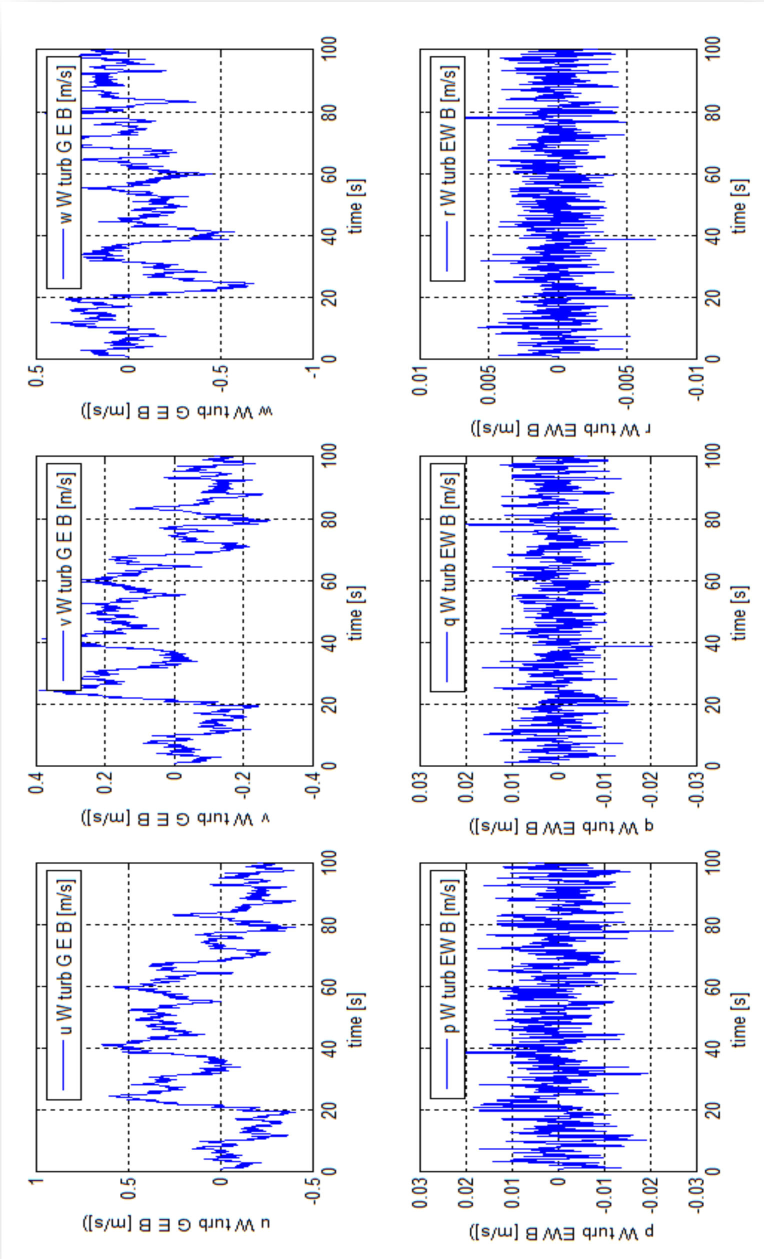


Figure 4-44 Wind Velocity and Wind Angular Rate

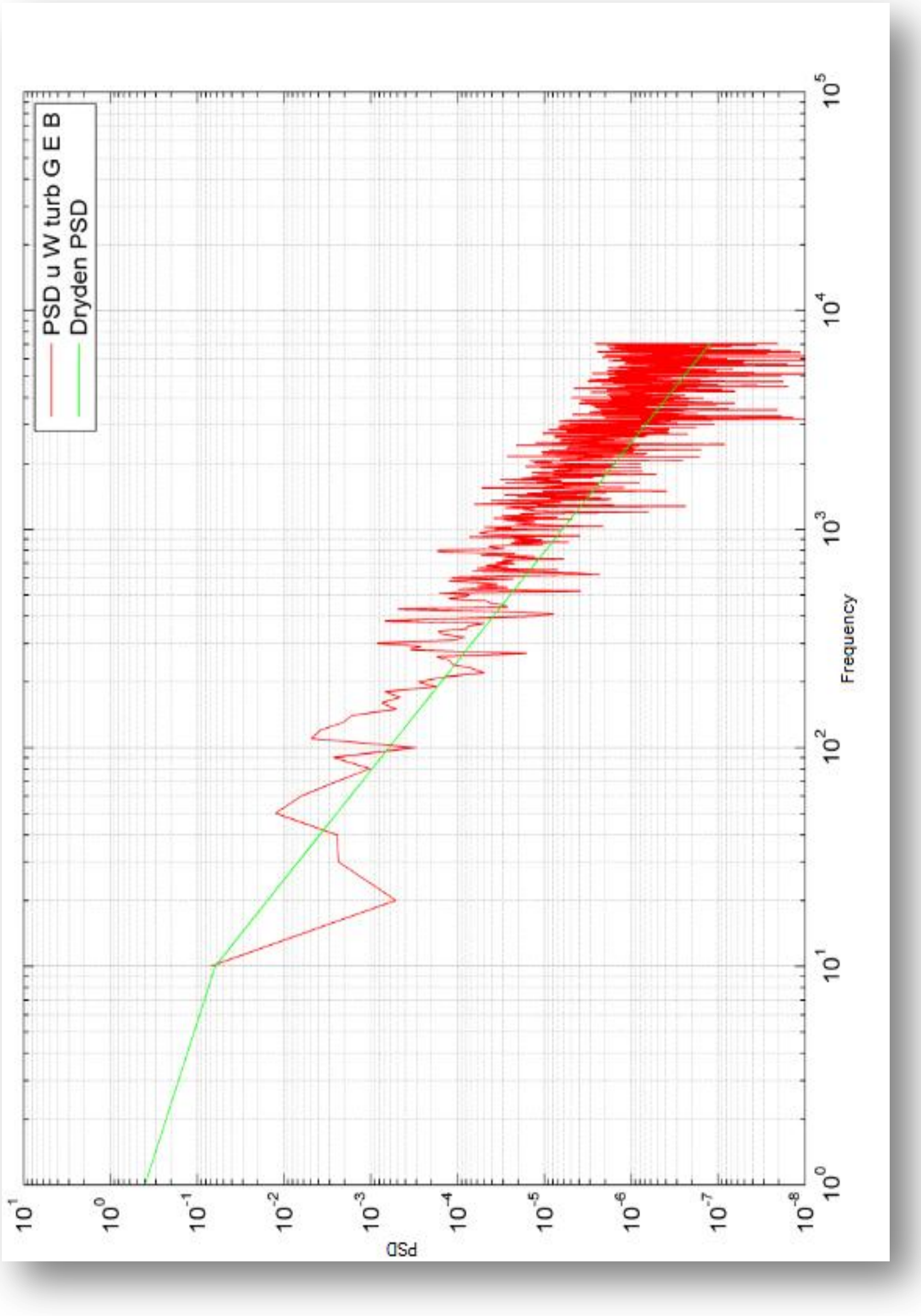


Figure 4-45 Power Spectral Density Function relative to u_{turb}

Correct Nominal Behavior?	YES ☒	NO ☐

Test Name:	**Equivalence of Implementation and Dissimilar Implementation**
Test ID:	ENV_TURB-Nominal_TC2
Verification Plan:	See section 4.7.2.1

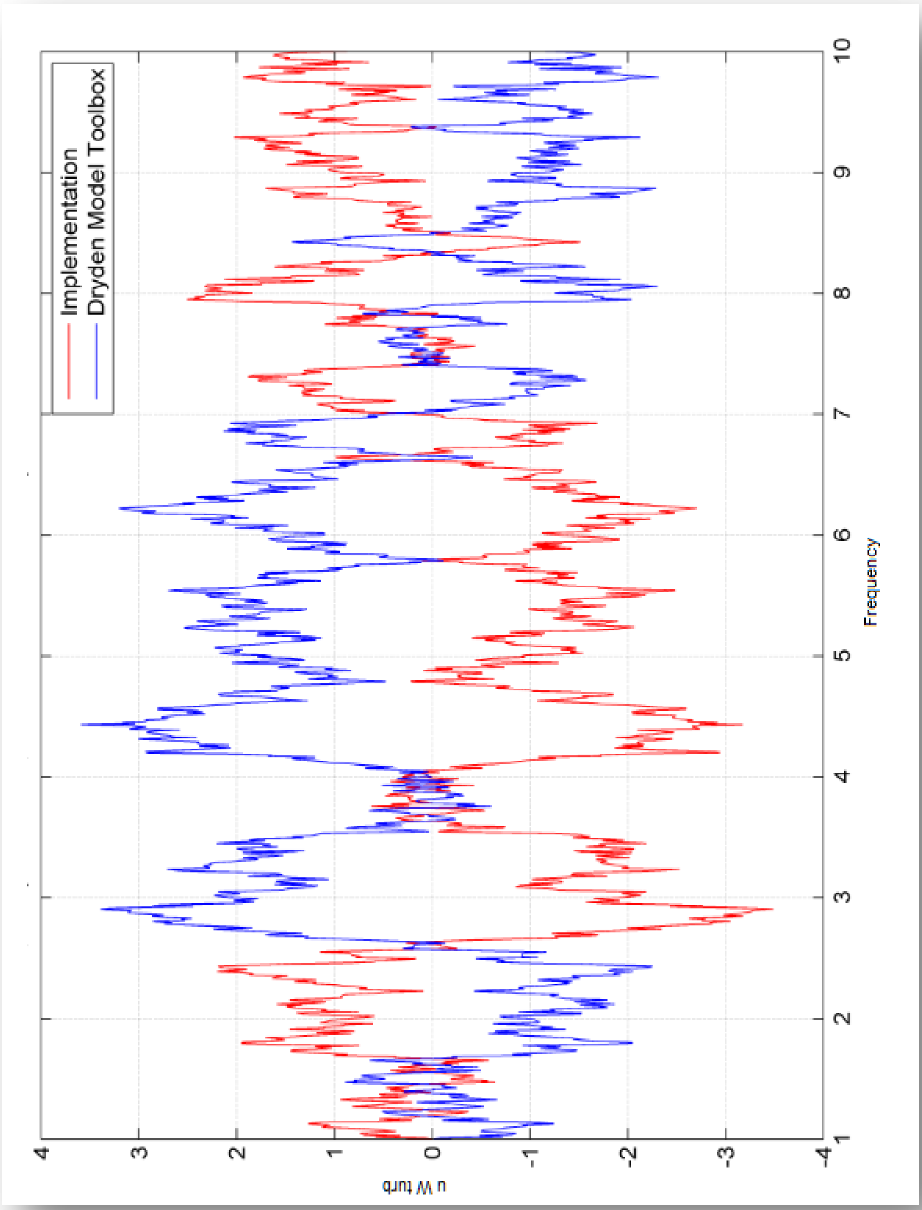


Figure 4-46 Wind Velocity in Implementation and Dissimilar Implementation: different sign conventions

Course of Wind Velocity realistic?	**YES ☒**	**NO ☐**

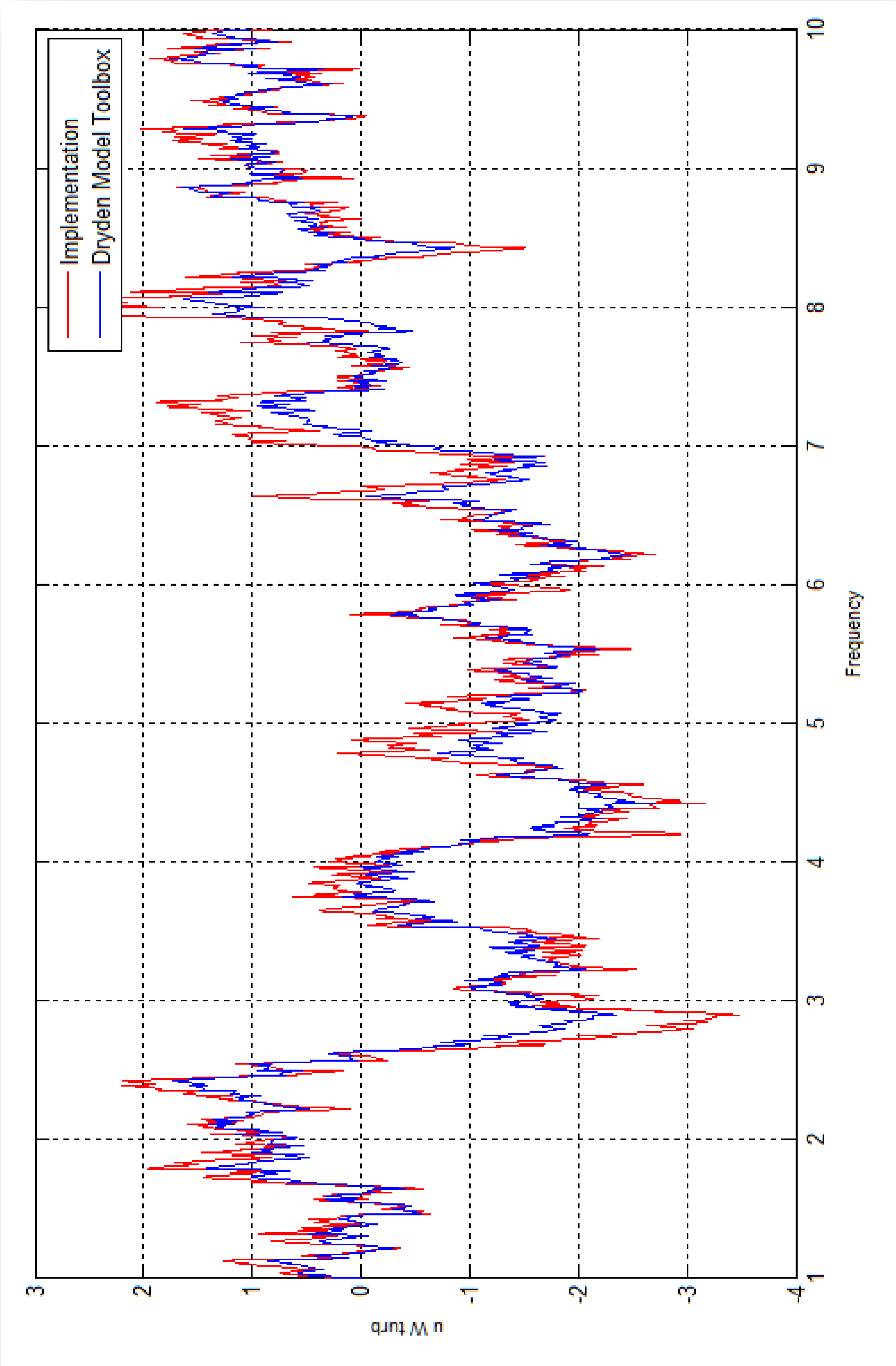


Figure 4-47 Wind Velocity in Implementation and Dissimilar Implementation: same sign conventions

Course of Wind Velocity realistic?	**YES ☒**	**NO ☐**

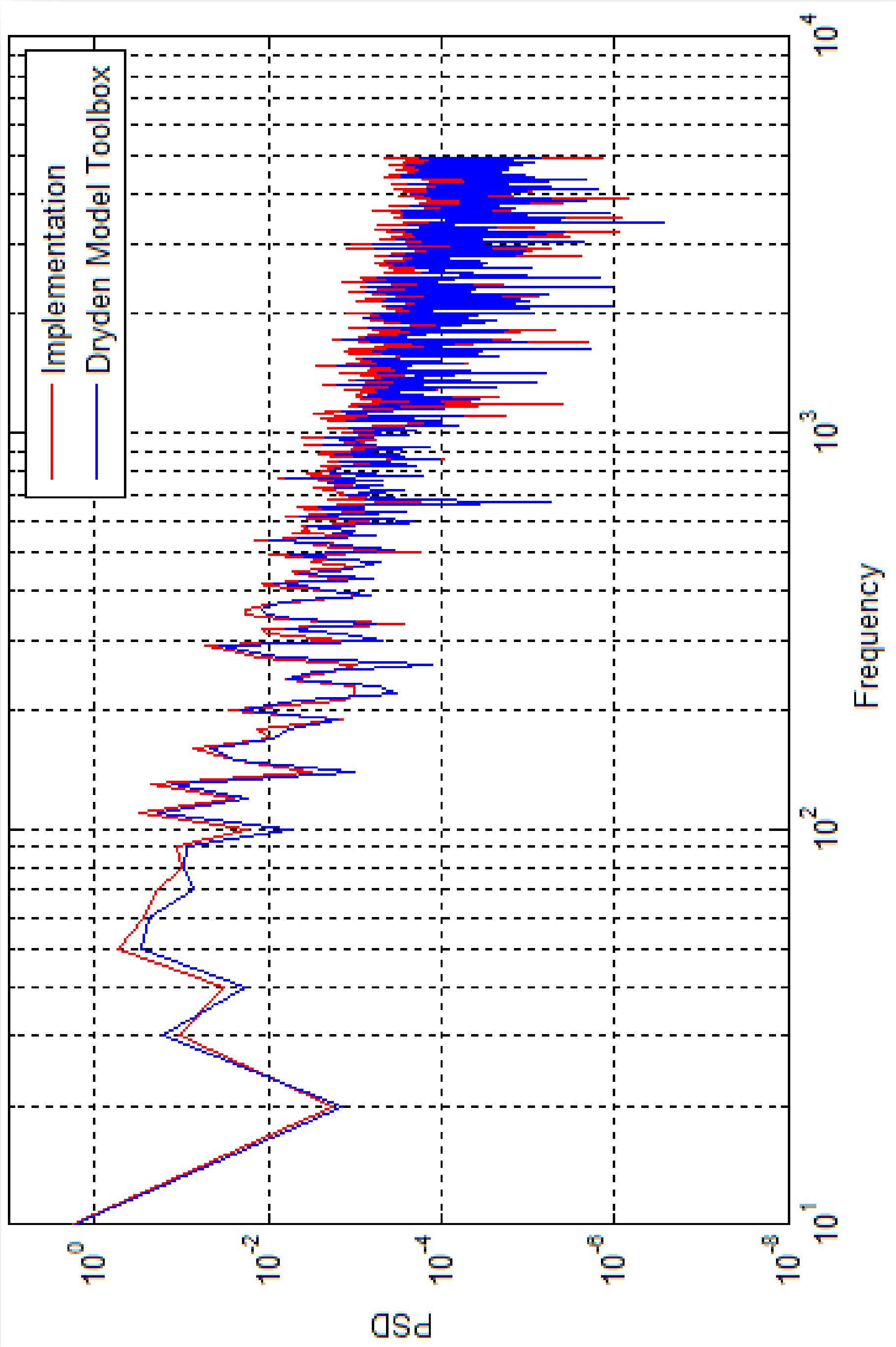


Figure 4-48 PSD in Implementation and Dissimilar Implementation

Course of PSD realistic?	**YES ☒**	**NO ☐**

4.8.3 Verification of Operational Requirements

4.8.3.1 Results for Code Generation and Equivalence Testing

Test Name:	**Code Generation and Equivalence Testing**
Test ID:	ENV_TURB-Operational_TC1
Verification Plan:	See section 4.7.3.1

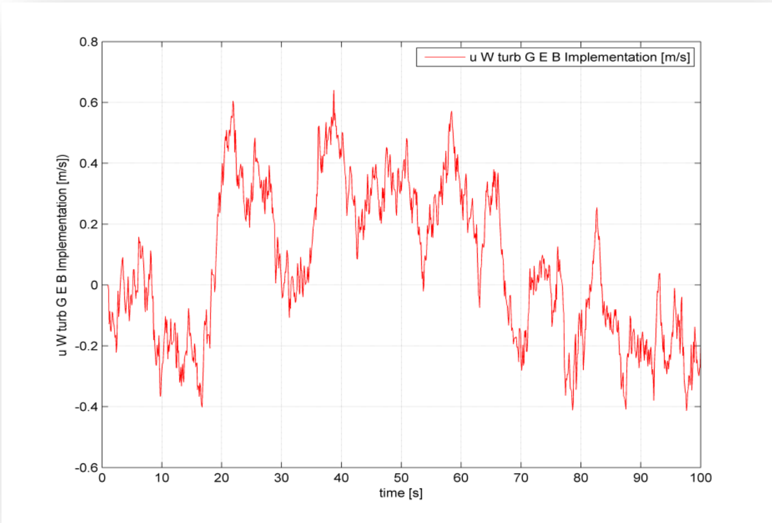


Figure 4-49 u_{turb} Implementation

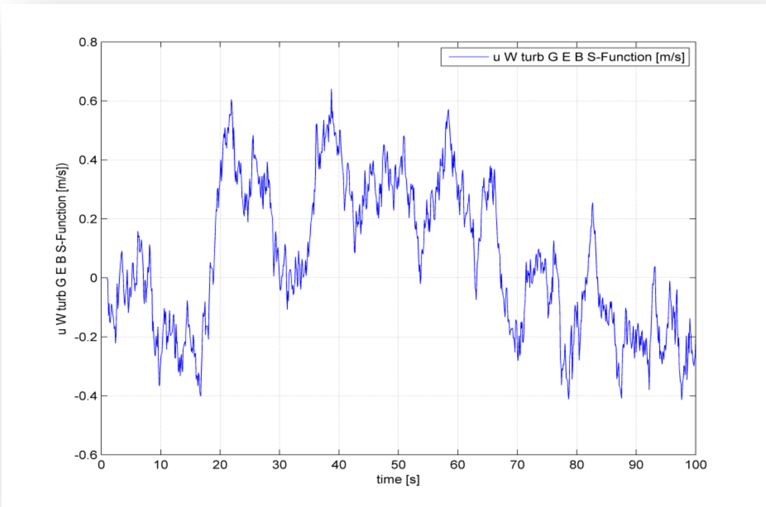


Figure 4-50 u_{turb} S-Function

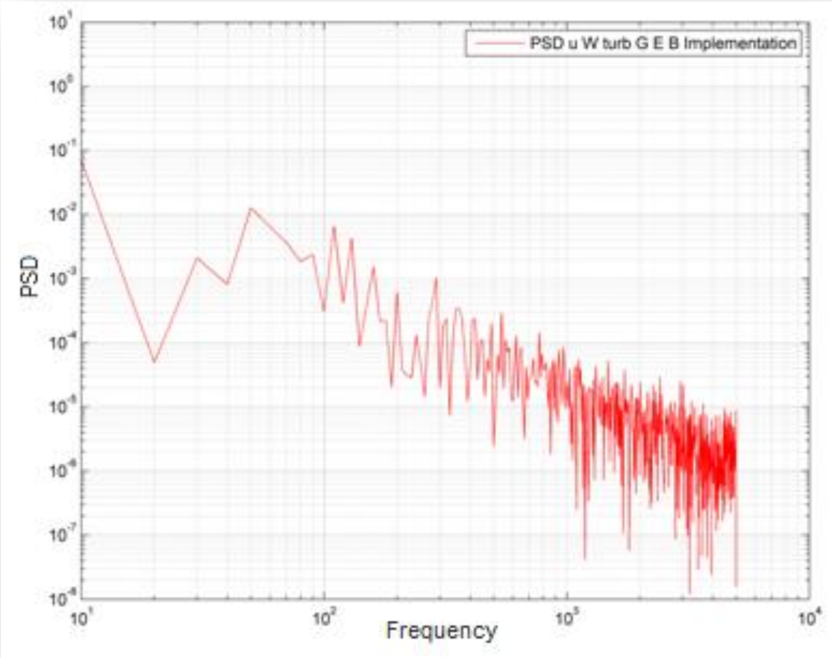


Figure 4-51 PSD Implementation

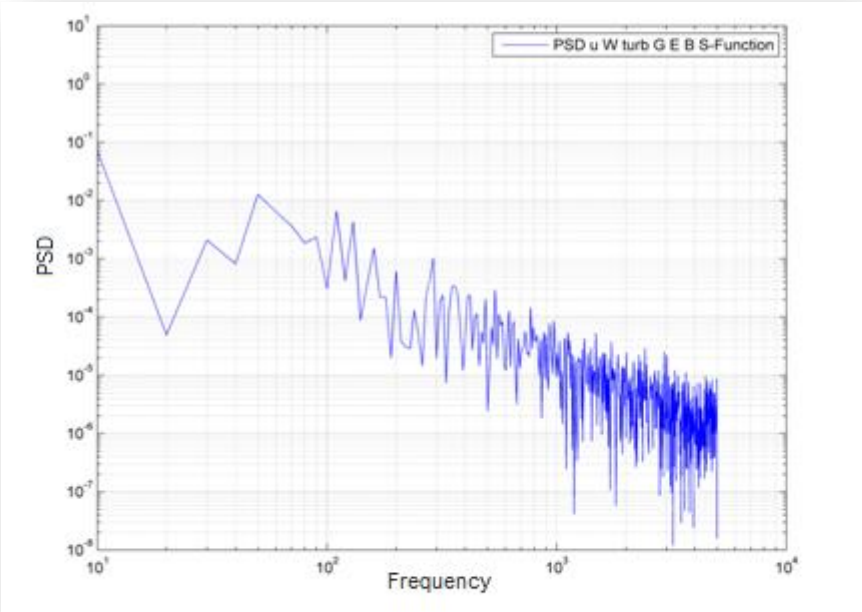


Figure 4-52 PSD S-Function

Compilation of S-function successful?	YES ☒	NO ☐
Compilation of standalone executable successful?	YES ☒	NO ☐
Description of Compilation Errors		
Warnings during Compilation?	YES ☐	NO ☒
Description of Compilation Warnings		

Run-Time Errors or Warnings?	YES ☐	NO ☒
Description of Error / Warning Messages		

Code Generation successful and Coded Version equivalent to Simulation Model?	**YES ☒**	**NO ☐**

5 Gust Model

5.1 Introduction

The wind gust is the maximum wind speed measured during a specified time period.

In other words, the gust is defined as a sudden short increase in the wind velocity.

According to the Military Specification [3], the description used for the gust is the Discrete Wind Gust Model, in which the wind gust is described with the standard "1-cosine" shape (see Figure 5-1).

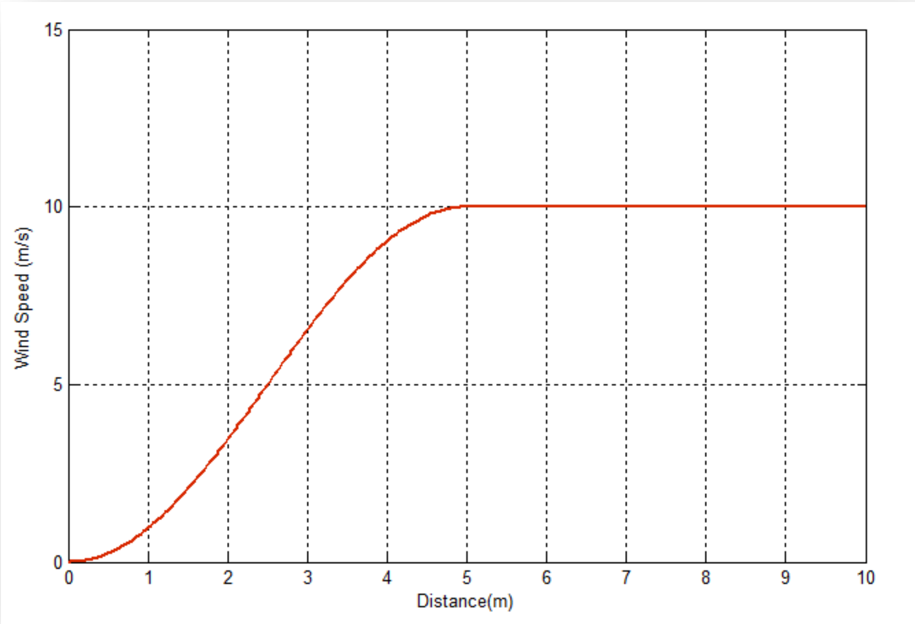


Figure 5-1 Wind Gust "1-cosine" shape

The model used, allows to obtain the three gust-velocity components.

By derivation is then possible to calculate the corresponding three angular components.

5.2 Description of the Functional and Operational Intent

The system is intended to compute the wind velocity of the center of gravity G defined with respect to the Earth Centered Fixed Frame (*E*) components written in Body Frame (*B*) : $\left(\vec{v}_{W_{gust}}^{G}\right)_{B}^{E}$.

In order to take into account the beginning and the end of the gust, it is chosen to use a gust with a symmetrical shape.

The final model will be used as part of a simulation model, which shall be incorporated into the simulation framework of a flight simulation device. Therefore it is necessary that all subsystems are compliant to code generation requirements.

5.3 Requirements

5.3.1 Functional Requirements

Requirement Name	Requirement ID
Computation of Wind Gust Velocity	R-FUN-ENV_GUST
Derived from	
Purpose of the system.	
Requirement Definition	
The Wind Gust Velocity shall be computed.	

Table 5-1 R-FUN-ENV_GUST

5.3.2 Operational Requirements

Requirement Name	Requirement ID
Incorporation into Flight Simulator Simulation Model	R-OPS-ENV_GUST
Derived from	
Usage intents	
Requirement Definition	
The model shall be incorporated as a child system into the simulation model of an (atmospheric) flight simulation device. Therefore it is necessary that all components support code generation.	

Table 5-2 R-OPS-ENV_GUST

5.3.3 Implementation Requirements

Requirement Name **Numeric Efficiency**	Requirement ID R-NUM-ENV_GUST
Derived from Global Implementation Guidelines	
Requirement Definition The coded algorithm must not contain any of the numerical inefficient programming techniques listed below unless detailed justification substantiates indispensability.	
Programming Techniques to be Avoided:	
Unused / Dead Code Branches	
Computational Redundancies	
Matrix Inversions	
Scalar Expansion of Vector / Matrix Math	
Circle Computations	
Inefficient Lookup Table Programming	
Algebraic Loops	

Table 5-3 R-NUM-ENV_GUST

Requirement Name **Input / Output Interface Compliance to Parent System and Child Systems**	Requirement ID R-IOC-ENV_GUST
Derived from Global Implementation Guidelines	
Requirement Definition Input and Output interface must comply with parent system.	
Compliance required to:	
Global bus object definitions	
I/O signal name matching to parent system	
I/O signal unit matching to parent system	
I/O signal data type matching to parent system	
I/O signal data range compatibility matching to parent system	

Table 5-4 R-IOC-ENV_GUST

Requirement Name	Requirement ID
Implementation Compliance to FSD Style Guidelines	R-SGC-ENV_GUST
Derived from	
Global Implementation Guidelines	
Requirement Definition	
Only a subset of SIMULINK blocks is allowed to be implemented.	
Allowed Libraries and Toolboxes:	
FSD Compliant Base	
Use of other Libraries and Toolboxes is Forbidden!	

Table 5-5 R-SGC-ENV_GUST

Requirement Name	Requirement ID
Implementation Standards Compliance	R-ISC-ENV_GUST
Derived from	
Global Implementation Guidelines	
Requirement Definition	
The coded algorithm must not contain any programming technique listed below unless detailed justification substantiates indispensability.	
Forbidden Programming Techniques:	
Discrete Switches *	
Memory Blocks	
Time Delays	
Time Dependent / Non-Autonomous Elements	
In-lined Integrations	
Hysteresis and Quantized Elements	
Stochastic / Random Elements	
Normal atan Blocks	
Operations with Sign Loss	
Value Flipping and Range Limiting	
Math Function out of Range	
Division by Zero	
Finite State Transition	

Table 5-6 R-ISC-ENV_GUST

* The switches in the system do not affect the correct operation.

5.4 Function Specification

5.4.1 Algorithm Abstract

The system is intended to compute Wind Gust Velocity of the center of gravity G defined with respect to the Earth Centered Fixed frame, components written in Body frame $\left(\vec{v}^{G}_{W_{gust}}\right)^{E}_{B}$.

The model used is the standard "1-cosine" shape.

To take into account the end of the gust, it is chosen to use a symmetric shape.

In addition, it is possible to enable or disable the gust in each of three directions using some input signals.

5.4.2 Modeling Assumptions, Scope of Validity & Limitations

- Wind Discrete Gust Model:
  The model used to describe the wind gust is the standard "1-cosine" shape;
- Wind Discrete Gust Model symmetric:
  It is assumed, that the gust ends in a symmetrical manner.

5.4.3 Detailed Algorithm Description

5.4.3.1 Wind Discrete Gust Model

According to Military Specification [3], the model used to describe the wing gust is the standard "1-cosine" shape:

$$\left(v^{G}_{W_{gust}}\right)^{E}_{B} = \begin{cases} v = 0, & x < 0 \\ v = \dfrac{v_m}{2}\left(1 - cos\left(\dfrac{\pi \cdot x}{d_m}\right)\right), & 0 \le x \le d_m \\ v = v_m, & x > d_m \end{cases} \qquad 5\text{-}1$$

where v_m is the gust amplitude [m/s], d_m is the gust length [m], x is the distance traveled [m] and $(v^{G}_{W_{gust}})^{E}_{B}$ is the resultant wind velocity in the body axis frame.

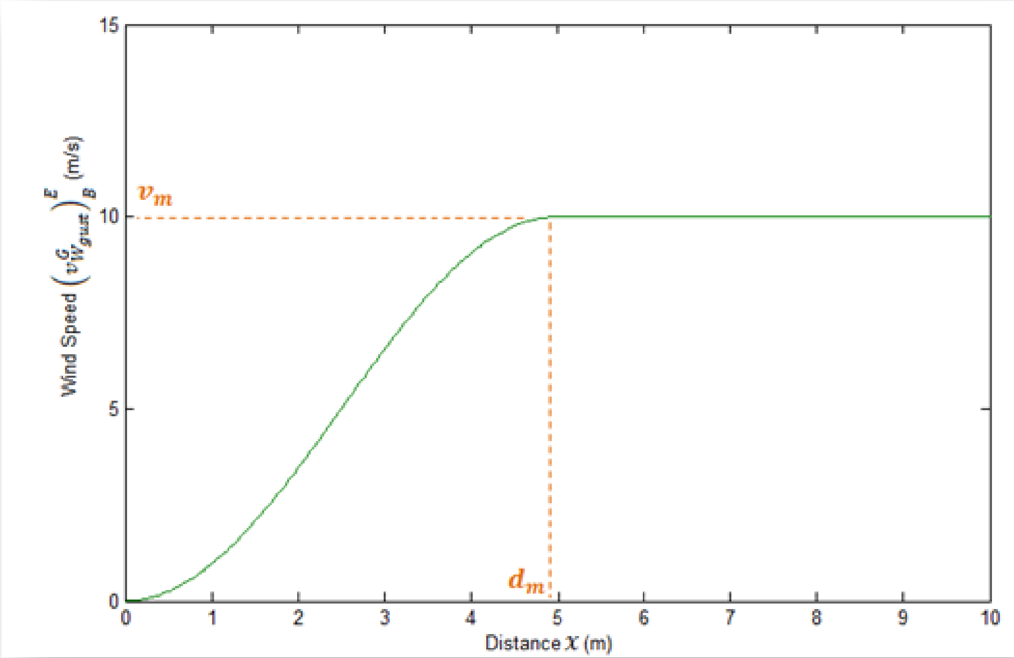


Figure 5-2 " 1 - cosine" shape

The gust amplitude may be positive or negative while the gust length must be positive.

5.4.3.2 Discrete Wind Gust Model symmetric

To take into account the beginning and the end of the wind gust, it is chosen to use a symmetric shape of the wind gust.

If after a gust length d_m, the gust velocity becomes constant, it was decided to decrease the velocity after two times the gust length d_m, with a symmetrical shape with respect at the beginning, up to zero for three times the gust length d_m :

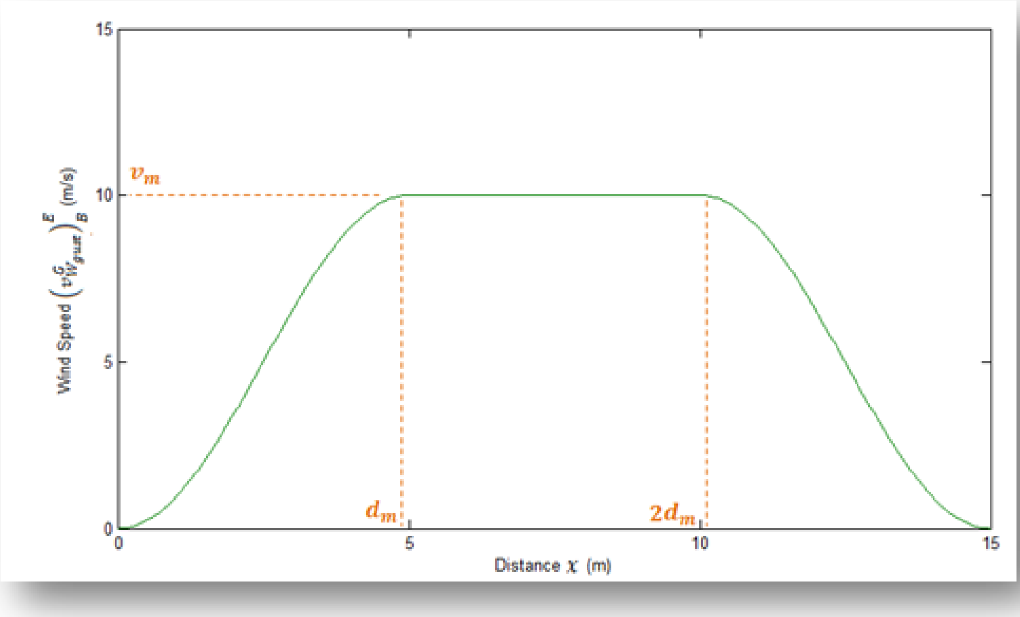


Figure 5-3 Symmetric shape

The model becomes:

$$\left(v_{W_{gust}}^{G}\right)_{B}^{E} = \begin{cases} v = 0, & x < 0 \\ v = \dfrac{v_m}{2}\left(1 - cos\left(\dfrac{\pi \cdot x}{d_m}\right)\right), & 0 \leq x \leq d_m \\ v = v_m, & d_m < x < 2d_m \\ v = \dfrac{v_m}{2}\left(1 + cos\left(\dfrac{\pi \cdot x}{d_m}\right)\right), & 2d_m \leq x \leq 3d_m \\ v = 0, & x > 3d_m \end{cases} \qquad \text{5-2}$$

5.4.3.3 Distance traveled

The distances traveled (x, y *and* z) are obtained as integration of the airspeed.

5.4.3.4 Input signals to start the gust

In order to decide the moment in which activate the gust, some input signals were added:

- flg_start_gust_x
- flg_start_gust_y
- flg_start_gust_z

Each of these signals can only take on values of zero (gust off) or one (gust on) and enable or disable the gust in the corresponding direction.

5.4.3.5 Algorithm for Implementation

In the implemented system, the equations 5-2 are used.

5.5 Architecture Specification

5.5.1 Parent / Child Systems

5.5.1.1 Parent System

The system will be embedded into the system “Atmosphere”, which will be embedded into the parent system "Environment", whose purpose it is to simulate all processes regarding the environment of the aircraft (atmosphere, terrain model, earth model, etc.).

5.5.1.2 Child Systems

This system does not contain any child systems.

5.5.2 Signal Definitions

5.5.2.1 Inputs

Inputs							
Symbol	**Name**	**Size**	**Components**	**Data Type**	**Min**	**Max**	**Description**
V_{air}	Airspeed_mDs	[1 1]	-	double	-Inf	Inf	Velocity of the Air.
v_x	Gust_Amplitude_u_mDs	[1 1]	-	double	-Inf	Inf	Gust Amplitude in direction of x
v_y	Gust_Amplitude_v_mDs	[1 1]	-	double	-Inf	Inf	Gust Amplitude in direction of y
v_z	Gust_Amplitude_w_mDs	[1 1]	-	double	-Inf	Inf	Gust Amplitude in direction of z
d_x	Gust_Length_x_m	[1 1]	-	double	0	Inf	Gust Length in direction of x
d_y	Gust_Length_y_m	[1 1]	-	double	0	Inf	Gust Length in direction of y
d_z	Gust_Length_z_m	[1 1]	-	double	0	Inf	Gust Length in direction of z
flg_x	flg_start_gust_x	[1 1]	-	double	0	1	Flag to start the gust component x
flg_y	flg_start_gust_y	[1 1]	-	double	0	1	Flag to start the gust component y
flg_z	flg_start_gust_z	[1 1]	-	double	0	1	Flag to start the gust component z

Table 5-7 Inputs

5.5.2.2 Outputs

Outputs							
Symbol	**Name**	**Size**	**Components**	**Data Type**	**Min**	**Max**	**Description**
$\left(\vec{v}^{G}_{W_{gust}}\right)^{E}_{B}$	Vel_W_gust_G_E_B_mDs	[3 1]	u_W_gust_G_E_B_mDs v_W_gust_G_E_B_mDs w_W_gust_G_E_B_mDs	double	-Inf -Inf -Inf	Inf Inf Inf	Wind Velocity of the Center of Gravity *G* defined with respect to the Earth Centered Fixed Frame *(E)*, components written in Body Frame *(B)*.

Table 5-8 Outputs

5.5.3 Bus Structure

Inputs

L	Bus Name	Elements	Element Types
0	*Gust_Amplitude_Bus*	Gust_Amplitude_u_mDs Gust_Amplitude_v_mDs Gust_Amplitude_w_mDs	double double double
0	*Gust_Length_Bus*	Gust_Length_x_m Gust_Length_y_m Gust_Length_z_m	double double double
0	*flg_start_gust_Bus*	flg_start_gust_x flg_start_gust_y flg_start_gust_z	double double double

Table 5-9 Inputs Bus Structure

Outputs

L	Bus Name	Elements	Element Types
0	*Vel_W_gust_G_E_B_Bus*	u_W_gust_G_E_B_mDs v_W_gust_G_E_B_mDs w_W_gust_G_E_B_mDs	double double double

Table 5-10 Outputs Bus Structure

5.6 Structural Layout

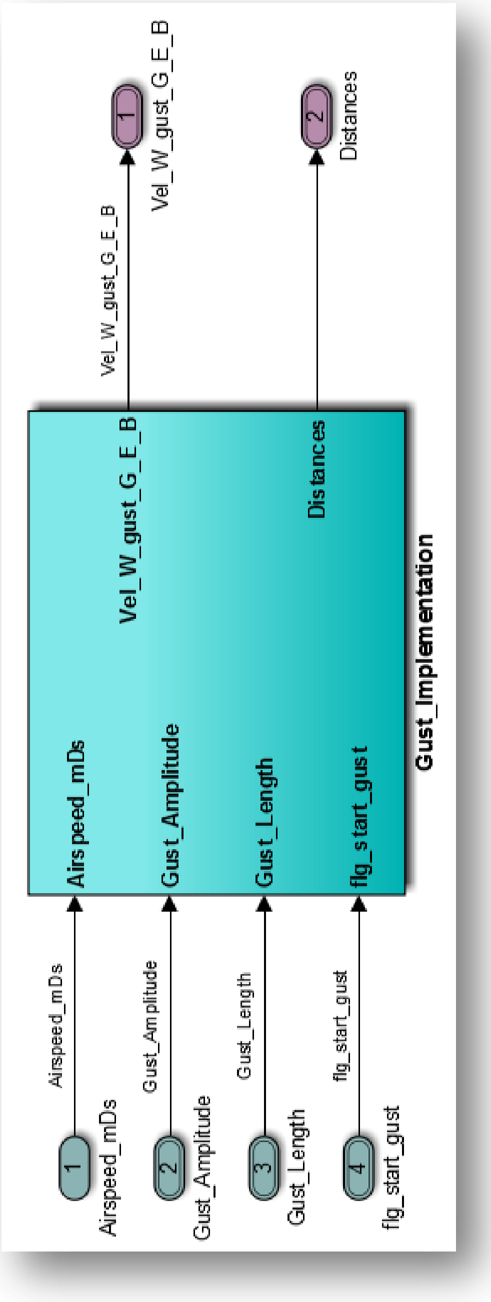


Figure 5-4 L0: ENV_GUST

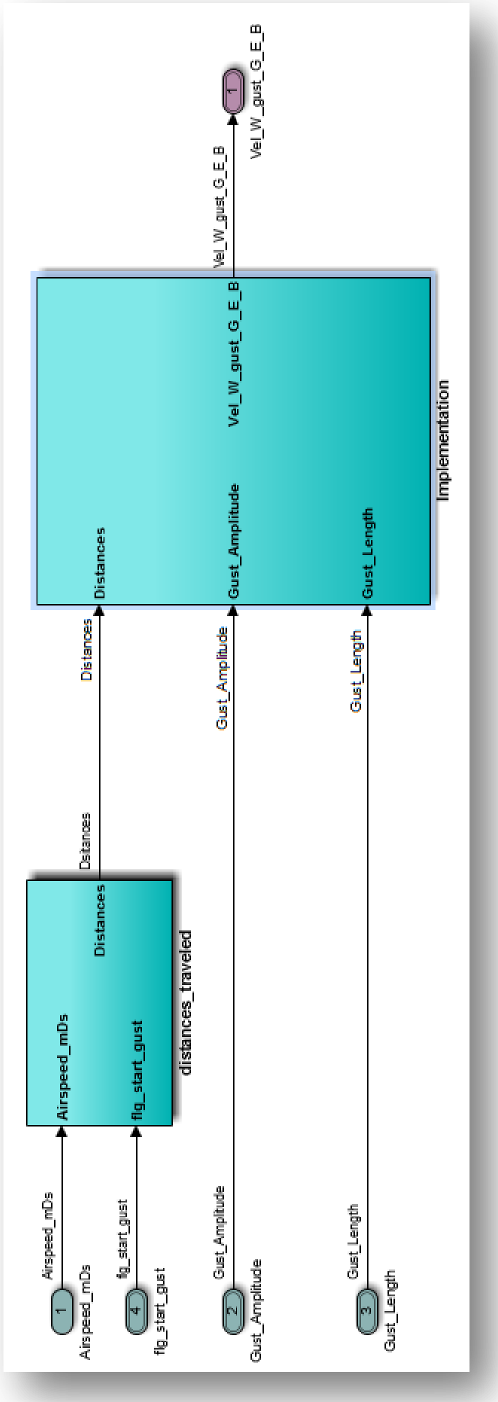


Figure 5-5 L1: Gust_Implementation

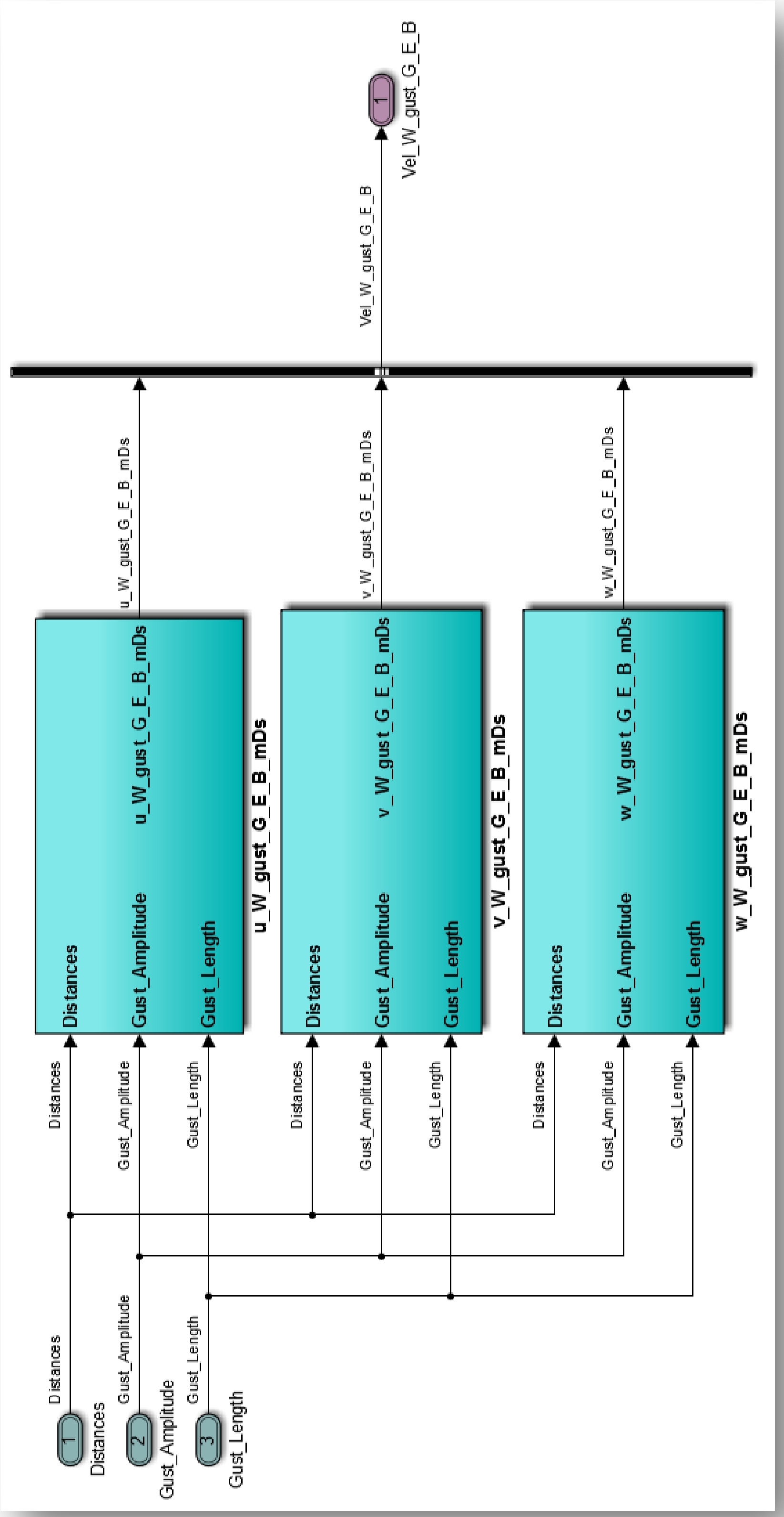


Figure 5-6 L2: Implementation

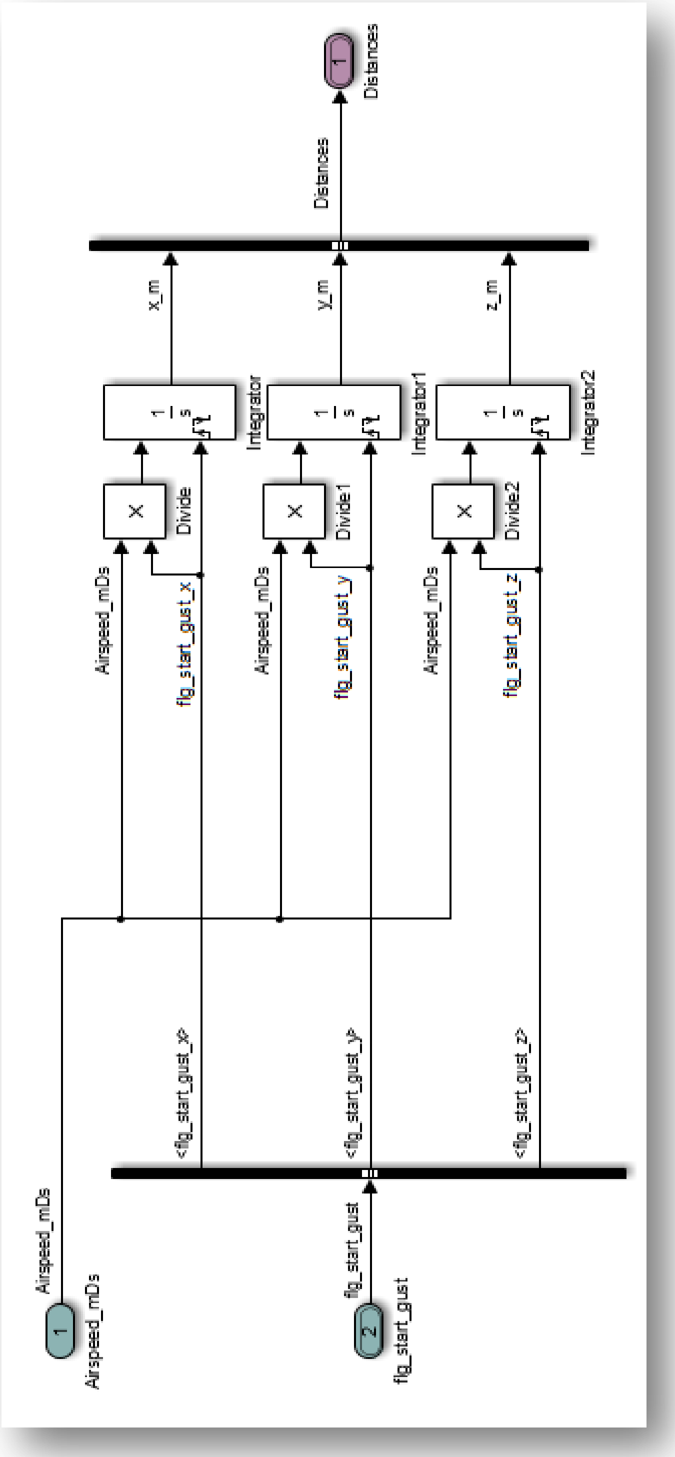


Figure 5-7 L2: distances_traveled

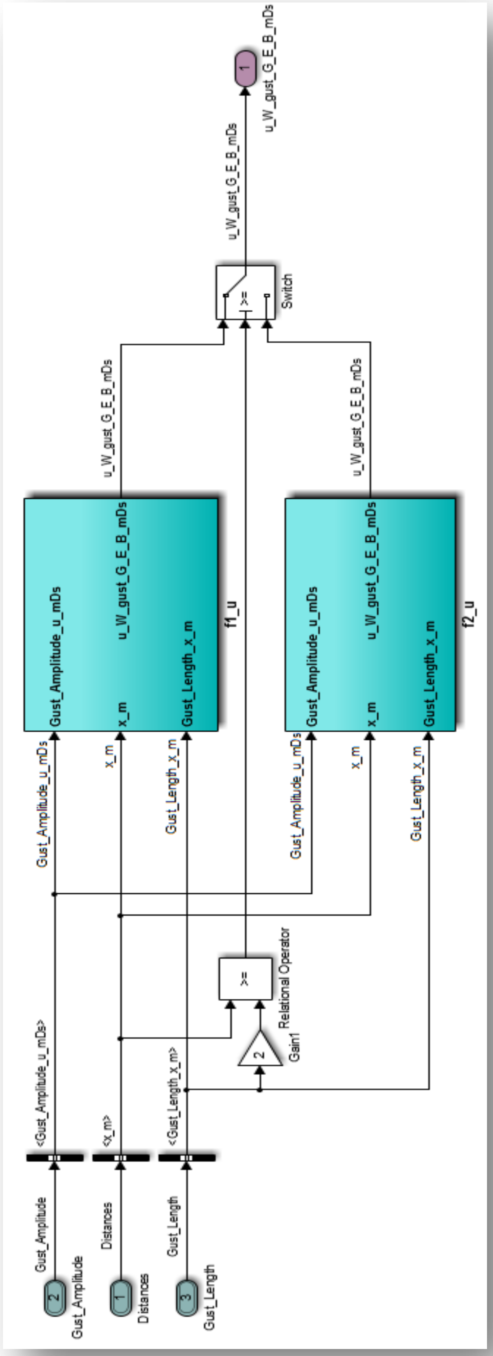


Figure 5-8 L3: u_W_gust_G_E_B_mDs

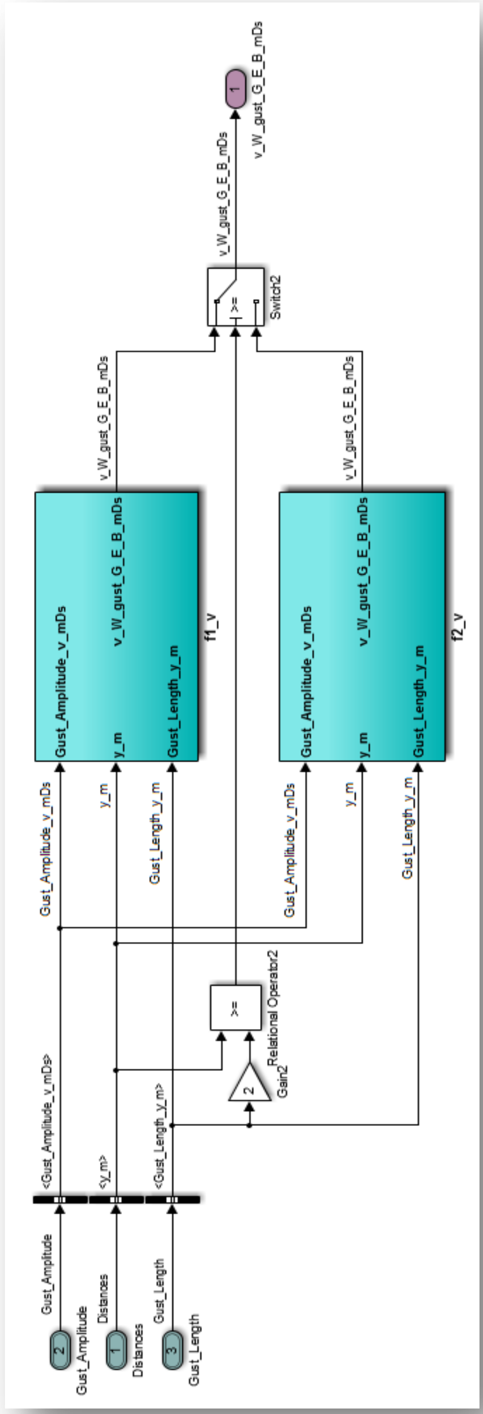


Figure 5-9 L3: v_W_gust_G_E_B_mDs

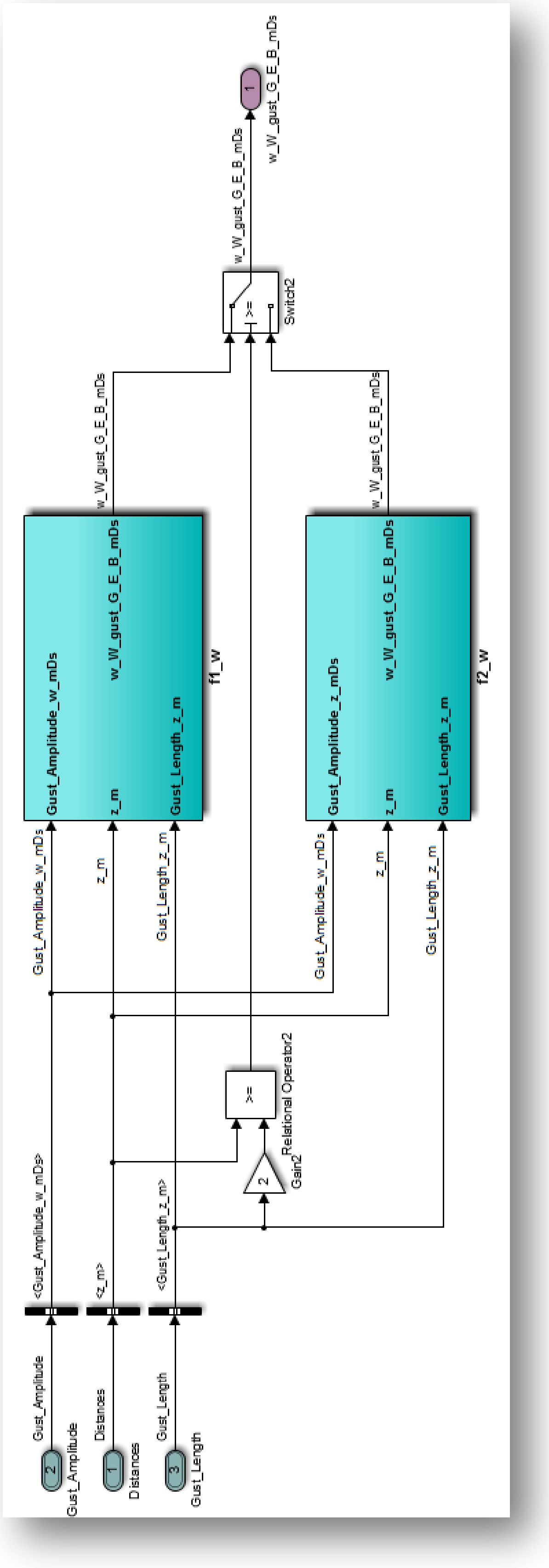


Figure 5-10 L3: w_W_gust_G_E_B_mDs

Figure 5-11 L4: f1_u

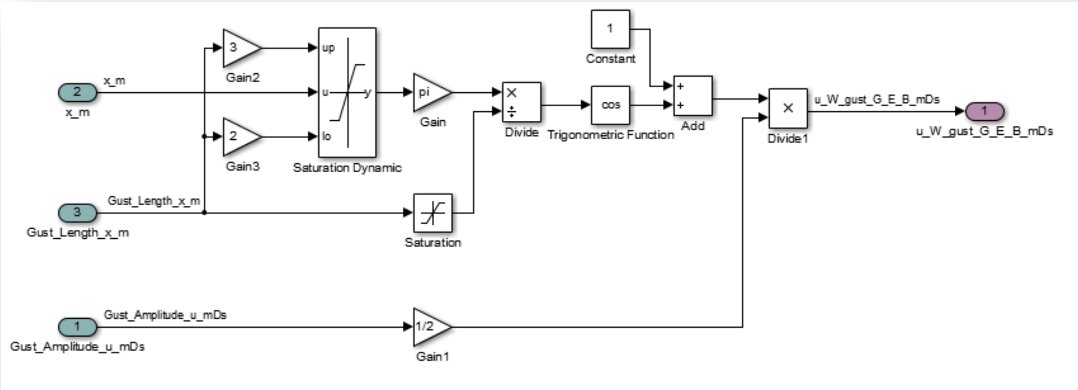


Figure 5-12 L4: f2_u

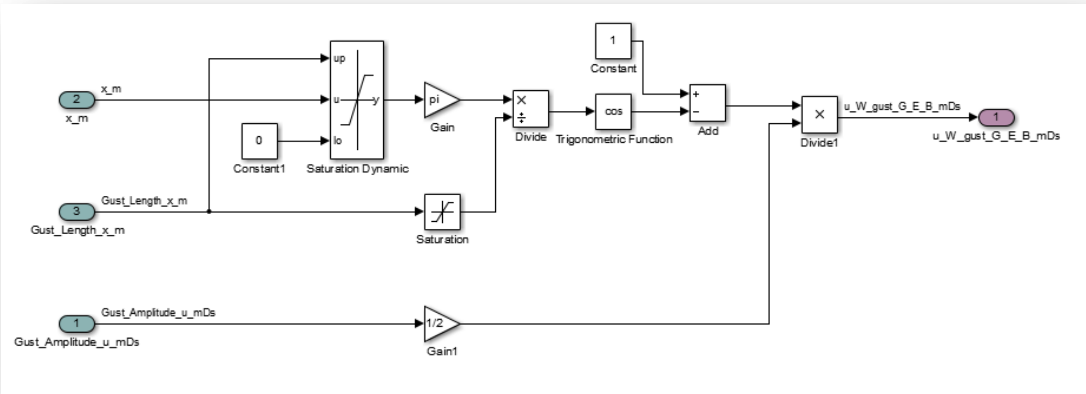


Figure 5-13 L4: f1_v

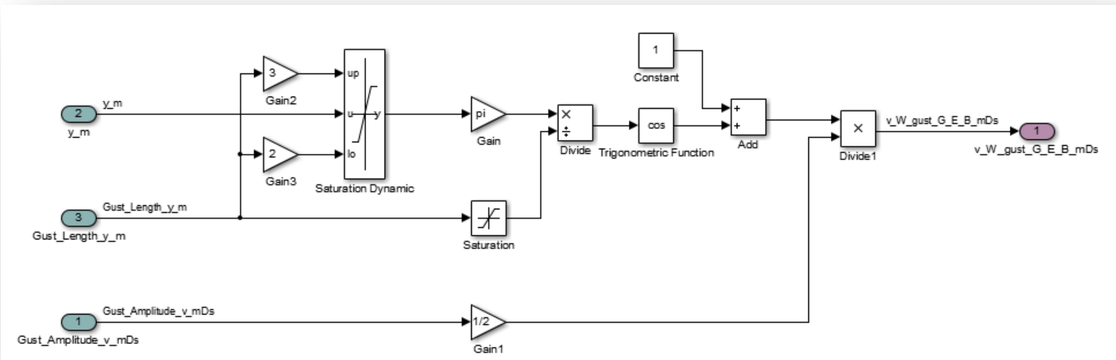


Figure 5-14 L4: f2_v

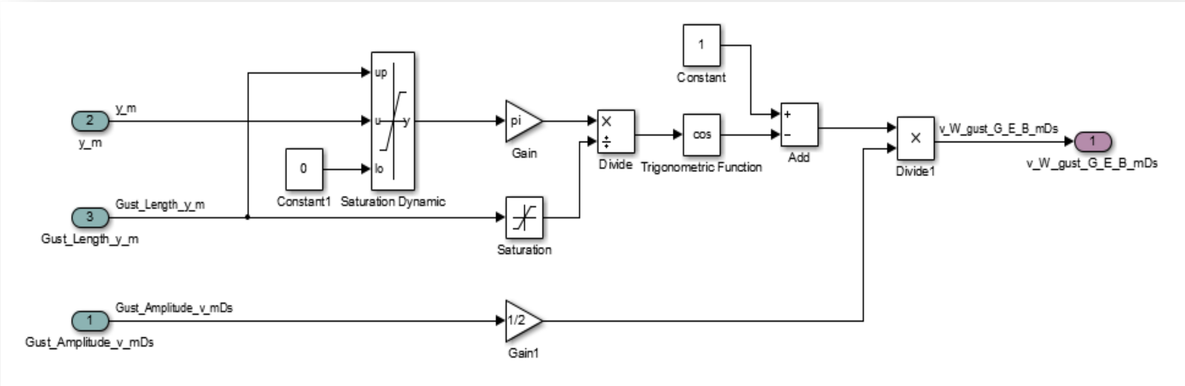


Figure 5-15 L4: f1_w

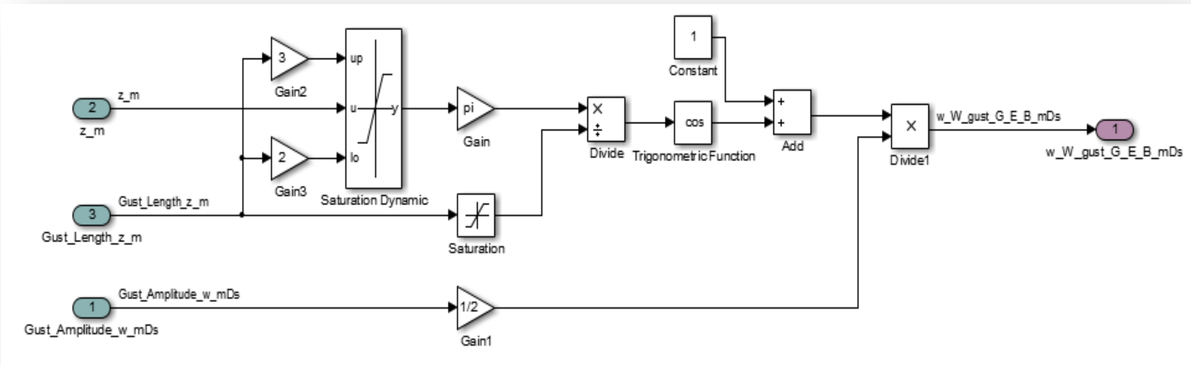
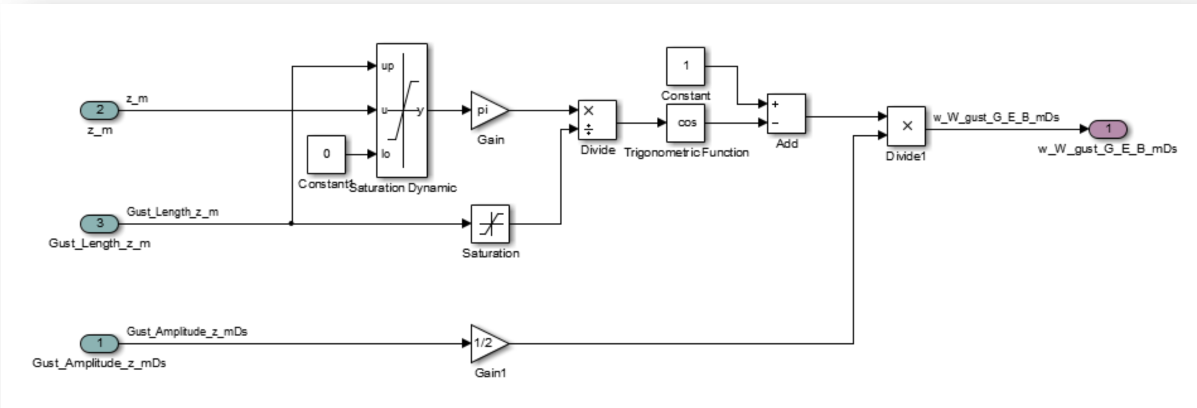


Figure 5-16 L4: f2_w



5.7 Verification Plan

5.7.1 Methods Used for Verification

5.7.1.1 Methods for Testing Functional Requirements

Requirement Name and ID	Description of Verification Method
R-FUN-ENV_GUST Computation of Wind Gust Velocity	Correct computation of Wind Gust Velocity demonstrated in ENV_GUST-Nominal_TC1, comparison to dissimilar implementation in ENV_GUST-Nominal_TC2 (see section 5.7.2.1).

Table 5-11 Methods for Testing Functional Requirements

5.7.1.2 Methods for Testing Implementation Requirements

Requirement Name and ID	Description of Verification Method
R-NUM-ENV_GUST Numeric Efficiency	Manual review of the implemented model. Records according to section 5.8.1.1
R-IOC-ENV_GUST Input / Output Interface Compliance to Parent System	Compliance to parent system will be verified at integration with parent system.
R-SGC-ENV_GUST Implementation Compliance to FSD Style Guides	Manual review of the implemented model. Records according to section 5.8.1.2
R-ISC-ENV_GUST Implementation Standards Compliance	Manual review of the implemented model. Records according to section 5.8.1.3.

Table 5-12 Methods for Testing Implementation Requirements

5.7.1.3 Methods for Testing Operational Requirements

Derived Standard Requirement Matrix:		
SIMULINK Modes	Simulink Offline Simulation	Demonstrated during test ENV_GUST-Nominal_TC1, ENV_GUST-Nominal_TC2 (see sections 5.7.2.1).
	Simulink Pseudo Real Time Simulation	Will be demonstrated on a higher level of the simulation model.
External Control for SIMULINK Execution	Bypassing of Non-Autonomous Elements	Not applicable as there are no non-autonomous elements present in the model.
	Single Point Execution	Demonstrated during test ENV_GUST-Nominal_TC1, ENV_GUST-Nominal_TC2 (see sections 5.7.2.1).
	Online Integration Freeze and Reset	Not applicable as there are no integrators present in the model.
	Workspace Initialization	Not applicable as there are no variables present to be initialized in the workspace.
	Runtime Parameter Tuning	Not applicable as there are no tunable parameters present in the model
RTW Code Generation	S-function	Generation of S-function demonstrated during test ENV_GUST-Operational_TC1 (see section 5.7.3.1).
Code Modes	Stand-alone Batch Simulation	Will be demonstrated on a higher level of the simulation model.
	Stand-alone Real Time Simulation	Will be demonstrated on a higher level of the simulation model.

Table 5-13 Methods for Testing Operational Requirements

5.7.2 Verification Plan for Functional Requirements

5.7.2.1 *Nominal Testing Procedure*

Test Name:	**Correct Calculation of Wind Gust Velocity**
Test ID:	ENV_GUST-Nominal_TC1
Related Requirements:	R-FUN-ENV_GUST: Computation of Wind Gust Velocity
Verification Data:	See section 5.8.2.1

The implemented simulation model is excited with different input combinations. Afterwards, the course of the Wind Gust Velocity is plotted over the distance traveled.

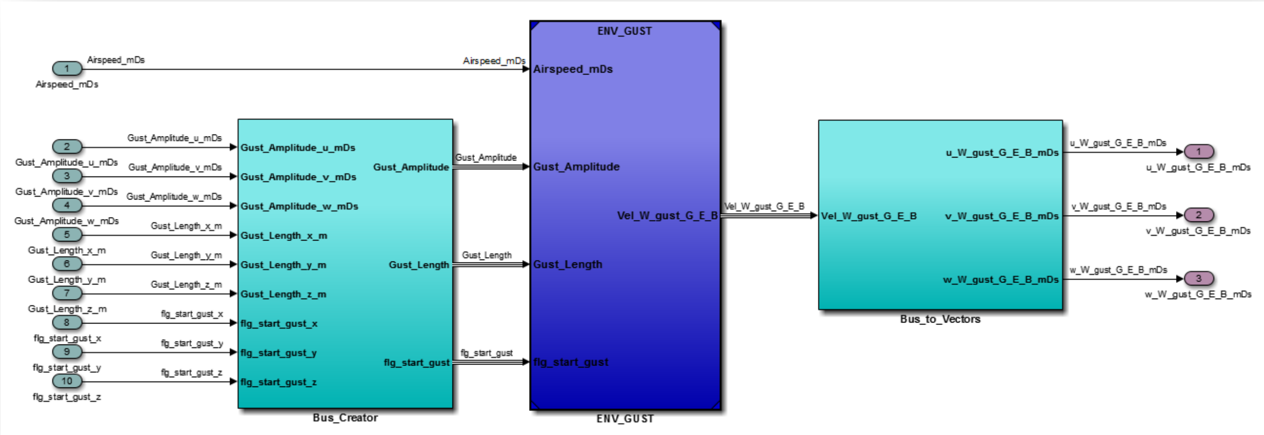


Figure 5-17 ENV_GUST-Nominal_TC1

To perform the test were used the following input signals:

- Airspeed = 35 m/s
- Gust_Amplitude = [3.0, 3.5, 4.0] m/s
- Gust_Length = [80,100,110] m
- flg_start_gust= [1, 1, 1]

Test Name:	**Equivalence of Implementation and Dissimilar Implementation**
Test ID:	ENV_GUST-Nominal_TC2
Related Requirements:	R-FUN-ENV_GUST: Computation of Wind Gust Velocity
Verification Data:	See section 5.8.2.1

The implemented model and a dissimilar implementation (Discrete Wind Gust Model SIMULINK Toolbox) are both excited with the same input signals. Afterwards, the output is checked by comparing the results.

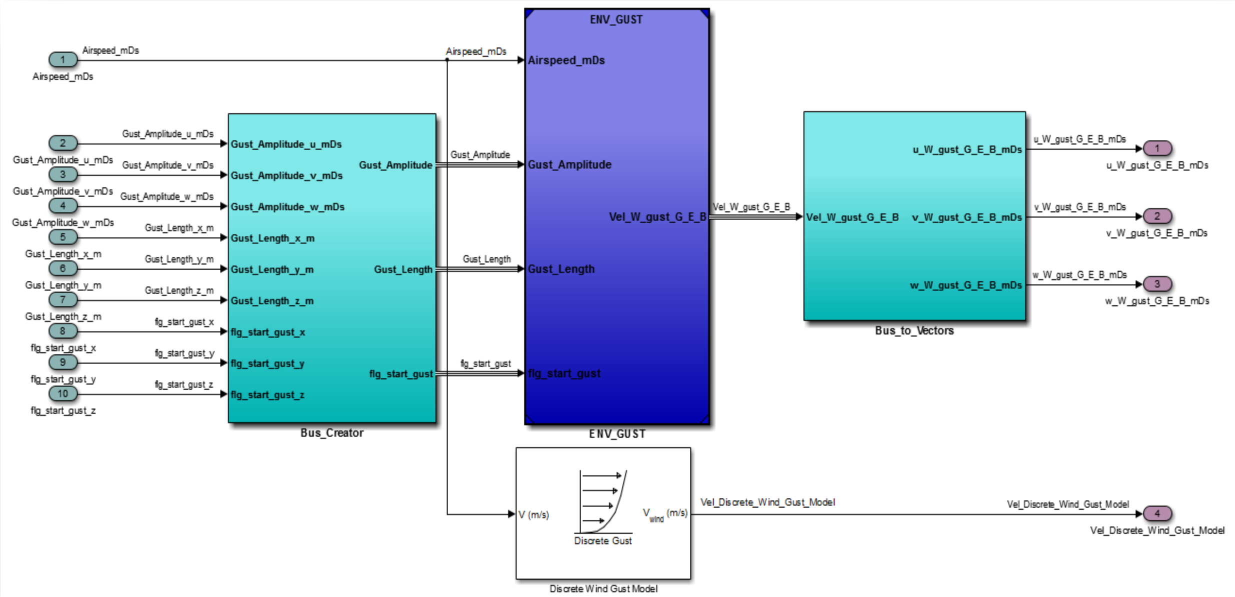


Figure 5-18 ENV_GUST-Nominal_TC2

To perform the test were used the following input signals:

- ❖ Airspeed = 35 m/s
- ❖ Gust_Amplitude = [3.0, 3.5, 4.0] m/s
- ❖ Gust_Length = [80,100,110] m
- ❖ flg_start_gust= [1, 1, 1]

5.7.3 Verification Plan for Operational Requirements

5.7.3.1 Code Generation and Equivalence Testing

Test Name:	**Code Generation and Equivalence Testing**
Test ID:	ENV_GUST-Operational_TC1
Related Requirements:	R-OPS-ENV_GUST: Operation Standard Requirements Matrix
Verification Data:	See section 5.8.3.1

The implemented simulation model running in normal mode and a S-function are excited with the same inputs signals.

Afterwards the outputs are checked by comparing the results .

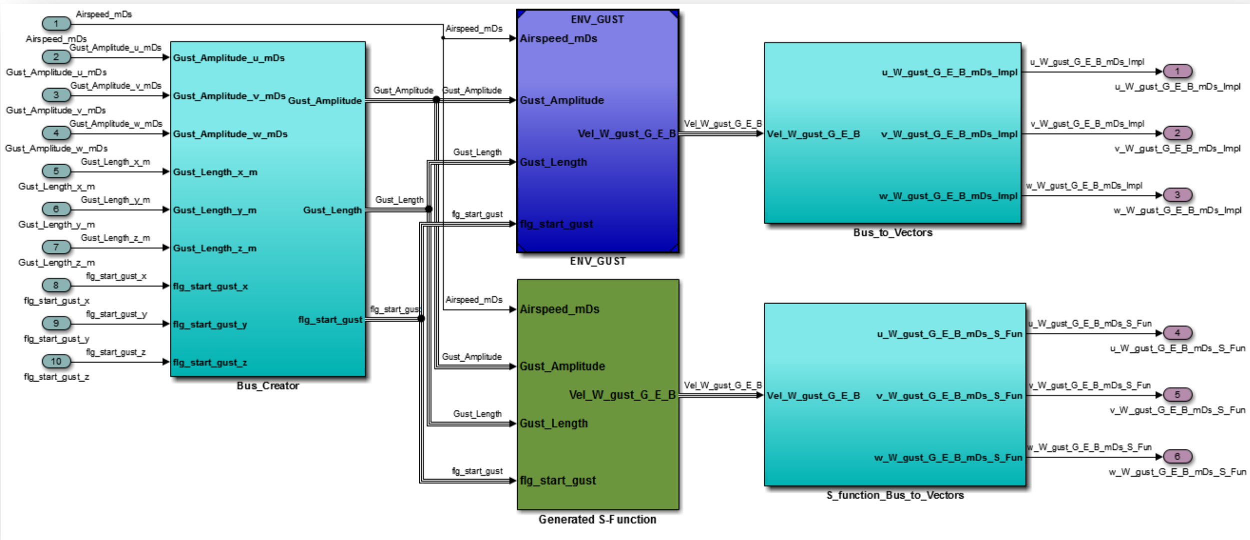


Figure 5-19 ENV_GUST-Operational_TC1

To perform the test were used the following input signals:

- Airspeed = 35 m/s
- Gust_Amplitude = [3.0, 3.5, 4.0] m/s
- Gust_Length = [80,100,110] m
- flg_start_gust= [1, 1, 1]

5.8 Verification Data

5.8.1 Verification of Implementation Requirements

5.8.1.1 Numeric Efficiency

Requirement Name **Numeric Efficiency**		Requirement ID R-NUM-ENV_GUST
Requirement is violated if the model contains one or more of the following items:		
Unused / Dead Code Branches Description	YES ☐	NO ☒
Computational Redundancies Description	YES ☐	NO ☒
Matrix Inversions Description	YES ☐	NO ☒
Scalar Expansions of Vector/Matrix Math Description	YES ☐	NO ☒
Circle Computations Description	YES ☐	NO ☒
Inefficient Lookup Table Programming Description	YES ☐	NO ☒
Algebraic Loops Description	YES ☐	NO ☒
Numeric Efficiency met?	**YES ☒**	**NO ☐**

Table 5-14 R-NUM-ENV_GUST

5.8.1.2 Implementation Compliance to FSD Style Guidelines

Requirement Name	Requirement ID
Implementation Compliance to FSD Style Guidelines	R-SGC-ENV_GUST
Requirement is violated if the model contains other blocks than the specified ones.	
Non-Specified Blocks within the Model?	YES ☐ NO ☒
Description	
Style Guide Compliance met?	**YES ☒ NO ☐**

Table 5-15 R-SGC-ENV_GUST

5.8.1.3 Implementation Standards Compliance

Requirement Name	Requirement ID	
Implementation Standards Compliance	R-ISC-ENV_GUST	
The coded algorithm must not contain any programming technique listed below unless detailed justification substantiates indispensability.		
Discrete Switches *	YES ☒	NO ☐
Memory Blocks	YES ☐	NO ☒
Time Delays	YES ☐	NO ☒
Time Dependent / Non-Autonomous Elements	YES ☐	NO ☒
In-lined Integrations	YES ☐	NO ☒
Hysteresis and Quantized Elements	YES ☐	NO ☒
Stochastic / Random Elements	YES ☐	NO ☒
Normal atan Blocks	YES ☐	NO ☒
Operations with Sign Loss	YES ☐	NO ☒
Value Flipping and Range Limiting	YES ☐	NO ☒
Math Function out of Range	YES ☐	NO ☒
Division by Zero	YES ☐	NO ☒
Finite State Transition	YES ☐	NO ☒
Implementation Standards Compliance met?	**YES ☒**	**NO ☐**

Table 5-16 R-ISC-ENV_GUST

* The switches in the system do not affect the correct operation.

5.8.2 Verification of Functional Requirements

5.8.2.1 Results for Nominal Testing

Test Name:	**Correct Calculation of Wind Gust Velocity**
Test ID:	ENV_GUST-Nominal_TC1
Verification Plan:	See section 5.7.2.1

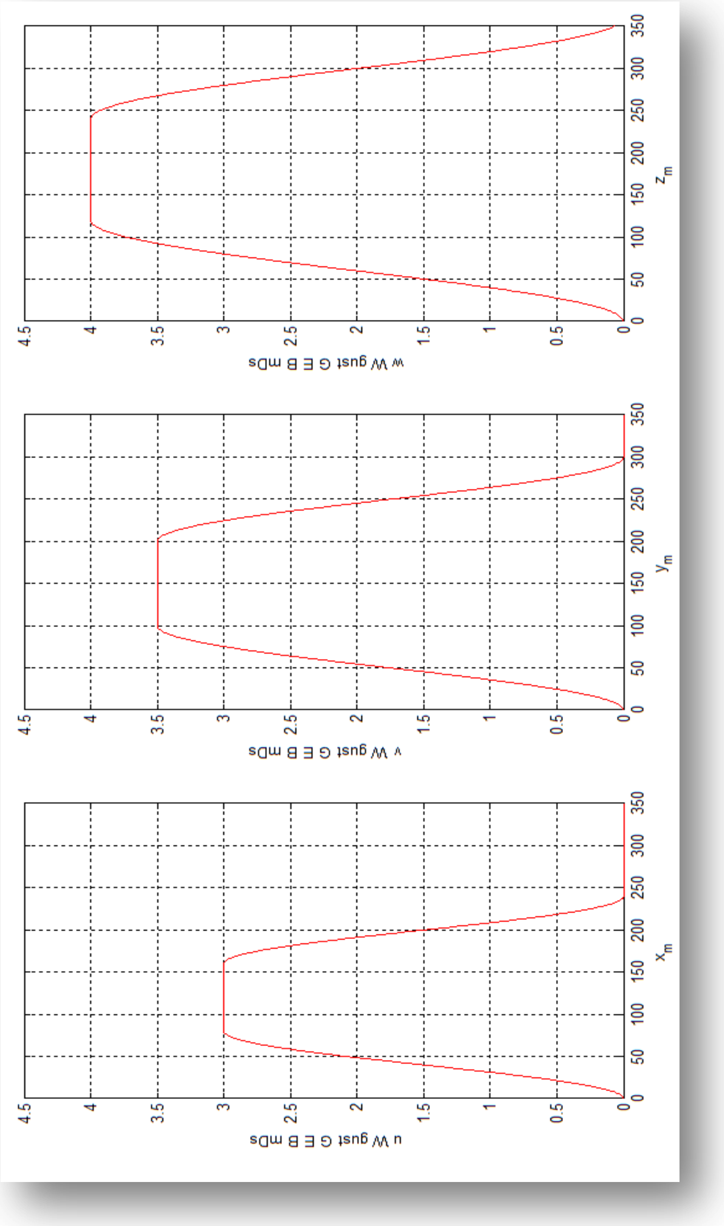
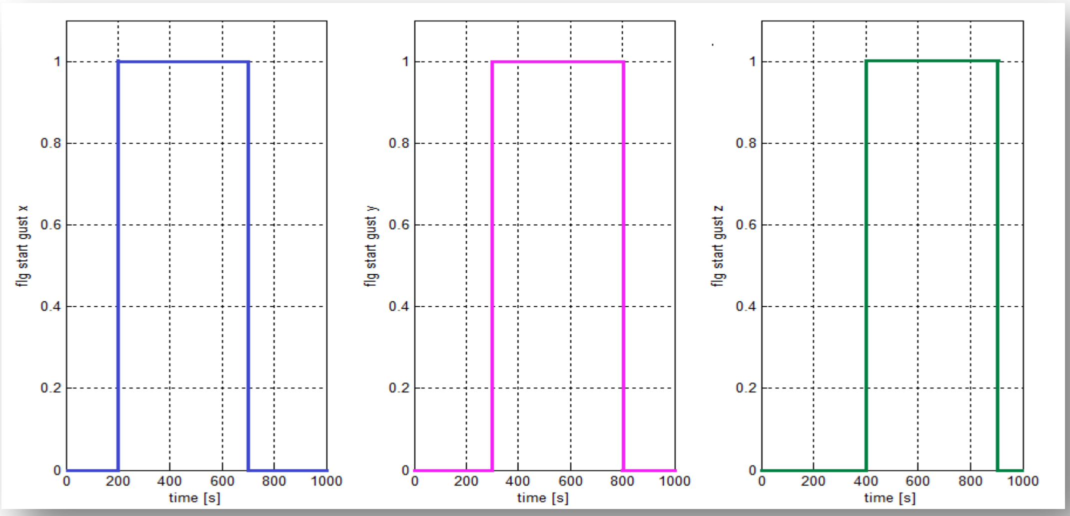


Figure 5-20 Correct Calculation of Wind Gust Velocity

Course of Wind Gust Velocity realistic?	**YES ☒**	**NO ☐**

Below is shown the behavior of the system with the following input signals:



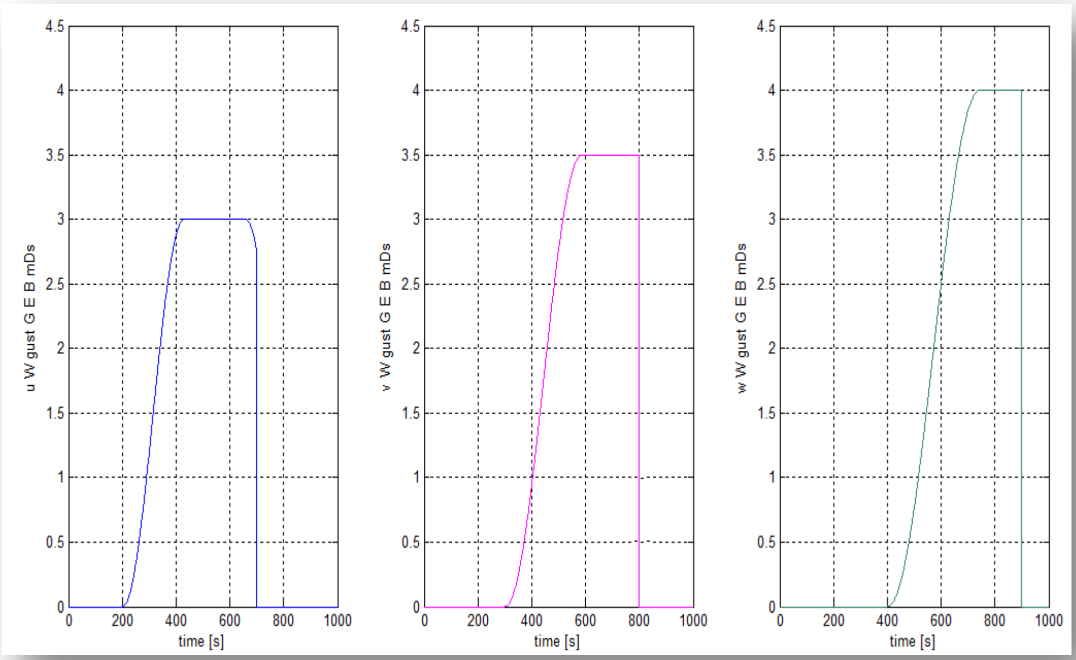


Figure 5-21 Behavior of the system

Test Name:	**Equivalence of Implementation and Dissimilar Implementation**
Test ID:	ENV_GUST-Nominal_TC2
Verification Plan:	See section 5.7.2.1

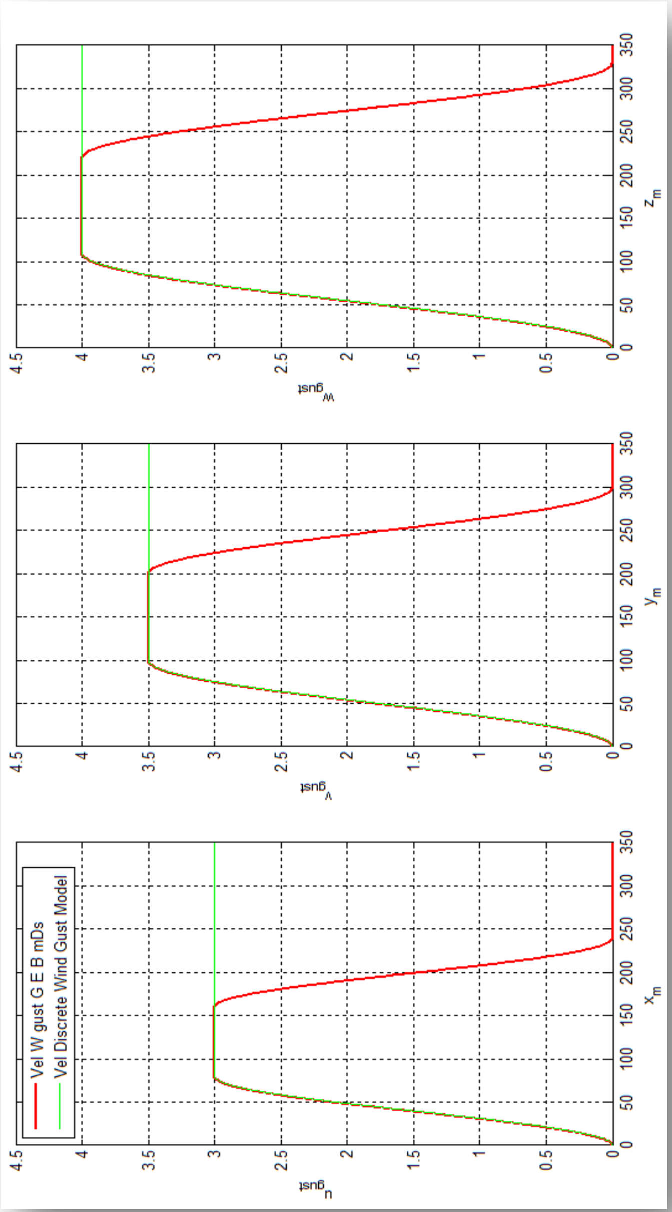


Figure 5-22 Equivalence of Implementation and Dissimilar Implementation (Discrete Wind Gust Model SIMULINK Toolbox)

Equivalence of Implemented Model and Dissimilar Implementation?	**YES ☒**	**NO ☐**
Correct Nominal Behavior?	**YES ☒**	**NO ☐**

5.8.3 Verification of Operational Requirements

5.8.3.1 Results for Code Generation and Equivalence Testing

Test Name:	**Code Generation and Equivalence Testing**
Test ID:	ENV_GUST-Operational_TC1
Verification Plan:	See section 5.7.3.2

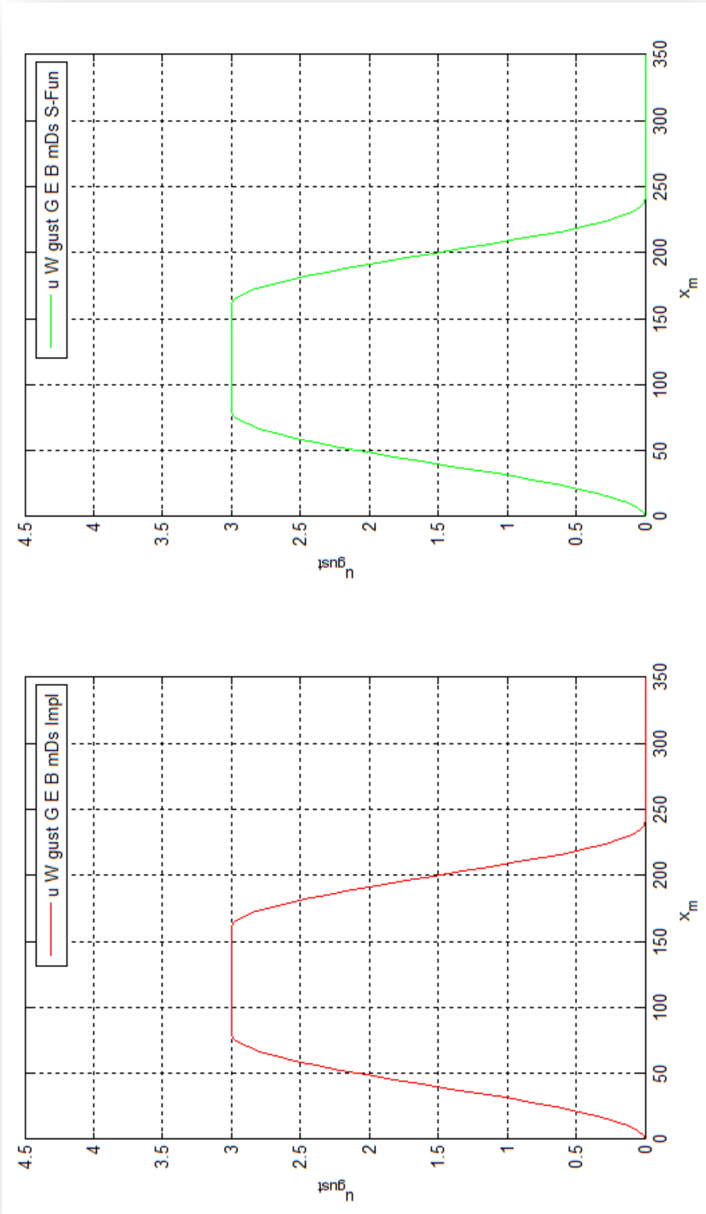


Figure 5-23 Equivalence of component u_W_gust_G_E_B_mDs

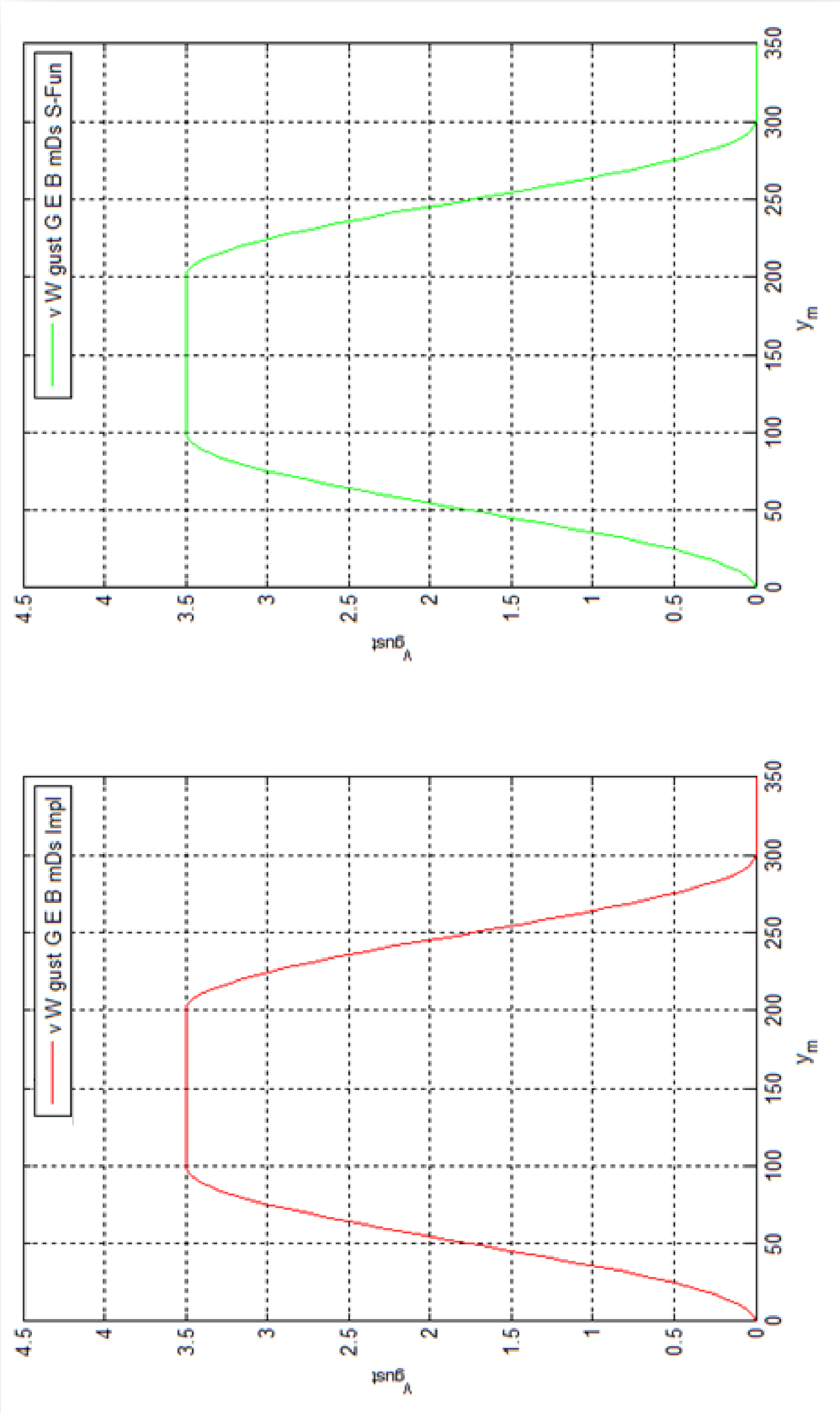


Figure 5-24 Equivalence of component v_W_gust_G_E_B_mDs

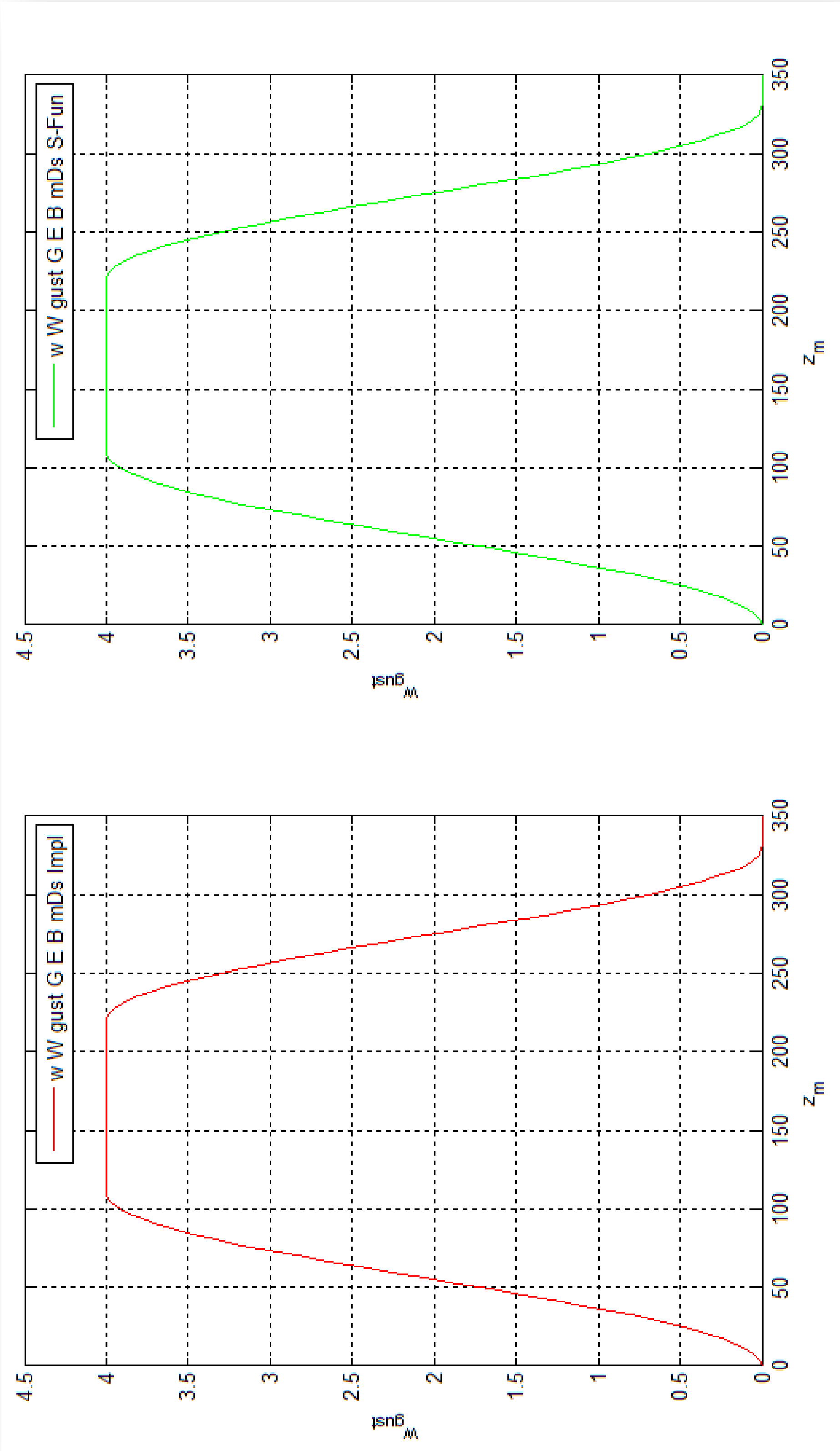


Figure 5-25 Equivalence of component w_W_gust_G_E_B_mDs

Compilation of S-function successful?	YES ☒	NO ☐
Compilation of standalone executable successful?	YES ☒	NO ☐
Description of Compilation Errors		
Warnings during Compilation?	YES ☐	NO ☒
Description of Compilation Warnings		

Run-Time Errors or Warnings?	YES ☐	NO ☒
Description of Error / Warning Messages		

Code Generation successful and Coded Version equivalent to Simulation Model?	**YES ☒**	**NO ☐**

6 Correlation Model

6.1 Introduction

In a turbulent flow, the velocity at a fixed point varies with time randomly. [3]-[4]

Alternatively, the fluid velocity at a fixed time depends on the position in a random manner.

An approximation universally accepted, is that temporal changes in the velocity field are negligible compared with the apparent temporal changes felt by the aircraft as it passes through spatial gradients. This is known as the frozen field approximation (Taylor's approximation). [11]

Using the Taylor's approximation and the assumption that the velocity field is homogeneous and isotropic, it is possible to define a correlation tensor as a function only of the distance between different points in the space and not on their location within the velocity field. [5]

6.2 Description of the Functional and Operational Intent

The system is intended to compute correlated Gaussian white noises.

The calculation of the correlation is necessary for the correct evaluation of Gaussian white noises used to compute the air velocity and the angular velocity of the air in a turbulent flow in twenty different points in the space, to take into account the case in which there are twenty aircraft flying in a restricted area.

The final model will be used as part of a simulation model, which shall be incorporated into the simulation framework of a flight simulation device. Therefore it is necessary that all subsystems are compliant to code generation requirements.

6.3 Requirements

6.3.1 Functional Requirements

Requirement Name **Computation of Signals**	Requirement ID R-FUN-ENV_CORR
Derived from Purpose of the system.	
Requirement Definition Correlated Gaussian white noises shall be computed.	

Table 6-1 R-FUN-ENV_CORR_01

6.3.2 Operational Requirements

Requirement Name **Incorporation into Flight Simulator Simulation Model**	Requirement ID R-OPS-ENV_CORR
Derived from Usage intents	
Requirement Definition The model shall be incorporated into the simulation model of an (atmospheric) flight simulation device. Therefore it is necessary that all components support code generation.	

Table 6-2 R-OPS-ENV_CORR

6.3.3 Implementation Requirements

Requirement Name **Numeric Efficiency**	Requirement ID R-NUM-ENV_CORR
Derived from Global Implementation Guidelines	
Requirement Definition The coded algorithm must not contain any of the numerical inefficient programming techniques listed below unless detailed justification substantiates indispensability.	
Programming Techniques to be Avoided:	
Unused / Dead Code Branches	
Computational Redundancies	
Matrix Inversions	
Scalar Expansion of Vector / Matrix Math	
Circle Computations	
Inefficient Lookup Table Programming	
Algebraic Loops	

Table 6-3 R-NUM-ENV_CORR

Requirement Name **Input / Output Interface Compliance to Parent System and Child Systems**	Requirement ID R-IOC-ENV_CORR
Derived from Global Implementation Guidelines	
Requirement Definition Input and Output interface must comply with parent system.	
Compliance required to:	
Global bus object definitions	
I/O signal name matching to parent system	
I/O signal unit matching to parent system	
I/O signal data type matching to parent system	
I/O signal data range compatibility matching to parent system	

Table 6-4 R-IOC-ENV_CORR

Requirement Name **Implementation Compliance to FSD Style Guidelines**	Requirement ID R-SGC-ENV_CORR
Derived from Global Implementation Guidelines	
Requirement Definition Only a subset of SIMULINK blocks is allowed to be implemented.	
Allowed Libraries and Toolboxes:	
FSD Compliant Base	
Use of other Libraries and Toolboxes is Forbidden!	

Table 6-5 R-SGC-ENV_CORR

Requirement Name **Implementation Standards Compliance**	Requirement ID R-ISC-ENV_CORR
Derived from Global Implementation Guidelines	
Requirement Definition The coded algorithm must not contain any programming technique listed below unless detailed justification substantiates indispensability.	
Forbidden Programming Techniques:	
Discrete Switches*	
Memory Blocks	
Time Delays	
Time Dependent / Non-Autonomous Elements	
In-lined Integrations	
Hysteresis and Quantized Elements	
Stochastic / Random Elements**	
Normal atan Blocks	
Operations with Sign Loss	
Value Flipping and Range Limiting	
Math Function out of Range	
Division by Zero	
Finite State Transition	

Table 6-6 R-ISC-ENV_CORR

* The switches in the system do not affect the correct operation.

** Stochastic / Random Elements in the system do not affect the correct operation.

6.4 Function Specification

6.4.1 Algorithm Abstract

The system is intended to compute the correlated Gaussian white noises necessary to calculate the Wind Velocity and the Wind Angular Rate in a turbulent flow.

6.4.2 Modeling Assumptions, Scope of Validity & Limitations

- Existence and determination of a characteristic length in the atmosphere:
  The characteristic length of the atmosphere is assumed to be given by the scale of turbulence. This length appears as a parameter in the mathematical description of turbulence and serves as useful indication of the influence of the turbulence environment on the response of the airplane [4];
- Model used for the description of Turbulence:
  The expressions used for the description of atmospheric turbulence are the Dryden spectral density functions;
- Correlation:
  The correlation between components of the air velocities are given by the correlation tensor;
- The atmosphere is described as a continuous, homogeneous, stationary and isotropic random process:
  These conditions are satisfied by the atmosphere in a localized area for a short periods of time;
- The Turbulence field is frozen: Taylor's Hypothesis:
  It is assumed, that the Turbulence field is frozen : the wind turbulence is a stochastic function of position but is not dependent on time [4];
- Cross-Correlation negligible:
  It is assumed, that the cross-correlation between the components of Air Velocity are negligible [3].

6.4.3 Detailed Algorithm Description

6.4.3.1 Correlation tensor

The mathematical model used to describe the correlation between the components of air velocity is the correlation tensor.

Suppose the velocity components at two points $P(\, x_1, x_2, x_3\,)$ and $P'(\, x_1', x_2', x_3'\,)$ are $(\, u_1, u_2, u_3\,)$ and $(\, u_1', u_2', u_3'\,)$ respectively. The nine quantities $\overline{u_\iota u_J'}$ for $i, j = 1{,}2{,}3$ may be shown to be the components of a second rank tensor. The correlation tensor *R* is defined by:

$$\bar{u}^2 R = \bar{u}^2 \begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix} = \begin{pmatrix} \overline{u_1 u_1'} & \overline{u_1 u_2'} & \overline{u_1 u_3'} \\ \overline{u_2 u_1'} & \overline{u_2 u_2'} & \overline{u_2 u_3'} \\ \overline{u_3 u_1'} & \overline{u_3 u_2'} & \overline{u_3 u_3'} \end{pmatrix} \qquad \text{6-1}$$

The components of *R* can be evaluated in terms of correlation functions $f(r,t)$ and $g(r,t)$ and vector $\boldsymbol{r}$ whose components are $\xi_1 = x_1 - x_1'$, $\xi_2 = x_2 - x_2'$, $\xi_3 = x_3 - x_3'$. In this way the correlation tensor *R* is defined by:

$$R = \frac{\{\, f(r,t) - g(r,t)\,\}}{r^2}.\boldsymbol{rr} + g(r,t)\cdot \boldsymbol{I} \qquad \text{6-2}$$

where $r = |\boldsymbol{r}|$ and $\boldsymbol{I}$ is the unit tensor $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$. [5]

6.4.3.2 Mathematical Representation

A measure of the dependence of a gust velocity component on a spatial coordinate is given by comparing the characteristic length of turbulence, which is given by the scale of turbulence *L*.

The scale of turbulence is used as a parameter in the mathematical description of atmospheric turbulence and is a function of the altitude.

The most common expression for the representation of atmospheric turbulence are the Dryden and Von Karman spectral density functions. Here the Dryden representation is used. Using the Taylor's approximation, Dryden presented an expression for the description of turbulence in terms of correlation functions [5]:

$$\begin{cases} f(r) = e^{-\frac{r}{L}} \\ g(r) = e^{-\frac{r}{L}}\left(1-\frac{r}{2L}\right) \end{cases} \qquad 6\text{-}3$$

The parameters defined as a function of the altitude h are listed below [3]:

Low Altitude ($h^G < 1000\,ft$)

The scale of turbulence in low altitude is defined by:

$$\begin{cases} L_w = h^G \\ L_u = L_v = \dfrac{h^G}{(0.177+0.000823\cdot h^G)^{1.2}} \end{cases} \qquad 6\text{-}4$$

where h^G is the altitude in feet.

In this region, the longitudinal turbulence velocity u_{turb} is aligned along the horizontal relative mean wind vector and the vertical turbulence velocity w_{turb} is aligned with vertical.

Medium/High Altitude ($h^G > 1000\,ft$)

The scale of turbulence is based on the assumption that the turbulence is isotropic. The scales of turbulence are:

$$L_u = L_v = L_w = 1750\,ft \qquad 6\text{-}5$$

In this region the axes are aligned with the body coordinates.

6.4.3.3 Correlation tensor for twenty aircraft

The turbulence components u_{turb} , v_{turb}, w_{turb} and p_{turb} shall be considered mutually independent (uncorrelated) in a statistical sense. However, q_{turb} is correlated with w_{turb} and r_{turb} with v_{turb} [3].

Thus, the correlation tensor between two aircraft is diagonal:

$$R = \begin{pmatrix} R_{11} & 0 & 0 \\ 0 & R_{22} & 0 \\ 0 & 0 & R_{33} \end{pmatrix} \quad \text{6-6}$$

To see what happens when twenty aircraft flying in a limited area, it is necessary to calculate the correlation tensor for twenty aircraft.

It will look like:

$$R = \begin{pmatrix} [R_u] & \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{bmatrix} & \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{bmatrix} \\ \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{bmatrix} & [R_v] & \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{bmatrix} \\ \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{bmatrix} & \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{bmatrix} & [R_w] \end{pmatrix} \quad \text{6-7}$$

where R_u , R_v and R_w are matrices of size [20x20] and each contain the correlation between the velocity components *u, v* and *w* respectively, while outside the diagonal there are matrices of zeroes, each of size [20x20].

The diagonal elements are all unitary. The matrix *R* is then size [60x60].

For example, the matrix R_u will be of the type:

$$R_u = \begin{pmatrix} R_{u11} & R_{u12} & R_{u13} & R_{u14} & R_{u15} & \cdots & R_{u1\,20} \\ R_{u21} & R_{u22} & R_{u23} & R_{u24} & R_{u25} & \cdots & R_{u2\,20} \\ R_{u31} & R_{u32} & R_{u33} & R_{u34} & R_{u35} & \cdots & R_{u3\,20} \\ R_{u41} & R_{u42} & R_{u43} & R_{u44} & R_{u45} & \cdots & R_{u4\,20} \\ R_{u51} & R_{u52} & R_{u53} & R_{u54} & R_{u55} & \cdots & R_{u5\,20} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ R_{u20\,1} & R_{u20\,2} & R_{u20\,3} & R_{u20\,4} & R_{u20\,5} & \cdots & R_{u20\,20} \end{pmatrix} \quad \text{6-8}$$

Each element R_{uij} represents the correlation between u_i and u_j.

The position of each aircraft is univocally determined by knowing its coordinates:

- altitude h^G
- latitude φ^G
- longitude λ^G

for $i, j = 1, \dots, 20$ it is possible to define the following quantities:

$$\Delta h^G{}_{ij} = \left| h^G{}_i - h^G{}_j \right| \quad \text{6-9}$$

$$\Delta \varphi^G{}_{ij} = \left| \varphi^G{}_i - \varphi^G{}_j \right| \quad \text{6-10}$$

$$\Delta \lambda^G{}_{ij} = \left| \lambda^G{}_i - \lambda^G{}_j \right| \quad \text{6-11}$$

$$h^G{}_{ij} = \frac{h^G{}_i + h^G}{2} \quad \text{6-12}$$

$$\varphi^G{}_{ij} = \frac{\varphi^G{}_i + \varphi^G{}_j}{2} \quad \text{6-13}$$

These are [20x20] symmetric matrices. If N_φ is the radius of the Earth, its average value for each pair of planes will be:

$$N_{\varphi_{ij}} = \frac{N_{\varphi_i} + N_{\varphi_j}}{2} \quad \text{6-14}$$

It is possible to find the relative position of all the aircraft in the NED (O) frame:

$$\begin{cases} \Delta x^o_{ij} = \Delta \varphi^G{}_{ij} \cdot \left(N_{\varphi_{ij}} + h^G{}_{ij} \right) \\ \Delta y^o_{ij} = \Delta \lambda^G{}_{ij} \cdot \left(N_{\varphi_{ij}} + h^G{}_{ij} \right) \cdot cos\left(\varphi^G{}_{ij} \right) \\ \Delta z^o_{ij} = \Delta h^G{}_{ij} \end{cases} \quad \text{6-15}$$

All these values are given as [20x20] matrices. To perform the calculation it is necessary to collect the components into vectors [400x1].

Knowing the attitude of each aircraft, it is possible to calculate their average values in order to obtain "averages" body frames:

$$\Psi_{ij} = \frac{\Psi_i + \Psi_j}{2} \quad \text{6-16}$$

$$\Theta_{ij} = \frac{\Theta_i + \Theta_j}{2} \quad \text{6-17}$$

$$\Phi_{ij} = \frac{\Phi_i + \Phi_j}{2} \quad \text{6-18}$$

These are symmetric matrices [20x20]. To perform the calculation it is necessary to collect the components into vectors [400x1].

Using the rotation matrix:

$$Rot_{O,B} = \begin{pmatrix} cos\Psi cos\Theta & cos\Phi sin\Psi + cos\Psi sin\Phi sin\Theta & sin\Phi sin\Psi - cos\Phi cos\Psi sin\Theta \\ -cos\Theta sin\Psi & cos\Phi cos\Psi - sin\Phi sin\Theta sin\Psi & cos\Psi sin\Phi + cos\Phi sin\Psi sin\Theta \\ sin\Theta & -cos\Theta sin\Phi & cos\Phi cos\Theta \end{pmatrix} \quad \text{6-19}$$

it is possible to obtain the distances between all the aircraft in the body frame:

$$\begin{cases} \Delta x_{ij}^B = (cos\Psi_{ij} cos\Theta_{ij})\Delta x_{ij}^O + (cos\Phi_{ij} sin\Psi_{ij} + cos\Psi_{ij} sin\Phi_{ij} sin\Theta_{ij})\Delta y_{ij}^O + \\ \quad + (sin\Phi_{ij} sin\Psi_{ij} - cos\Phi_{ij} cos\Psi_{ij} sin\Theta_{ij})\Delta z_{ij}^O \\ \Delta y_{ij}^B = (-cos\Theta_{ij} sin\Psi_{ij})\,\Delta x_{ij}^O + (cos\Phi_{ij} cos\Psi_{ij} - sin\Phi_{ij} sin\Theta_{ij} sin\Psi_{ij})\Delta y_{ij}^O + \\ \quad + (cos\Psi_{ij} sin\Phi_{ij} + cos\Phi_{ij} sin\Psi_{ij} sin\Theta_{ij})\Delta z_{ij}^O \\ \Delta z_{ij}^B = (sin\Theta_{ij})\Delta x_{ij}^O + (-cos\Theta_{ij} sin\Phi_{ij})\Delta y_{ij}^O + (cos\Phi_{ij} cos\Theta_{ij})\Delta z_{ij}^O \end{cases} \quad \text{6-20}$$

These are all vectors [400x1]. Starting from these vectors, it is possible to rebuild symmetric matrices [20x20].

The elements of these matrices are necessary for the calculation of the correlation functions in the high altitude model.

In the low altitude model all the distances are given in the wind frame (W). Starting from the distances in the NED (O) frame as vectors [400x1] it is possible to obtain the distances in wind frame using the rotation matrix:

$$Rot_{O,W} = \begin{pmatrix} -cos\chi & -sin\chi & 0 \\ sin\chi & -cos\chi & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \text{6-21}$$

where χ is the angle between the North direction and the x direction in the wind frame.

By defining a matrix [20x20] of average values:

$$\chi_{ij} = \frac{\chi_i + \chi_j}{2} \tag{6-22}$$

collecting its components in a vector [400x1], is obtained the distances in wind frame:

$$\begin{cases} \Delta x_{ij}^W = -cos\chi_{ij} \cdot \Delta x_{ij}^o - sin\chi_{ij} \cdot \Delta y_{ij}^o \\ \Delta y_{ij}^W = sin\chi_{ij} \cdot \Delta x_{ij}^o - cos\chi_{ij} \cdot \Delta y_{ij}^o \\ \Delta z_{ij}^W = \Delta z_{ij}^o \end{cases} \tag{6-23}$$

Starting from these vectors, symmetric matrices [20x20] are rebuild. The elements of these matrices are necessary for the calculation of the correlation functions in the low altitude model.

Once calculated distances (*r*) and scales of turbulence (L) in models of high and low altitude, it is possible to calculate the correlation tensors by using equation (6-2) and (6-3). These tensors R_u, R_v and R_w are calculated for the three velocity components respectively u_{turb}, v_{turb} and w_{turb}.

6.4.4 White noises and correlation

Being known correlation tensors, i.e. knowing what correlation you would expect to find between the perceived speeds of the various aircraft, special attention has been spent in the research for a method that would allow to achieve this expected correlation.

The main idea to take into account the correlation between the velocities perceived by various aircraft, was to use white noises in the Dryden Model of Turbulence not completely random but capable of fulfilling the previously calculated correlation.

6.4.4.1 Construction of the procedure

<u>STEP 1</u>

The procedure was built in MATLAB®, from "correlation matrices" whose elements were arbitrarily chosen as functions of the type $f = e^{-x}$, for example

$$R = \begin{pmatrix} 1 & f \\ f & 1 \end{pmatrix} \tag{6-24}$$

Building two random signals, for example:

$$\begin{cases} signal_1 = (1+f)\cdot rand_1 \\ signal_2 = (1+f)\cdot rand_2 \end{cases} \quad \text{6-25}$$

where $rand_1$ and $rand_2$ are Gaussian random signals of length 5e4, computing their correlation with the Matlab command *xcorr* is seen as the result does not approach the element R_{12}, i.e. the starting function $f = e^{-x}$:

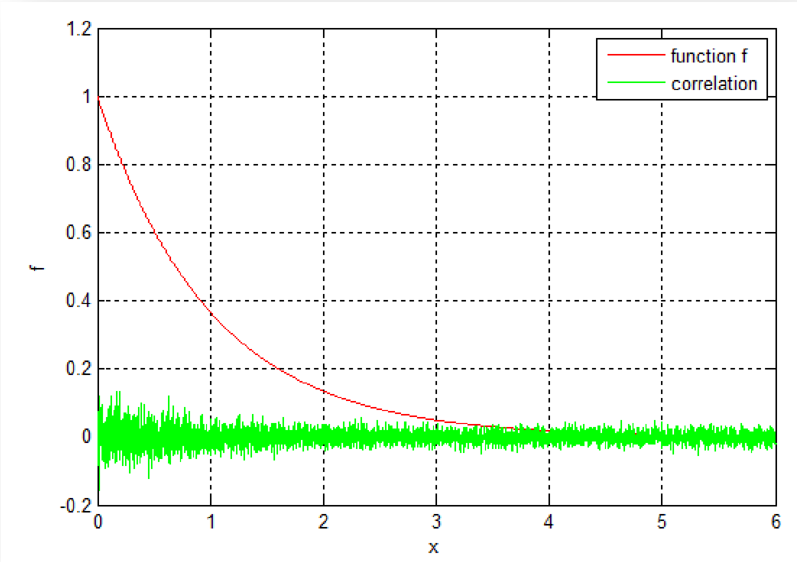


Figure 6-1 Correlation: test 1

Reflecting, however, on the definition of correlation, variance and its properties:

$$var(A\cdot rand_1 + B\cdot rand_2) = A^2\cdot var(rand_1) + B^2\cdot var(rand_2) + + 2\cdot A\cdot B\cdot cov(rand_1, rand_2) \quad \text{6-26}$$

where A and B are coefficients. Considering that $rand_1$ and $rand_2$ are Gaussian random signals (then with unitary variance) and that are to be obtained new random signals , $signal_1$ and $signal_2$, still Gaussian, the above equation becomes:

$$var(A\cdot rand_1 + B\cdot rand_2) = A^2 + B^2 = 1 \quad \text{6-27}$$

If D is the matrix of eigenvalues (λ) of R and V the matrix of the corresponding eigenvectors (v), then it will be:

$$V^T \cdot R \cdot V = D \ \rightarrow R = V \cdot D \cdot V^T \tag{6-28}$$

consequently the matrix R can be written as:

$$R = \left(V \cdot \sqrt{D}\right) \cdot \left(\sqrt{D} \ \cdot V^T\right) \tag{6-29}$$

where

$$\left(V \cdot \sqrt{D}\right) = \begin{pmatrix} v_1^{(1)} \cdot \sqrt{\lambda_1} & v_2^{(1)} \cdot \sqrt{\lambda_2} & \cdots & v_{20}^{(1)} \cdot \sqrt{\lambda_{20}} \\ v_1^{(2)} \cdot \sqrt{\lambda_1} & v_2^{(2)} \cdot \sqrt{\lambda_2} & \cdots & v_{20}^{(2)} \cdot \sqrt{\lambda_{20}} \\ \vdots & \vdots & \ddots & \vdots \\ v_1^{(20)} \cdot \sqrt{\lambda_1} & v_2^{(20)} \cdot \sqrt{\lambda_2} & \cdots & v_{20}^{(20)} \cdot \sqrt{\lambda_{20}} \end{pmatrix} \tag{6-30}$$

and

$$\left(\sqrt{D} \ \cdot V^T\right) = \begin{pmatrix} \sqrt{\lambda_1} \cdot v_1^{(1)} & \sqrt{\lambda_1} \cdot v_1^{(2)} & \cdots & \sqrt{\lambda_1} \cdot v_1^{(20)} \\ \sqrt{\lambda_2} \cdot v_2^{(1)} & \sqrt{\lambda_2} \cdot v_2^{(2)} & \cdots & \sqrt{\lambda_2} \cdot v_2^{(20)} \\ \vdots & \vdots & \ddots & \vdots \\ \sqrt{\lambda_{20}} \cdot v_{20}^{(1)} & \sqrt{\lambda_{20}} \cdot v_{20}^{(2)} & \cdots & \sqrt{\lambda_{20}} \cdot v_{20}^{(20)} \end{pmatrix} \tag{6-31}$$

Each element of matrix R, for $i, j = 1, \dots, 20$, is:

$$R_{ij} = \left(v_1^{(i)} \cdot \sqrt{\lambda_1}, \ v_2^{(i)} \cdot \sqrt{\lambda_2}, \dots, \ \sqrt{\lambda_{20}} \cdot v_{20}^{(i)}\right) \cdot \begin{pmatrix} \sqrt{\lambda_1} \cdot v_1^{(j)} \\ \sqrt{\lambda_2} \cdot v_2^{(j)} \\ \vdots \\ \sqrt{\lambda_{20}} \cdot v_{20}^{(j)} \end{pmatrix} \tag{6-32}$$

Whereas the R matrix is symmetric and has all unitary values on the diagonal, it will be:

$$R_{kk} = \lambda_1 \cdot \left(v_1^{(k)}\right)^2 + \lambda_2 \cdot \left(v_2^{(k)}\right)^2 + \dots + \lambda_{20} \cdot \left(v_{20}^{(k)}\right)^2 = 1 \tag{6-33}$$

For a matrix R of size [2x2] then with $i, j = 1,2$, the equation 6-32 will be:

$$R_{ij} = \lambda_1 \cdot v_1^{(i)} \cdot v_1^{(j)} + \lambda_2 \cdot v_2^{(i)} \cdot v_2^{(j)} \tag{6-34}$$

therefore:

$$\begin{cases} R_{11} = \lambda_1 \cdot \left(v_1^{(1)}\right)^2 + \lambda_2 \cdot \left(v_2^{(1)}\right)^2 \\ R_{22} = \lambda_1 \cdot \left(v_1^{(2)}\right)^2 + \lambda_2 \cdot \left(v_2^{(2)}\right)^2 \\ R_{12} = \lambda_1 \cdot v_1^{(1)} v_1^{(2)} + \lambda_2 \cdot v_2^{(1)} v_2^{(2)} \end{cases} \quad \text{6-35}$$

with reference to equation 6-27, will be:

$$\begin{cases} A = \sqrt{\lambda_1} \cdot [v_1] \\ B = \sqrt{\lambda_2} \cdot [v_2] \end{cases} \quad \text{6-36}$$

These considerations, together with a series of tests "by trial and error", have made it possible to build new signals to perform the additional tests:

$$\begin{bmatrix} signal_1 \\ signal_2 \end{bmatrix} = [A \cdot rand_1 + B \cdot rand_2] \quad \text{6-37}$$

then

$$\begin{bmatrix} signal_1 \\ signal_2 \end{bmatrix} = \left(\sqrt{\lambda_1} \cdot [v_1]\right) \cdot rand_1 + \left(\sqrt{\lambda_2} \cdot [v_2]\right) \cdot rand_2 \quad \text{6-38}$$

Considering now the definition of correlation for a pair of signals:

$$corr_{1,2} = E[signal_1 \cdot signal_2]$$

$$corr_{1,2} = E[\, \lambda_1 \cdot v_1^{(1)} \cdot v_1^{(2)} \cdot (rand_1)^2 +$$
$$+ \sqrt{\lambda_1 \cdot \lambda_2} \cdot \left(v_1^{(1)} \cdot v_2^{(2)} + v_1^{(2)} \cdot v_2^{(1)}\right) \cdot rand_1 \cdot rand_2 +$$
$$+ \lambda_2 \cdot v_2^{(1)} \cdot v_2^{(2)} \cdot (rand_2)^2 \,] \quad \text{6-39}$$
$$corr_{1,2} = \lambda_1 \cdot v_1^{(1)} \cdot v_1^{(2)} \cdot E[\,(rand_1)^2\,] +$$
$$+ \sqrt{\lambda_1 \cdot \lambda_2} \cdot \left(v_1^{(1)} \cdot v_2^{(2)} + v_1^{(2)} \cdot v_2^{(1)}\right) \cdot E[\,rand_1 \cdot rand_2] +$$
$$+ \lambda_2 \cdot v_2^{(1)} \cdot v_2^{(2)} \cdot E[\,(rand_2)^2\,]$$

But

$$\begin{cases} E[\,(rand_1)^2\,] = var(rand_1) = 1 \\ E[\,(rand_2)^2\,] = var(rand_2) = 1 \\ E[\,rand_1 \cdot rand_2] = cov(\,rand_1 \cdot rand_2) = 0 \end{cases} \quad \text{6-40}$$

The equation 6-39 becomes:

$$corr_{1,2} = \lambda_1 \cdot v_1^{(1)} \cdot v_1^{(2)} + \lambda_2 \cdot v_2^{(1)} \cdot v_2^{(2)} \quad \text{6-41}$$

which is equal to element R_{12} (see equation 6-35).

In conclusion, given a correlation matrix R, constructing each pair of signals i and j in this way, their correlation is corresponding to the element R_{ij}.

Then recalculating the correlation between the two new signals $signal_1$ and $signal_2$, is obtained:

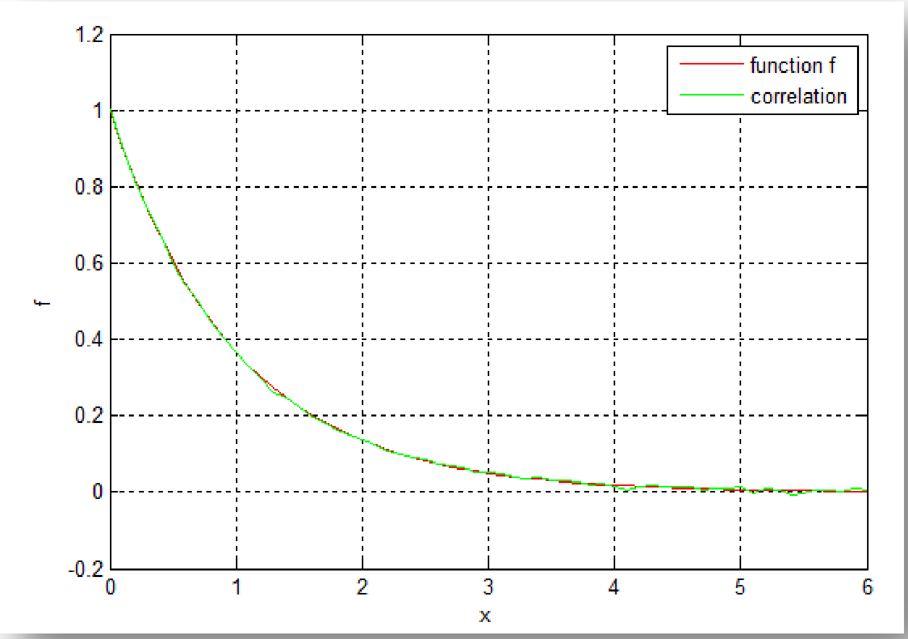


Figure 6-2 Correlation: test 2

The pair of signals obtained by equations 6-38, is obviously still Gaussian:

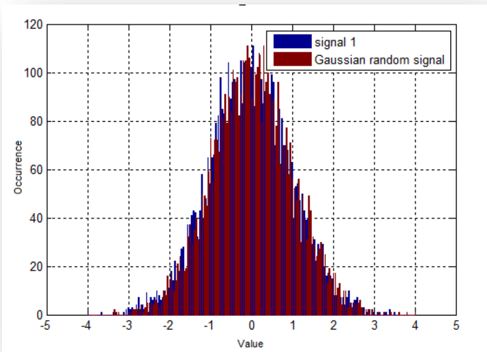


Figure 6-3 Signal 1

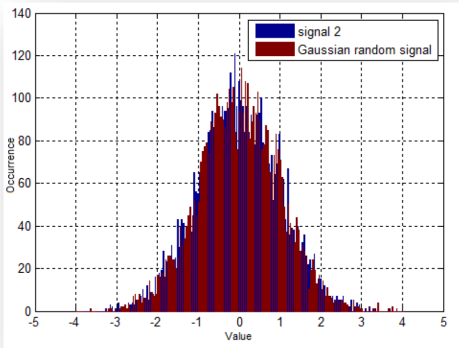
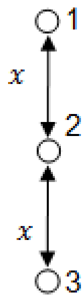


Figure 6-4 Signal 2

STEP 2

A first attempt to generalize the procedure, was done using a matrix of size [3x3], whose elements are functions of the distance x between different points:



$$f = e^{-x} \tag{6-42}$$

$$R = \begin{pmatrix} 1 & e^{-x} & e^{-2x} \\ e^{-x} & 1 & e^{-x} \\ e^{-2x} & e^{-x} & 1 \end{pmatrix} = \begin{pmatrix} 1 & f & f^2 \\ f & 1 & f \\ f^2 & f & 1 \end{pmatrix} \tag{6-43}$$

Constructing the new signals using the following formula:

$$[signal] = [\text{rand}] \cdot \sqrt{D} \cdot V^{\mathrm{T}} \tag{6-44}$$

where D is the matrix of the eigenvalues of R, V is the matrix of the corresponding eigenvectors and $[\text{rand}]$ is a vector of size $[n, 3]$ where, for example, $n = 5e4$, and by computing for each pair of signals their correlation, it is obtained:

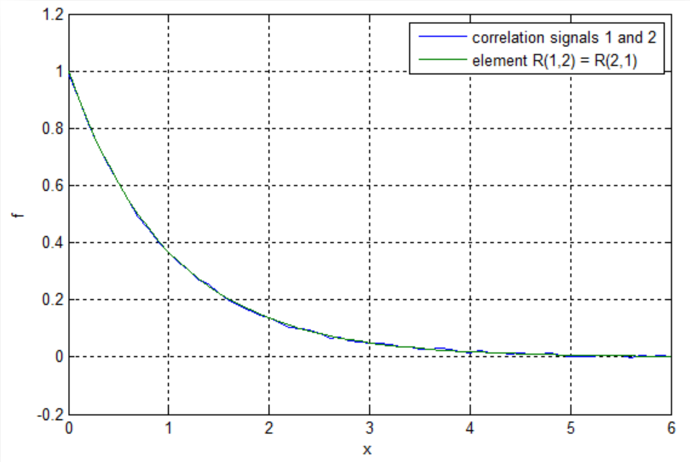


Figure 6-5 Signals 1 and 2

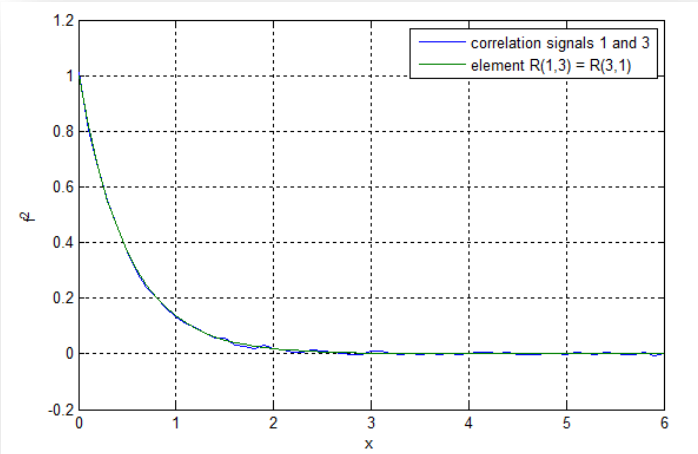


Figure 6-6 Signals 1 and 3

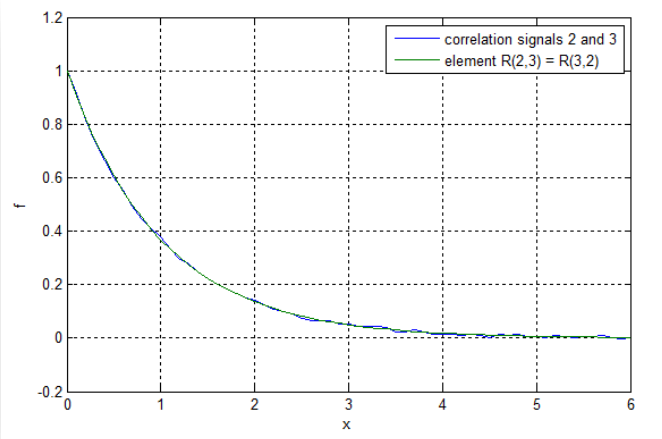


Figure 6-7 Signals 2 and 3

At this point, the same procedure was used on matrices of size [20x20]: R_u, R_v and R_w.

Starting from the three correlation tensors R_u, R_v and R_w and computing their eigenvalues (collected in the matrices D_u, D_v and D_w) and eigenvectors (collected in the matrices V_u, V_v and V_w) the signals were built with the formulas:

$$n_u = rand_u \cdot \sqrt{D_u} \cdot {V_u}^T \qquad 6\text{-}45$$

$$n_v = rand_v \cdot \sqrt{D_v} \cdot {V_v}^T \qquad 6\text{-}46$$

$$n_w = rand_w \cdot \sqrt{D_w} \cdot {V_w}^T \qquad 6\text{-}47$$

where $rand$ is a vector of Gaussian random signals of size $[n, 20]$.

The random signals obtained n_u, n_v and n_w have the correlation corresponding to that described by the correlation tensor: the correlation between a signal i and a signal j is the same one that is found in the correlation matrix at position $(i, j) = (j, i)$.

In this way it is possible to build Gaussian random signals that respect the correlation. These signals can be used in the Dryden model of Turbulence.

In the calculation of the noise n_p for the angular velocity p_{turb} , the values in D_w and V_w with new random signals $rand_p$ have been used.

The computation was performed for both the region of low altitude (*LA*) and for that of high altitude (*HA*), obtaining respectively the following signals:

$$\begin{cases} n_u^{LA} \\ n_v^{LA} \\ n_w^{LA} \\ n_p^{LA} \end{cases} and \begin{cases} n_u^{HA} \\ n_v^{HA} \\ n_w^{HA} \\ n_p^{HA} \end{cases} \tag{6-48}$$

6.4.4.2 Limitations of the Procedure

Following the procedure presented above and building three new correlation matrices, whose elements are the correlations between the new pairs of signals, it is possible to make a comparison with the "original" correlation matrices R_u, R_v and R_w.

The result of the comparison is that the new matrices approximate well the original matrices; the error made is in fact very small, in any case less than 1%.

As an example, here it is an excerpt of the comparison between the matrix R_u and the correlation matrix of the velocity components u:

	1	2	3	4	5	6	7	8	9	10
1	1	0.9591	0.8819	0.8441	0.8079	0.7706	0.7446	0.7250	0.6962	0.6651
2	0.9591	1	0.9195	0.8803	0.8426	0.8038	0.7766	0.7560	0.7262	0.6938
3	0.8819	0.9195	1	0.9575	0.9166	0.8745	0.8454	0.8226	0.7909	0.7559
4	0.8441	0.8803	0.9575	1	0.9575	0.9136	0.8831	0.8588	0.8260	0.7897
5	0.8079	0.8426	0.9166	0.9575	1	0.9542	0.9222	0.8959	0.8624	0.8249
6	0.7706	0.8038	0.8745	0.9136	0.9542	1	0.9649	0.9358	0.9021	0.8638
7	0.7446	0.7766	0.8454	0.8831	0.9222	0.9649	1	0.9686	0.9349	0.8951
8	0.7250	0.7560	0.8226	0.8588	0.8959	0.9358	0.9686	1	0.9577	0.9184
9	0.6962	0.7262	0.7909	0.8260	0.8624	0.9021	0.9349	0.9577	1	0.9558
10	0.6651	0.6938	0.7559	0.7897	0.8249	0.8638	0.8951	0.9184	0.9558	1

Figure 6-8 R_u

	1	2	3	4	5	6	7	8	9	10
1	1.0036	0.9640	0.8861	0.8479	0.8103	0.7740	0.7490	0.7303	0.7011	0.6701
2	0.9640	1.0058	0.9246	0.8851	0.8463	0.8085	0.7820	0.7624	0.7328	0.7009
3	0.8861	0.9246	1.0054	0.9626	0.9205	0.8785	0.8506	0.8287	0.7970	0.7632
4	0.8479	0.8851	0.9626	1.0043	0.9604	0.9173	0.8878	0.8648	0.8318	0.7963
5	0.8103	0.8463	0.9205	0.9604	1.0016	0.9567	0.9258	0.9010	0.8677	0.8311
6	0.7740	0.8085	0.8785	0.9173	0.9567	1.0027	0.9685	0.9407	0.9075	0.8706
7	0.7490	0.7820	0.8506	0.8878	0.9258	0.9685	1.0050	0.9746	0.9409	0.9025
8	0.7303	0.7624	0.8287	0.8648	0.9010	0.9407	0.9746	1.0059	0.9643	0.9264
9	0.7011	0.7328	0.7970	0.8318	0.8677	0.9075	0.9409	0.9643	1.0059	0.9625
10	0.6701	0.7009	0.7632	0.7963	0.8311	0.8706	0.9025	0.9264	0.9625	1.0079

Figure 6-9 Correlation matrix velocity components u

Despite the result can be described as excellent in MATLAB®, the result obtained in Simulink® is not so accurate.

In Simulink®, in fact, there is not a block that allows the calculation of the eigenvalues and eigenvectors. The same procedure carried out in MATLAB®, was re-created in Simulink® using the Embedded Matlab Functions.

So, by inserting in them the same calculation codes used in MATLAB®, the result is not as accurate as in the previous case but, in any case, the error remains small enough and always less than 5%.

The difference is probably due to the methods used in the two cases to compute eigenvalues and eigenvectors.

Further tests are listed in sections 6.7 and 6.8.

6.4.4.3 Algorithm for Implementation

In the implemented system, the equations 6-2 to 6-23 and 6-44 to 6-47 are used.

6.4.5 Integration with other systems

To use the correlation model, it is necessary to integrate it with other systems.

In fact, this model allows to evaluate the correlation between the velocities of twenty aircraft and therefore requires, as input, the data of all aircraft.

Similarly, the outputs of the correlation model contain the data necessary to all aircraft and they must be separated so that a single value corresponds to each airplane.

In order to take account of these needs, two Embedded Matlab Function were used:

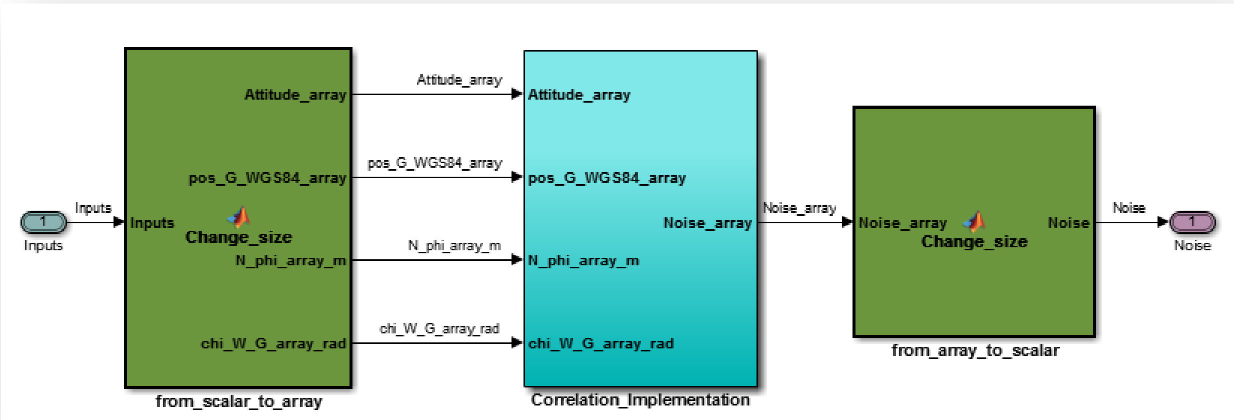


Figure 6-10 Correlation and Embedded Matlab Functions

In the first, starting from the data of each single aircraft, the inputs necessary for the operation of the correlation model are obtained.

First of all, for each aircraft it is possible to build the following bus:

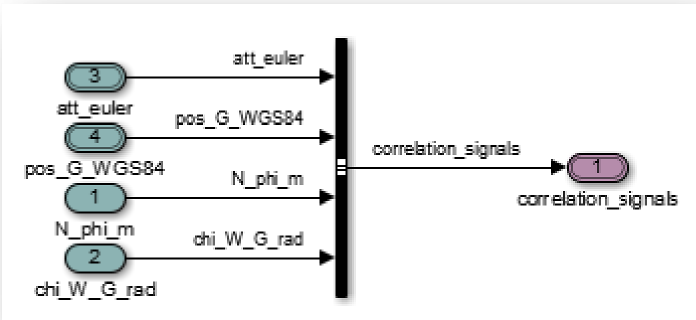


Figure 6-11 Correlation_signals_Bus

Combining these twenty buses via the Simulink block *Vector Concatenate*, it is possible to build the *Inputs* signal:

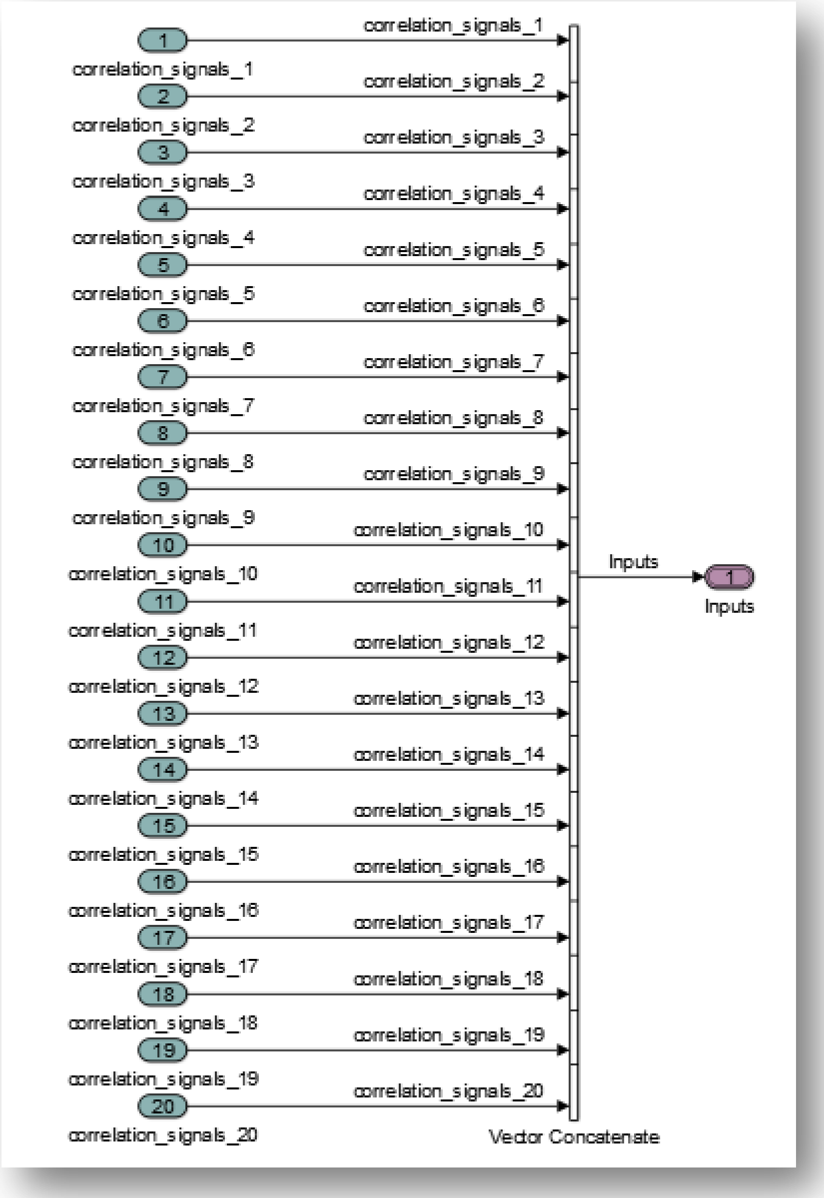


Figure 6-12 Vector_Concatenate

This Embedded Matlab Function is used to reconstruct the arrays that contain data of all twenty aircraft:

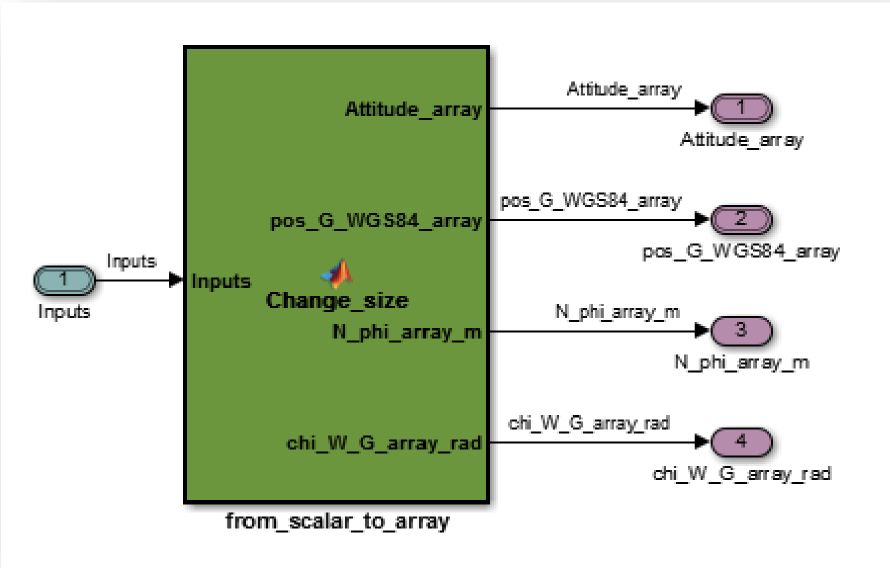


Figure 6-13 Embedded Matlab Function: from_scalar_to_array

For verifications see Appendix A.

Seen that the output of the correlation model is constituted by noises in the form of arrays, relative to all twenty planes, the second Embedded Matlab Function allows to separate them in order to obtain the signals for the individual aircraft.

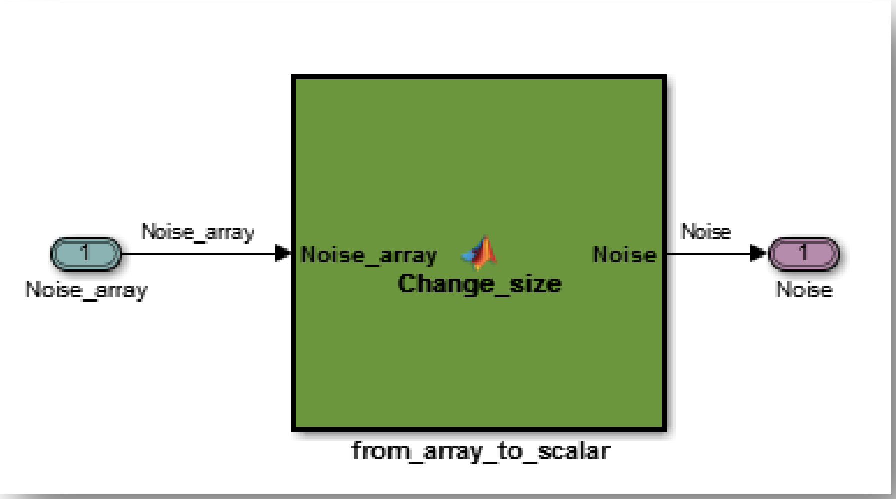


Figure 6-14 Embedded Matlab Function: from_array_to_scalar

Then, using Simulink blocks *Selector*, it is possible to select the *Noise_Bus* for each airplane. For verifications see Appendix A.

6.5 Architecture Specification

6.5.1 Parent / Child Systems

6.5.1.1 Parent System

The system is a top level system.

6.5.1.2 Child Systems

This system does not contain any child systems.

6.5.2 Signal Definitions

6.5.2.1 Inputs

Inputs							
Symbol	**Name**	**Size**	**Components**	**Data Type**	**Min**	**Max**	**Description**
Φ_{array}	Phi_array_rad	[1 20]	-	double	-pi	pi	Array containing the roll angles of twenty aircraft.
Θ_{array}	Theta_array_rad	[1 20]	-	double	-pi/2	pi/2	Array containing the pitch angles of twenty aircraft.
Ψ_{array}	Psi_array_rad	[1 20]	-	double	-pi	pi	Array containing the yaw angles of twenty aircraft.
φ^{G}_{array}	phi_G_WGS84_array_rad	[1 20]	-	double	-pi/2	pi/2	Array containing the Geodetic Latitudes of twenty aircraft.
λ^{G}_{array}	lambda_G_WGS84_rad	[1 20]	-	double	-pi	pi	Array containing the Geodetic Longitudes of twenty aircraft.
h^{G}_{array}	h_G_WGS84_array_rad	[1 20]	-	double	-500	20000	Array containing the Geodetic Altitudes of twenty aircraft.
$N_{\varphi_{array}}$	N_phi_array_m	[1 20]	-	double	0	Inf	Array containing the Earth's radii of twenty aircraft

Table 6-7 Inputs

6.5.2.2 Outputs

Outputs							
Symbol	**Name**	**Size**	**Components**	**Data Type**	**Min**	**Max**	**Description**
$n^{HA}_{u_{array}}$	n_u_HA_array	[1 20]	-	double	-	-	Array containing the signals needed to calculate the component of velocity u_{Turb} in High Altitude
$n^{HA}_{v_{array}}$	n_v_HA_array	[1 20]	-	double	-	-	Array containing the signals needed to calculate the component of velocity v_{Turb} in High Altitude
$n^{HA}_{w_{array}}$	n_w_HA_array	[1 20]	-	double	-	-	Array containing the signals needed to calculate the component of velocity w_{Turb} in High Altitude
$n^{HA}_{p_{array}}$	n_p_HA_array	[1 20]	-	double	-	-	Array containing the signals needed to calculate the component of angular velocity p_{Turb} in High Altitude
$n^{LA}_{u_{array}}$	n_u_LA_array	[1 20]	-	double	-	-	Array containing the signals needed to calculate the component of velocity u_{Turb} in Low Altitude
$n^{LA}_{v_{array}}$	n_v_LA_array	[1 20]	-	double	-	-	Array containing the signals needed to calculate the component of velocity v_{Turb} in Low Altitude
$n^{LA}_{w_{array}}$	n_w_LA_array	[1 20]	-	double	-	-	Array containing the signals needed to calculate the component of velocity w_{Turb} in Low Altitude
$n^{LA}_{p_{array}}$	n_p_LA_array	[1 20]	-	double	-	-	Array containing the signals needed to calculate the component of angular velocity p_{Turb} in Low Altitude

Table 6-8 Outputs

6.5.3 Bus Structure

<u>Inputs</u>

L	Bus Name	Elements	Element Types
0	*pos_G_WGS84_array_Bus*	phi_G_WGS84_array_rad lambda_G_WGS84_array_rad h_G_WGS84_array_m	double double double
0	*Attitude_array_Bus*	Phi_array_rad Theta_array_rad Psi_array_rad	double double double

Table 6-9 Inputs Bus Structure

<u>Outputs</u>

L	Bus Name	Elements	Element Types
0	*Noise_array_Bus*	Noise_HA_array_Bus Noise_LA_array_Bus	Noise_HA_array_Bus Noise_LA_array_Bus
1	*Noise_HA_array_Bus*	n_u_HA_array n_v_HA_array n_w_HA_array n_p_HA_array	double double double double
1	*Noise_HA_array_Bus*	n_u_LA_array n_v_LA_array n_w_LA_array n_p_LA_array	double double double double

Table 6-10 Outputs Bus Structure

6.6 Structural Layout

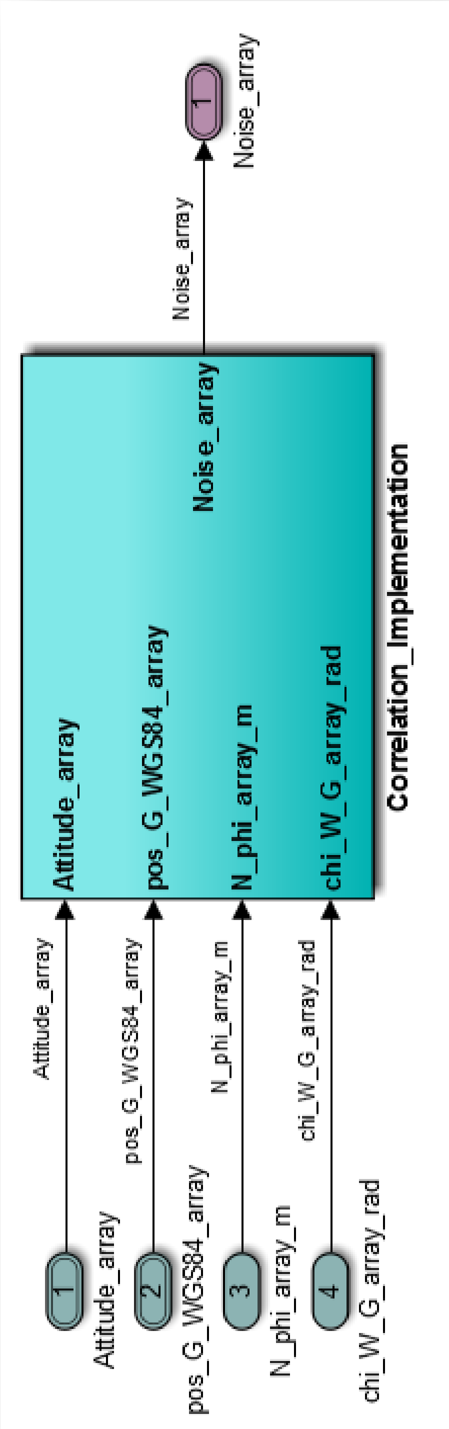


Figure 6-15 L0: ENV_CORR

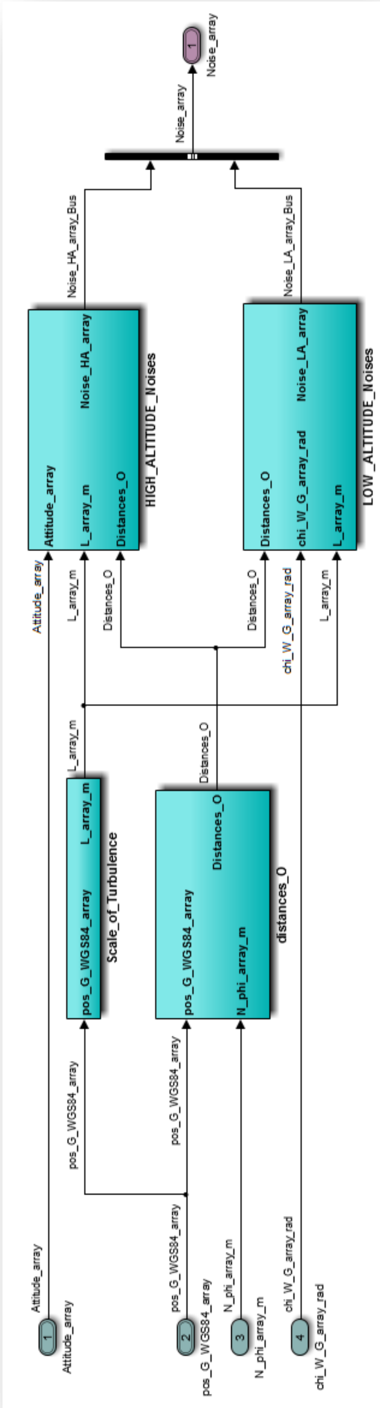


Figure 6-16 L1: Correlation_Implementation

Figure 6-17 L2: Scale_of_Turbulence

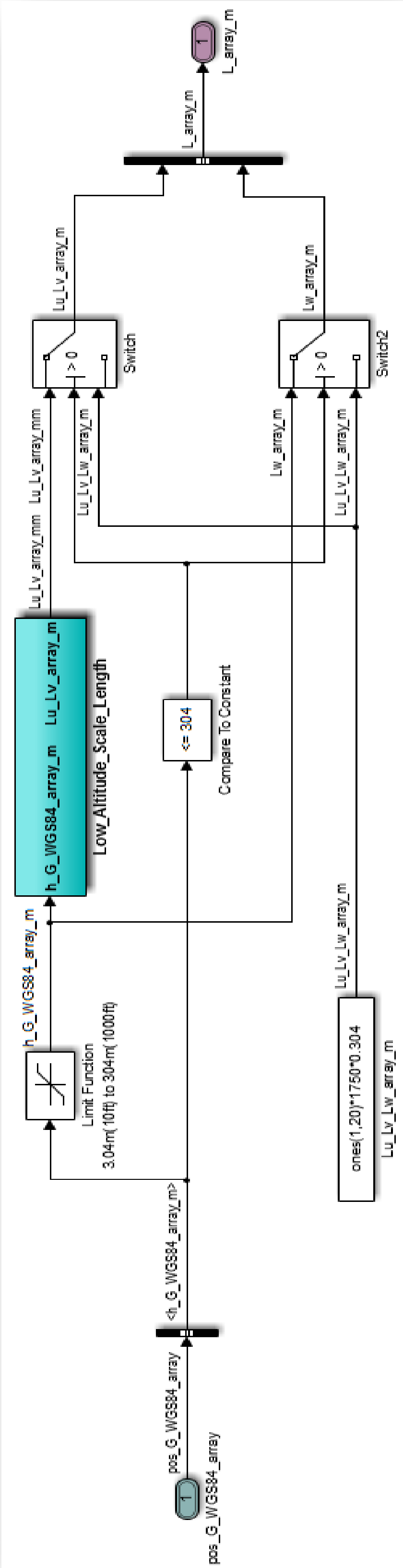


Figure 6-18 L2: HIGH_ALTITUDE_Noises

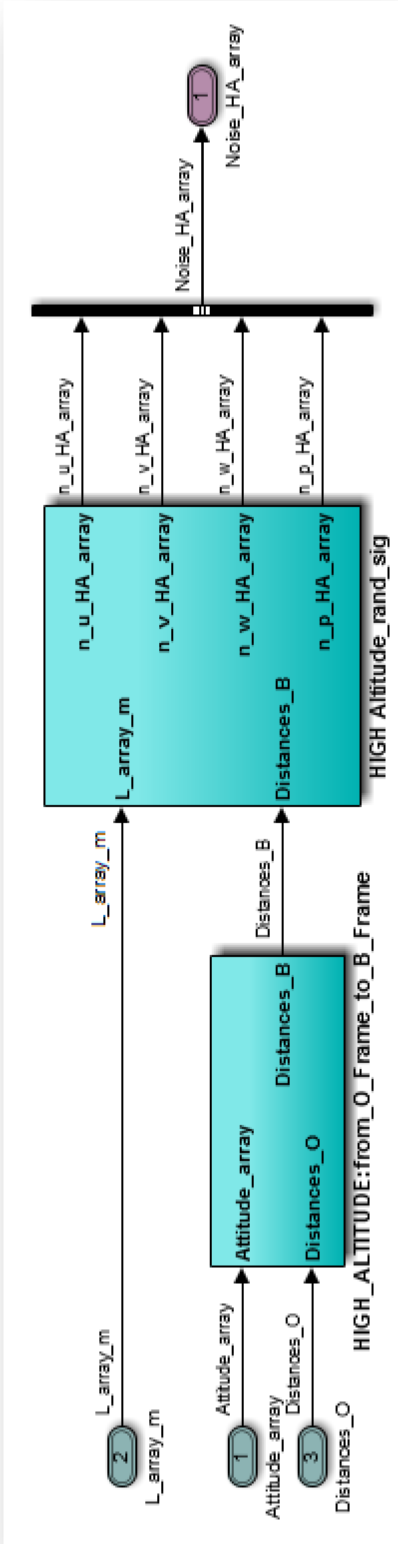


Figure 6-19 L2: LOW_ALTITUDE_Noises

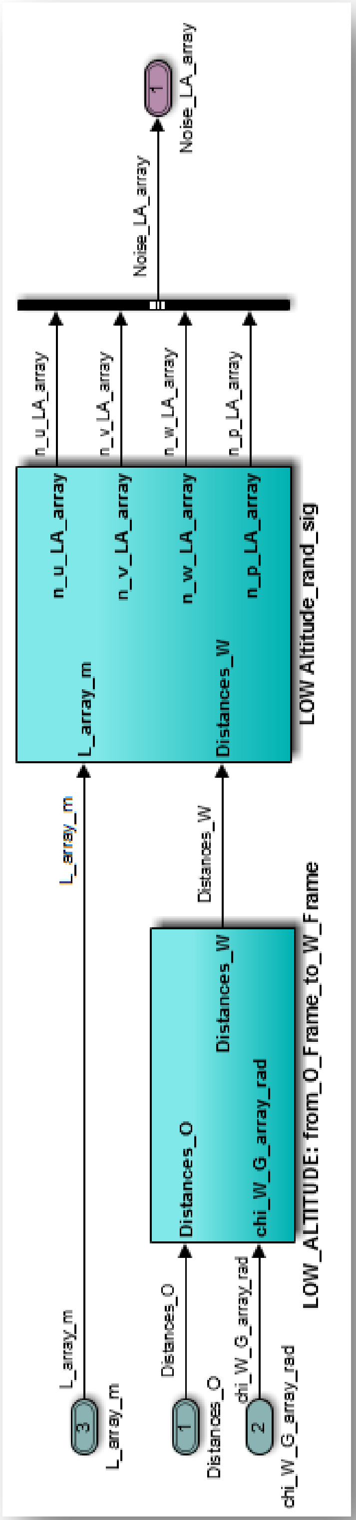
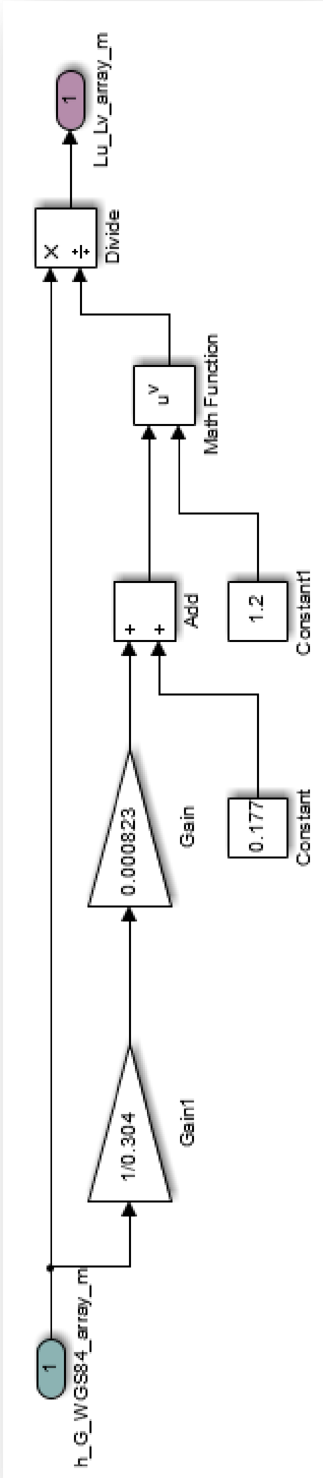


Figure 6-20 L3: Low_Altitude_Scale_Length



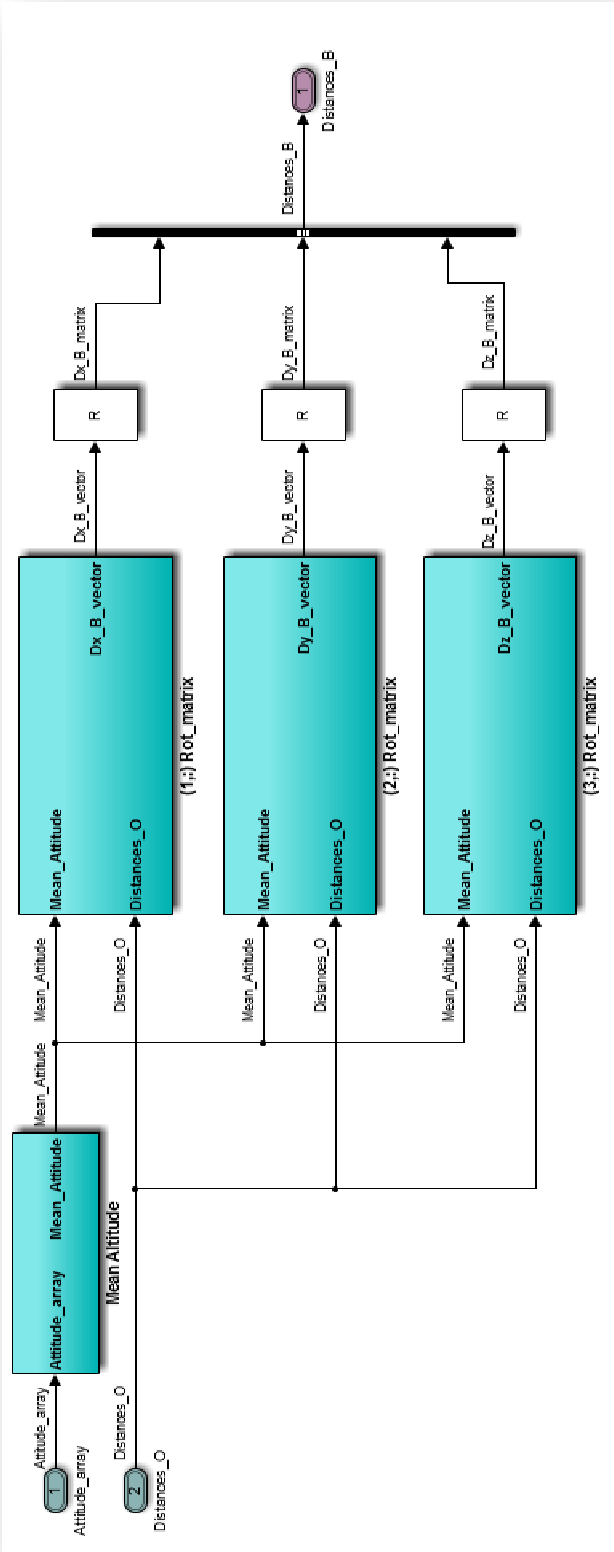


Figure 6-21 L3: HIGH ALTITUDE: from_O_Frame_to_B_Frame

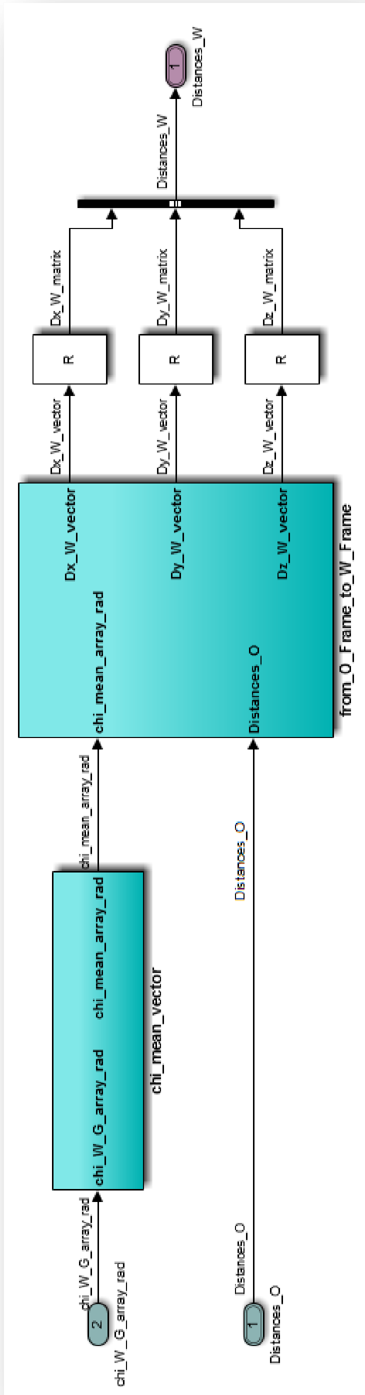


Figure 6-22 L3: LOW_ALTITUDE_from_O_Frame_to_W_Frame

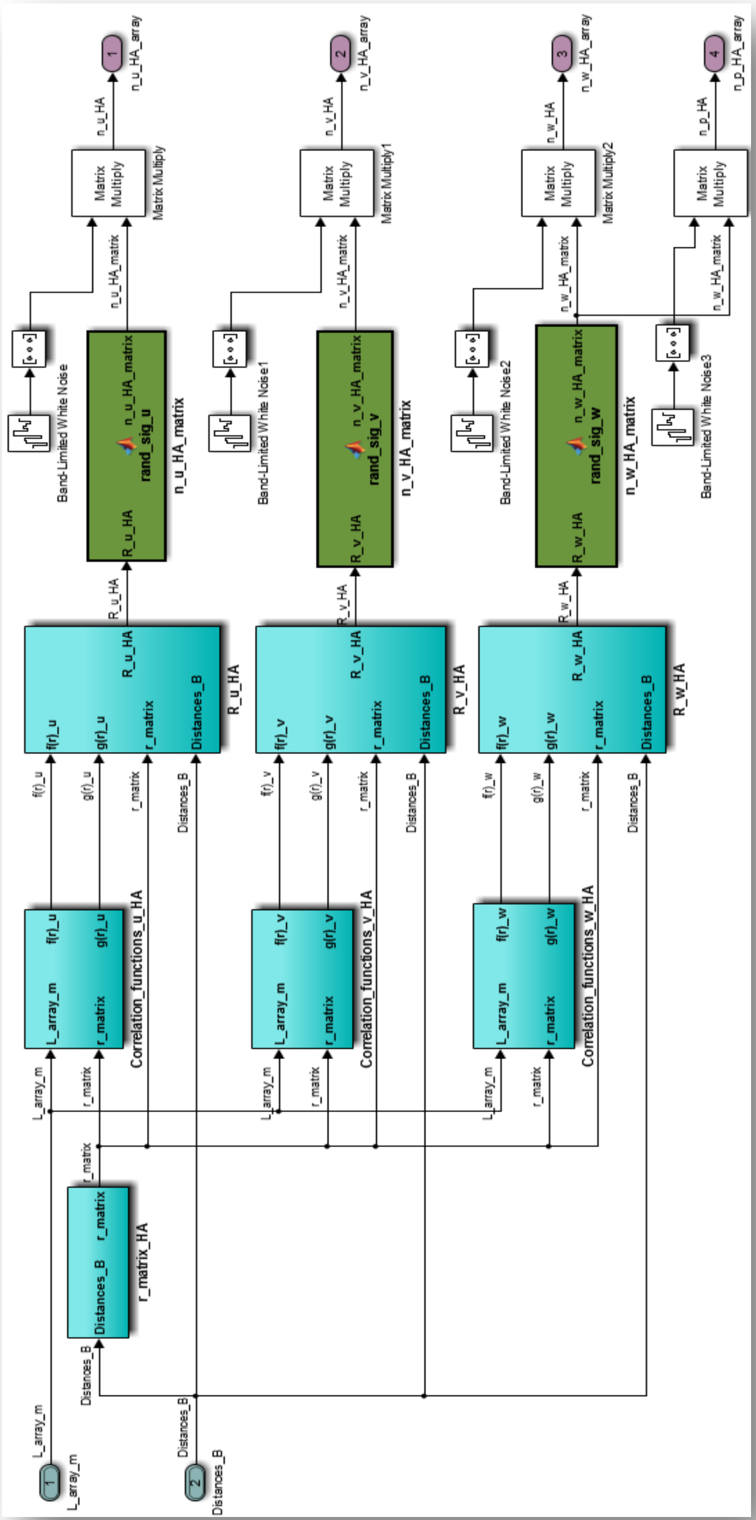


Figure 6-23 L3: HIGH ALTITUDE_rand_sig

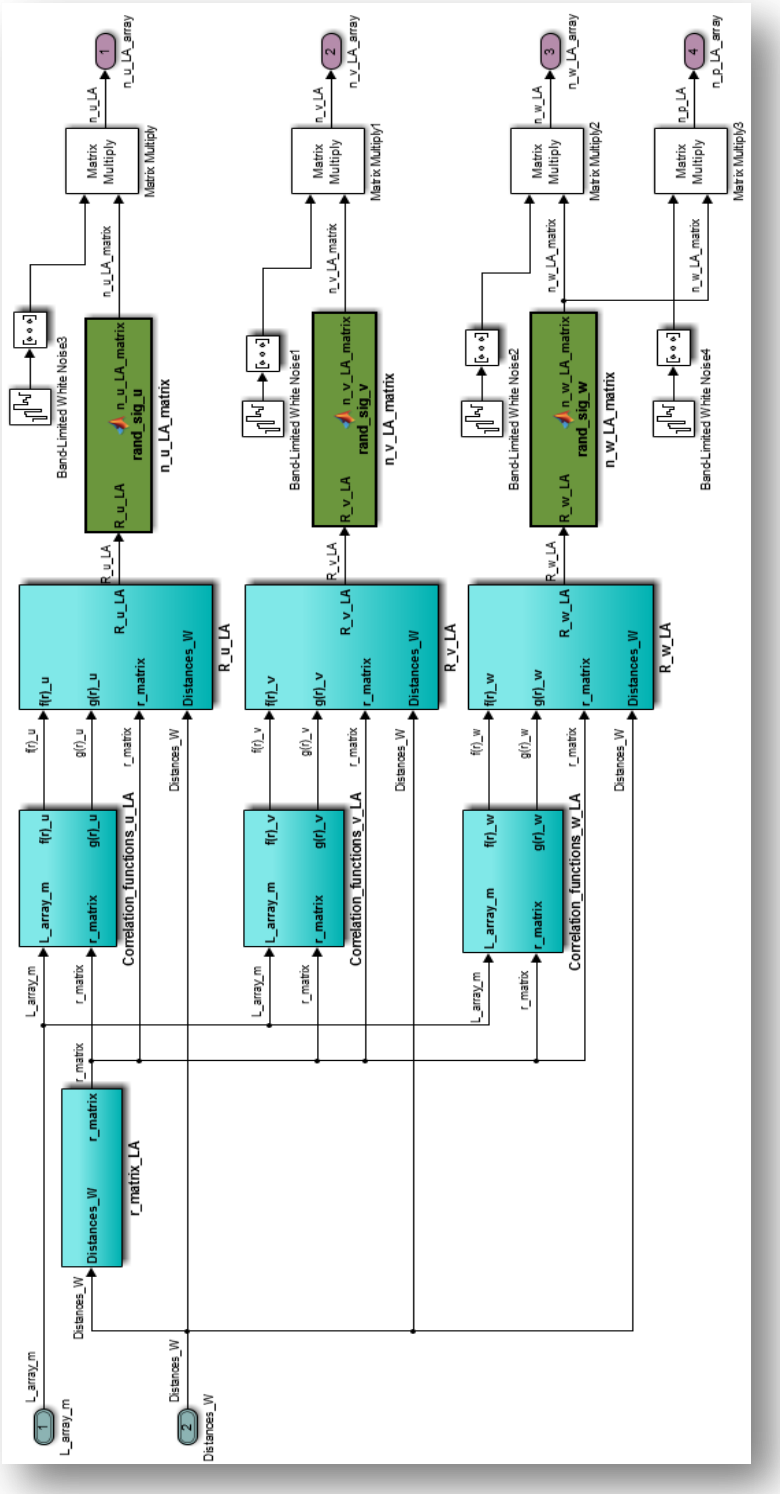


Figure 6-24 L3: LOW_ALTITUDE_rand_sig

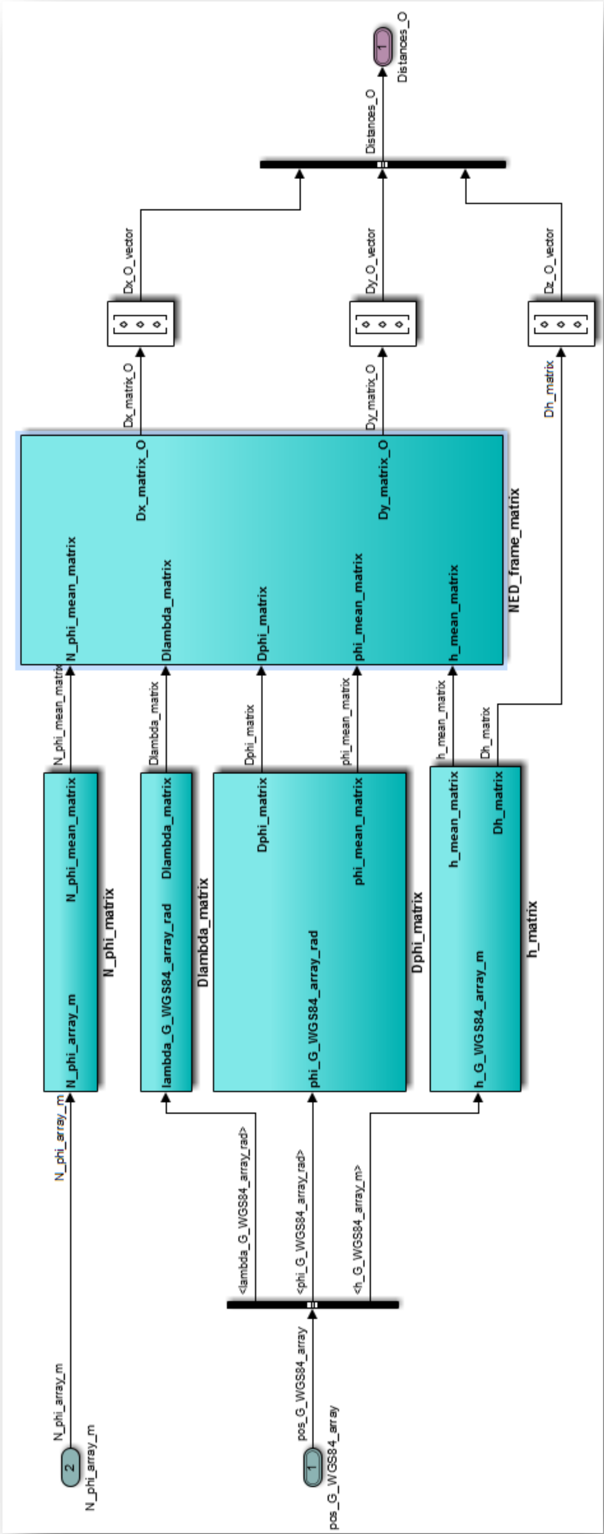


Figure 6-25 L2: distances_O

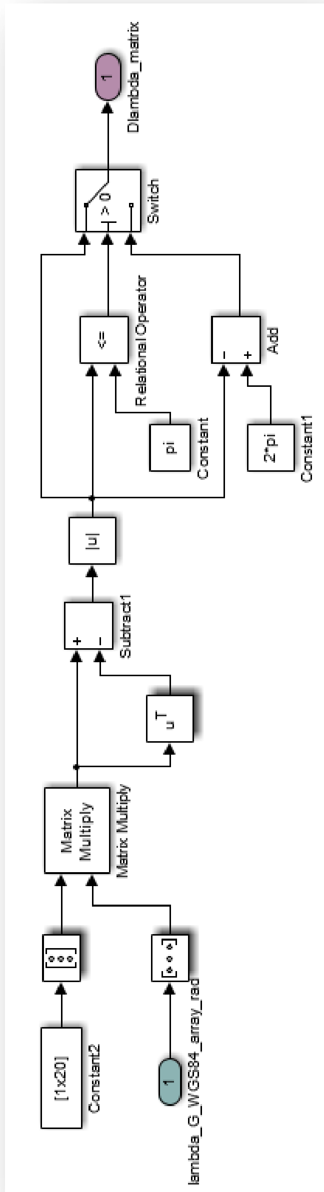


Figure 6-26 L3: Dlambda_matrix

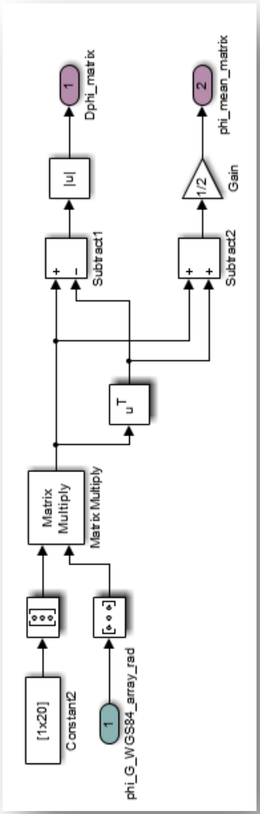


Figure 6-27 L3: Dphi_matrix

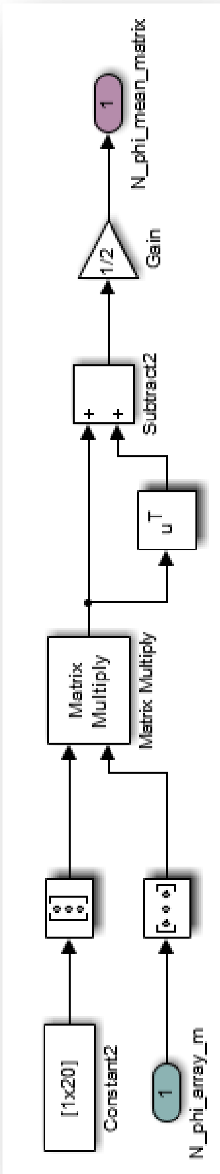


Figure 6-28 L3: N_phi_matrix

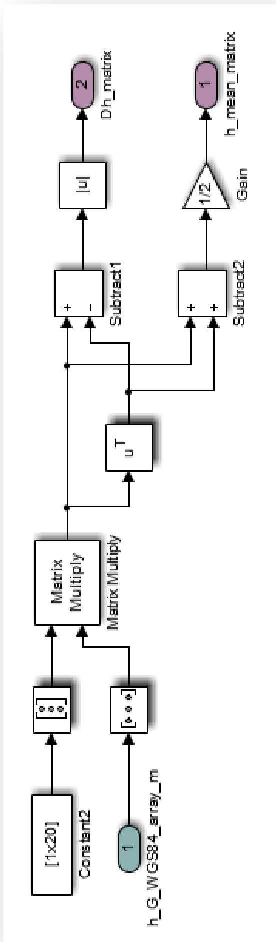


Figure 6-29 L3: h_matrix

Figure 6-30 L3: NED_frame_matrix

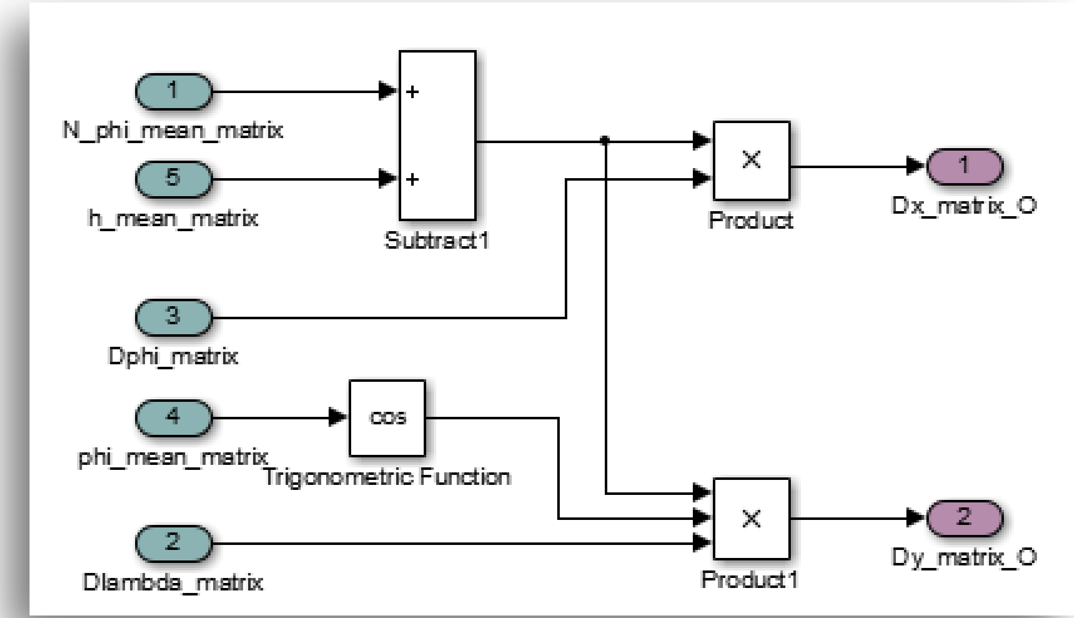
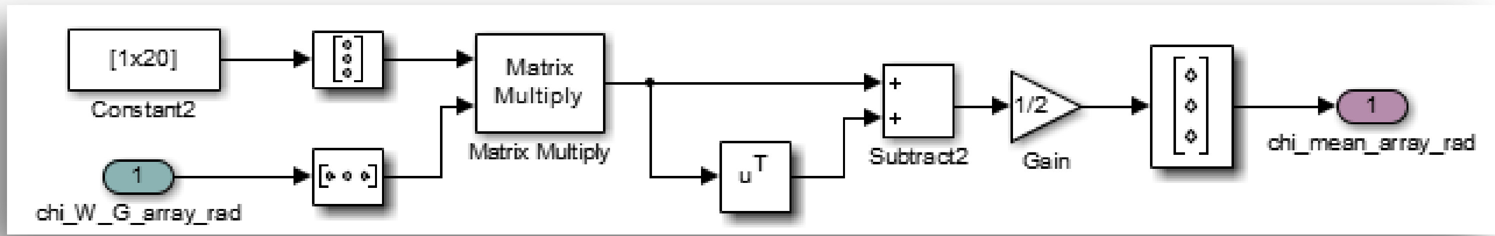


Figure 6-31 L4: chi_mean_vector



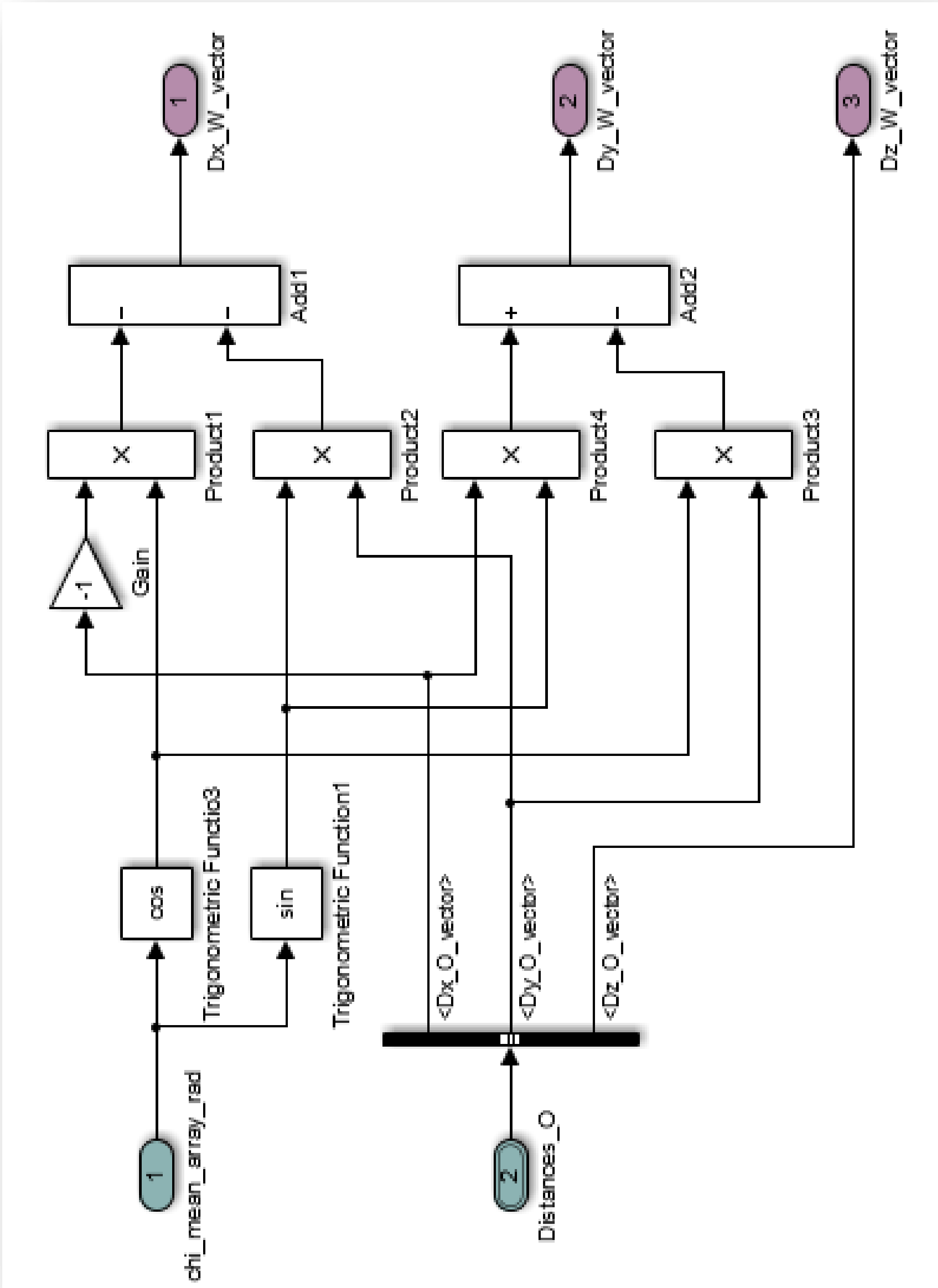


Figure 6-32 L4: from_O_frame_to_W_frame

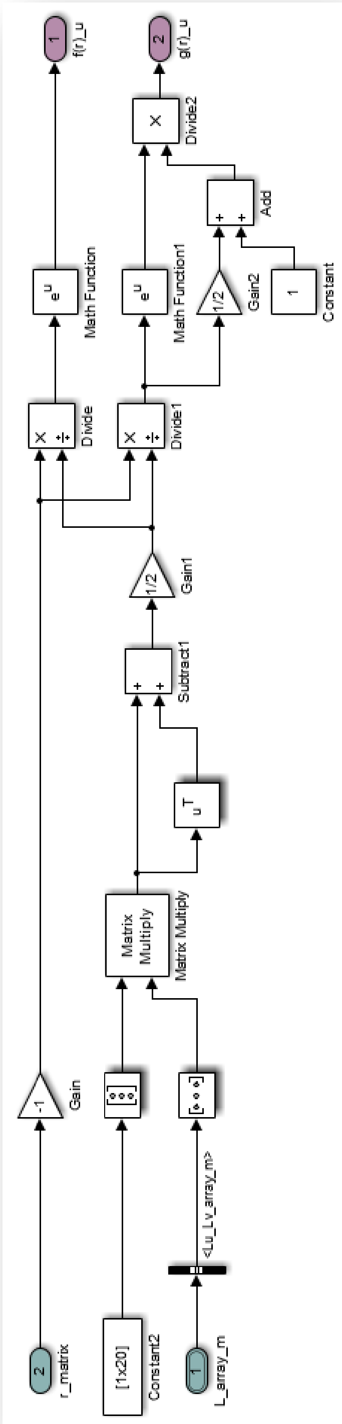


Figure 6-33 L4: Correlation_Function_u_HA

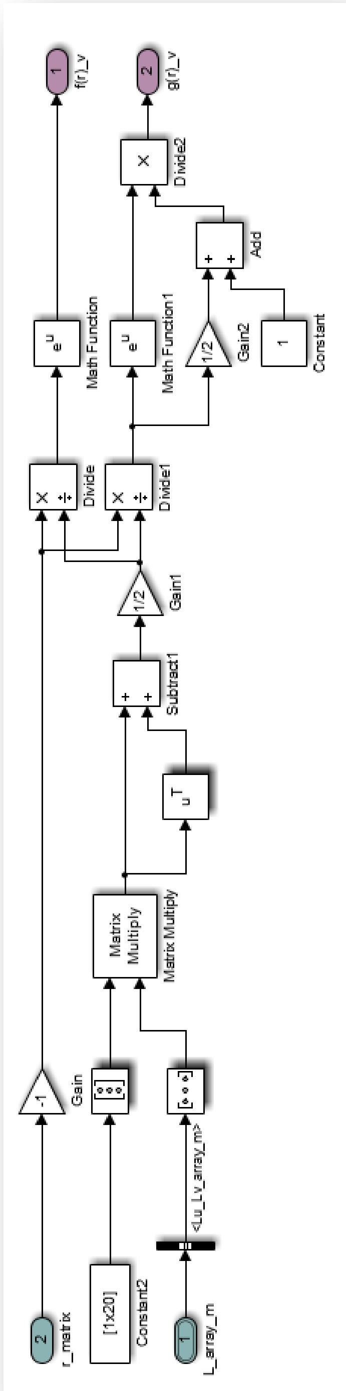


Figure 6-34 L4: Correlation_Function_v_HA

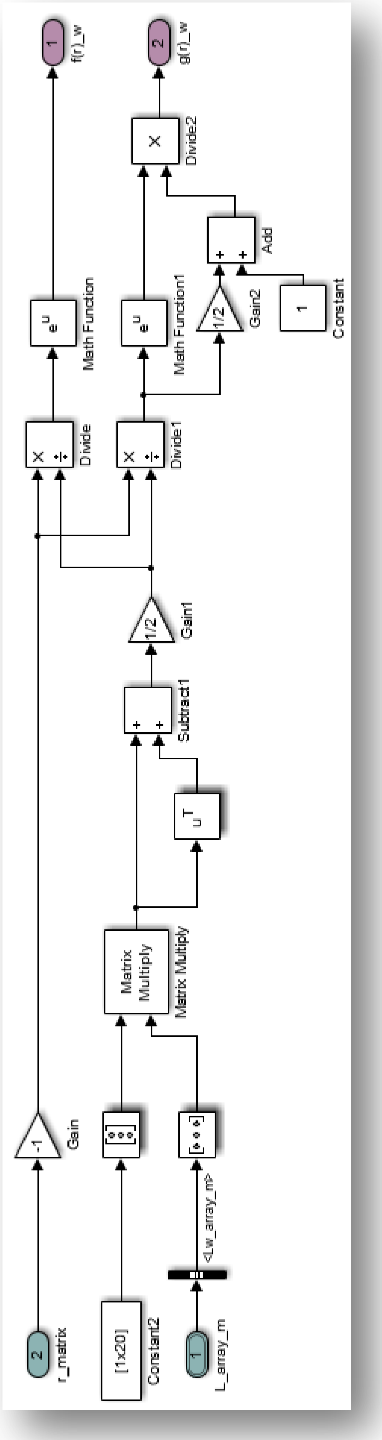


Figure 6-35 L4: Correlation_Function_w_HA

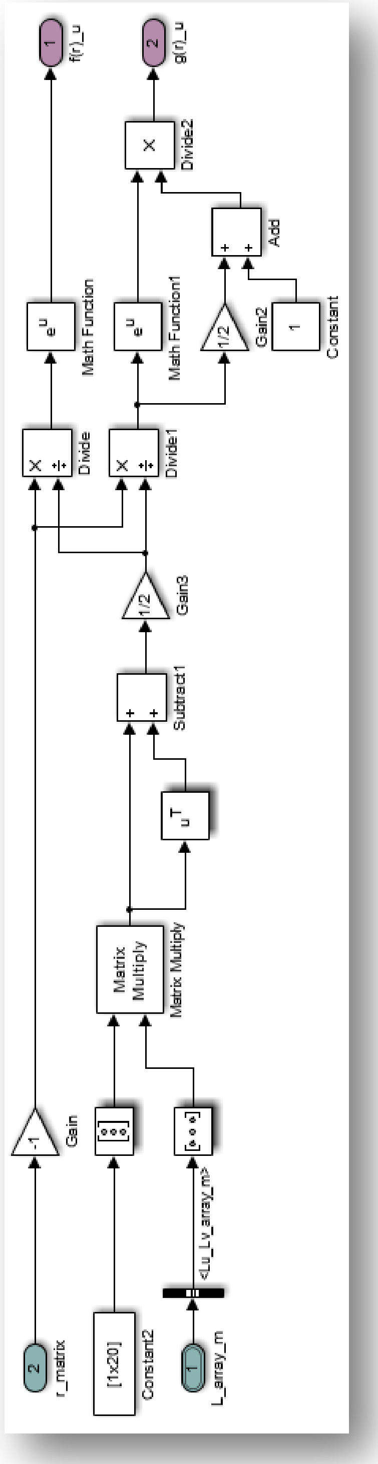


Figure 6-36 L4: Correlation_Function_u_LA

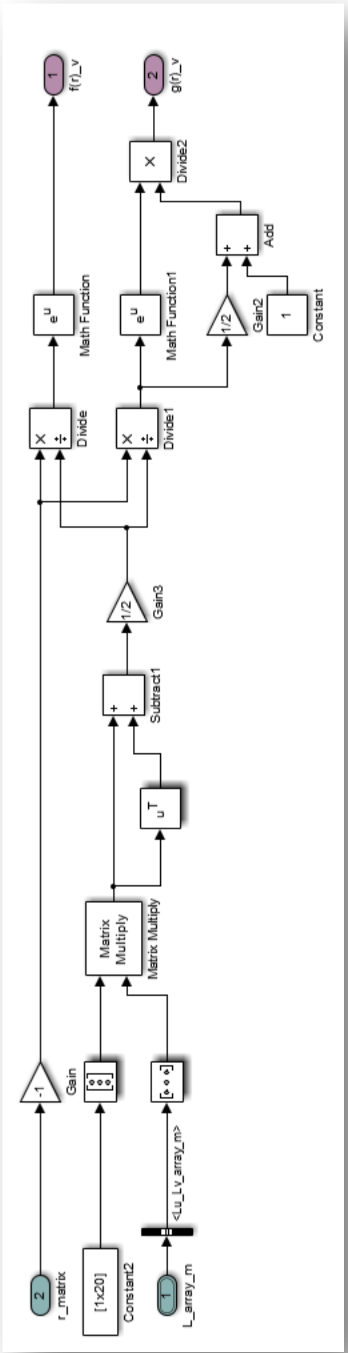


Figure 6-37 L4: Correlation_Function_v_LA

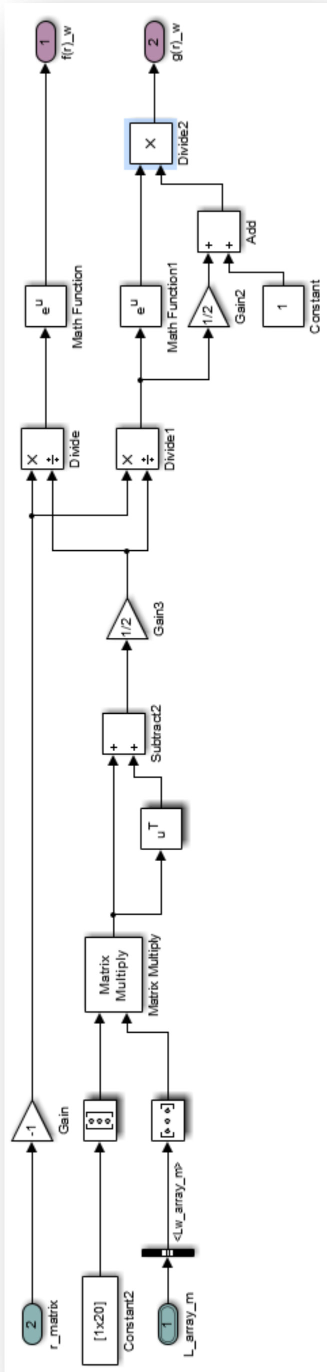


Figure 6-38 L4: Correlation_Function_w_LA

Figure 6-39 L4: R_u_HA

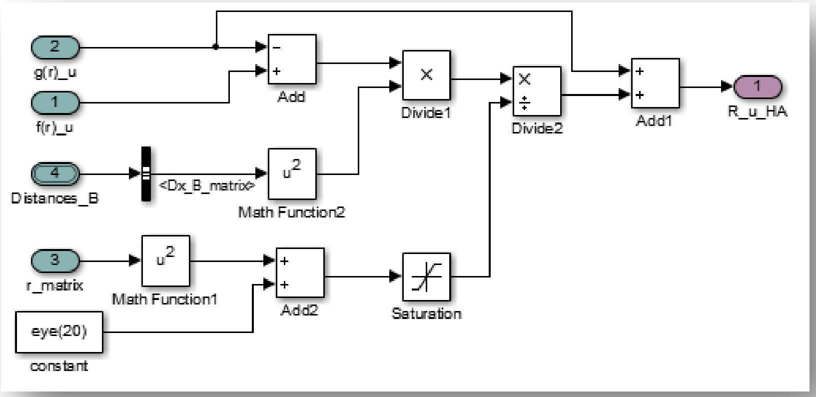


Figure 6-40 L4: R_v_HA

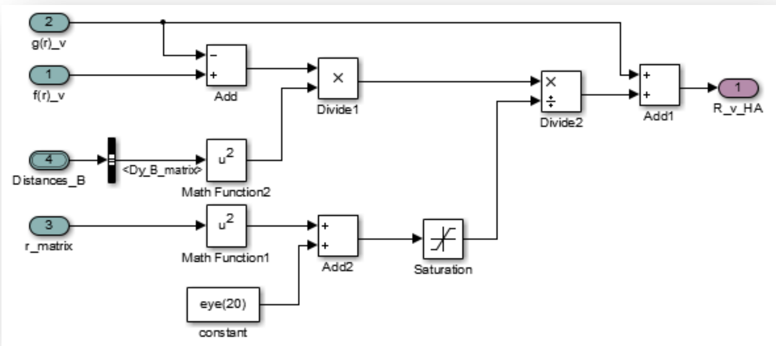


Figure 6-41 L4: R_w_HA

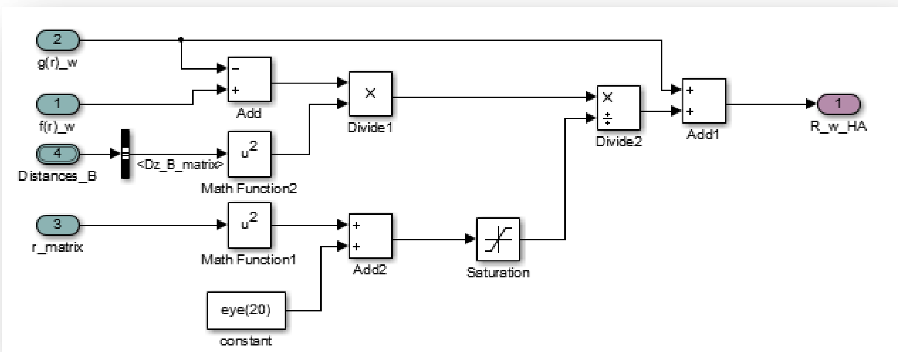


Figure 6-42 L4: R_u_LA

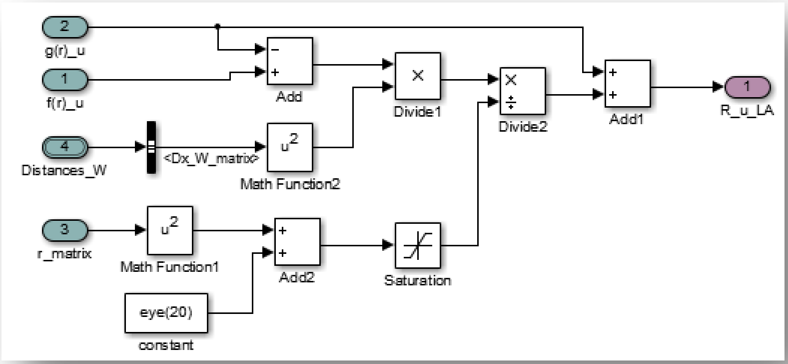


Figure 6-43 L4: R_v_LA

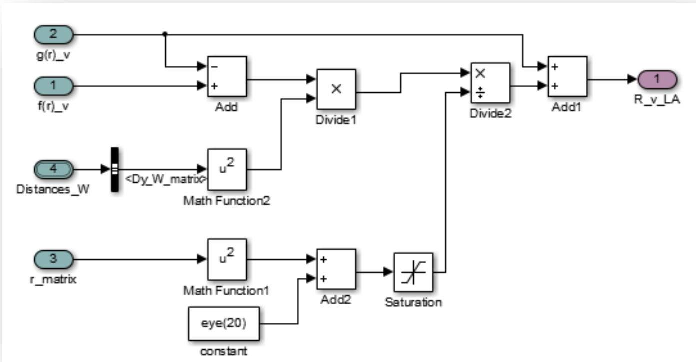


Figure 6-44 L4: R_w_LA

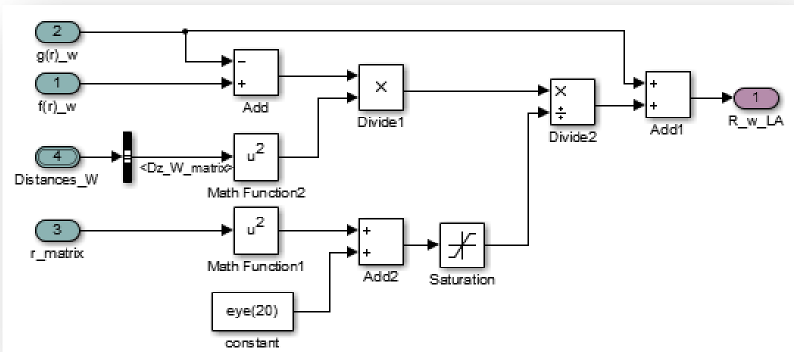


Figure 6-45 L4: Rot_matrix(1,:)

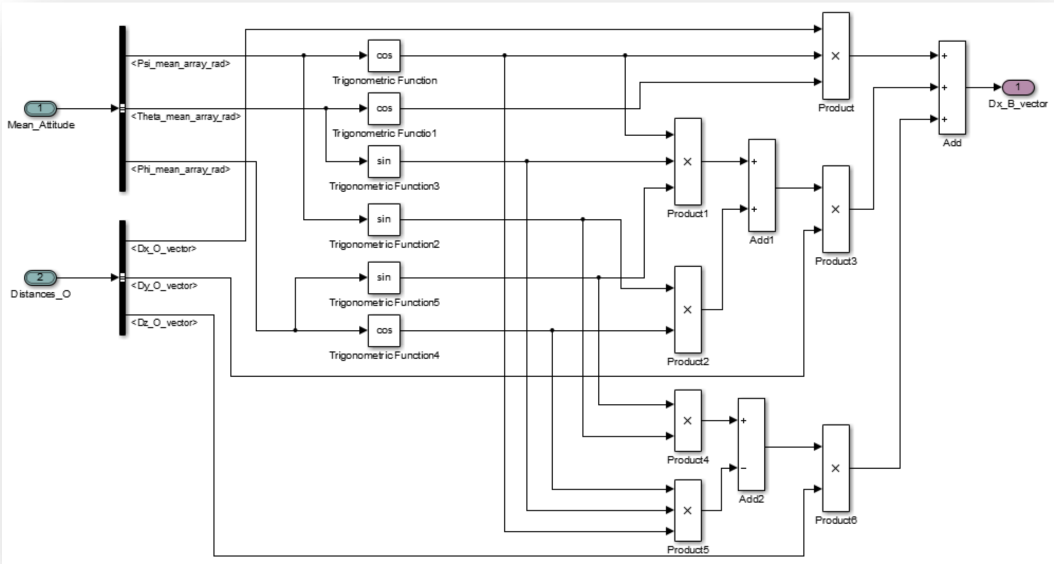


Figure 6-46 L4: Rot_matrix(2,:)

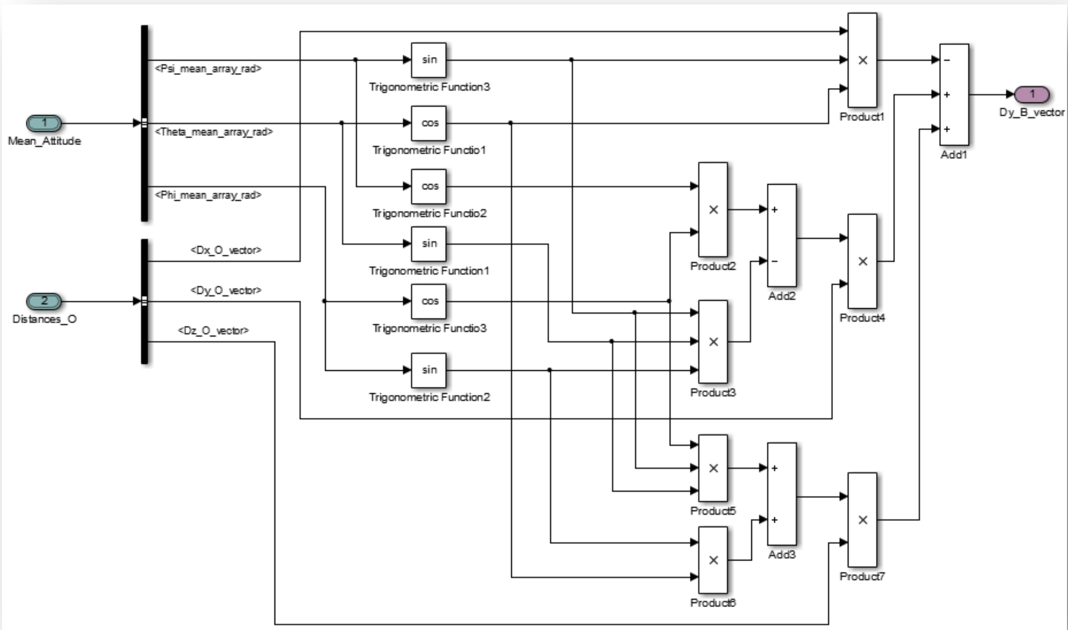


Figure 6-47 L4: Rot_matrix(3,:)

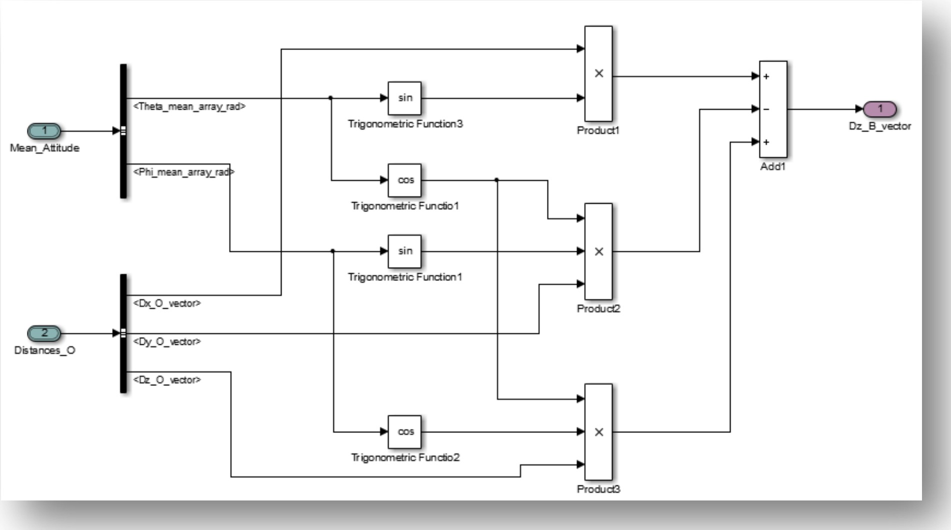


Figure 6-48 L4: r_matrix_HA

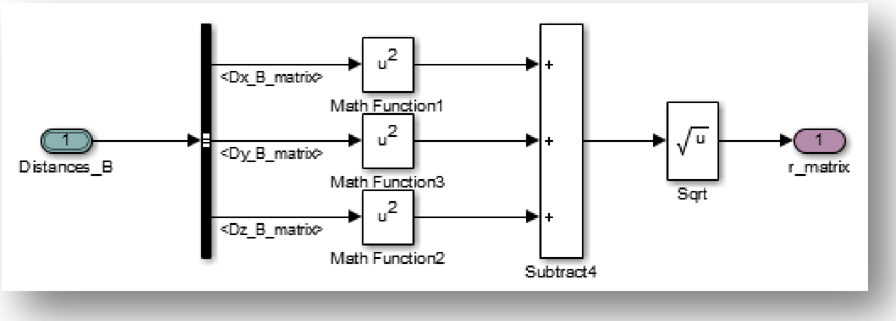


Figure 6-49 L4: r_matrix_LA

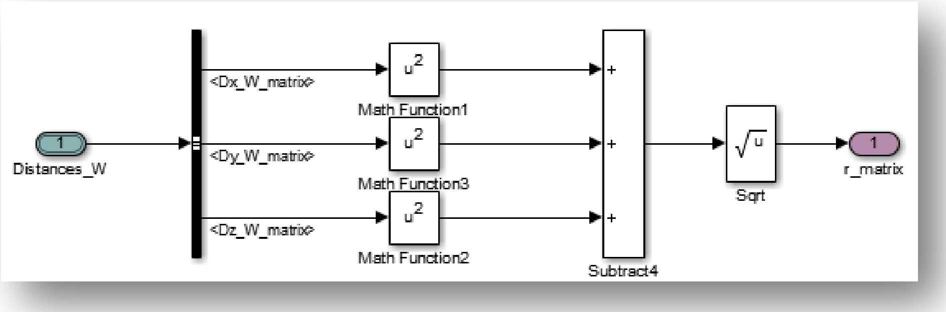


Figure 6-50 L4: Mean_Attitude

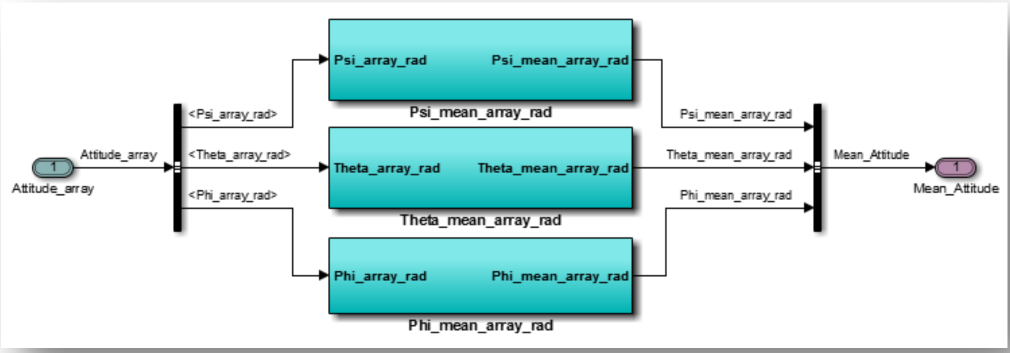


Figure 6-51 L5: Psi_mean_array_rad

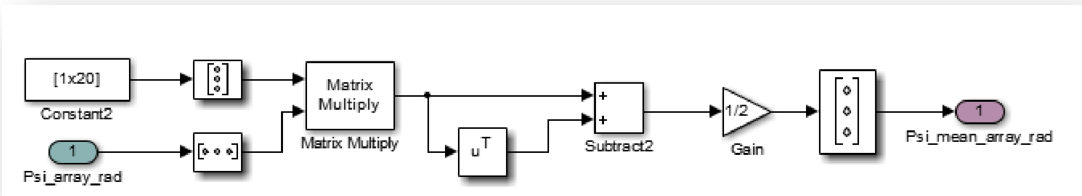


Figure 6-52 L5: Phi_mean_array_rad

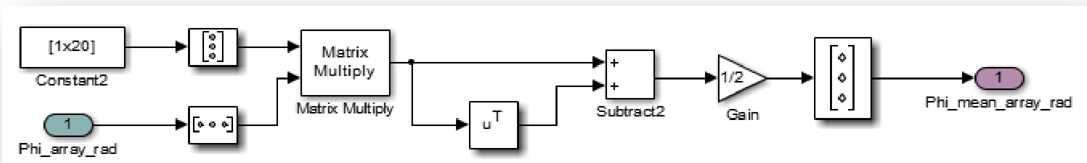
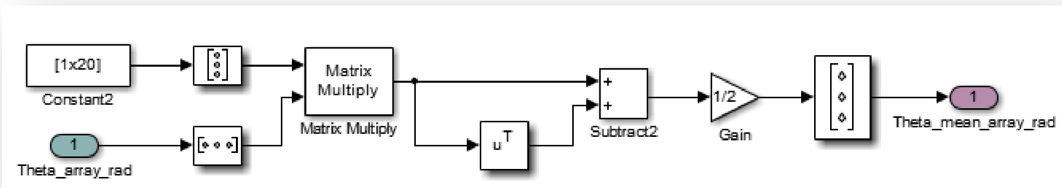


Figure 6-53 L5: Theta_mean_array_rad



6.7 Verification Plan

6.7.1 Methods Used for Verification

6.7.1.1 Methods for Testing Functional Requirements

Requirement Name and ID	Description of Verification Method
R-FUN-ENV_CORR Computation of Signals	Correct calculation of correlated signals demonstrated in ENV_CORR-Nominal_TC1 comparison to dissimilar implementation in ENV_CORR-Nominal_TC2 (see section 6.7.2.1).

Table 6-11 Methods for Testing Functional Requirements

6.7.1.2 Methods for Testing Implementation Requirements

Requirement Name and ID	Description of Verification Method
R-NUM-ENV_CORR Numeric Efficiency	Manual review of the implemented model. Records according to section 6.8.1.1.
R-IOC-ENV_CORR Input / Output Interface Compliance to Parent System	Compliance to parent system will be verified at integration with parent system.
R-SGC-ENV_CORR Implementation Compliance to FSD Style Guides	Manual review of the implemented model. Records according to section 6.8.1.2.
R-ISC-ENV_CORR Implementation Standards Compliance	Manual review of the implemented model. Records according to section 6.8.1.3.

Table 6-12 Methods for Testing Implementation Requirements

6.7.1.3 Methods for Testing Operational Requirements

Derived Standard Requirement Matrix:		
SIMULINK Modes	Simulink Offline Simulation	Demonstrated during test ENV_CORR-Nominal_TC1, ENV_CORR-Nominal_TC2 (see sections 6.7.2.1).
	Simulink Pseudo Real Time Simulation	Will be demonstrated on a higher level of the simulation model.
External Control for SIMULINK Execution	Bypassing of Non-Autonomous Elements	Not applicable as there are no non-autonomous elements present in the model.
	Single Point Execution	Demonstrated during test ENV_CORR-Nominal_TC1, ENV_CORR-Nominal_TC2 (see sections 6.7.2.1).
	Online Integration Freeze and Reset	Not applicable as there are no integrators present in the model.
	Workspace Initialization	Not applicable as there are no variables present to be initialized in the workspace.
	Runtime Parameter Tuning	Not applicable as there are no tunable parameters present in the model
RTW Code Generation	S-function	Generation of S-function demonstrated during test ENV_CORR-Operational_TC1 (see section 6.7.3.1).
Code Modes	Stand-alone Batch Simulation	Will be demonstrated on a higher level of the simulation model.
	Stand-alone Real Time Simulation	Will be demonstrated on a higher level of the simulation model.

Table 6-13 Methods for Testing Operational Requirements

6.7.2 Verification Plan for Functional Requirements

6.7.2.1 Nominal Testing Procedure

Test Name:	**Correct Calculation of Correlated Signals**
Test ID:	ENV_CORR-Nominal_TC1
Related Requirements:	R-FUN-ENV_CORR: Computation of Signals
Verification Data:	See section 6.8.2.1

The implemented simulation model is excited with different input combinations. Afterwards, the correlation is reviewed manually.

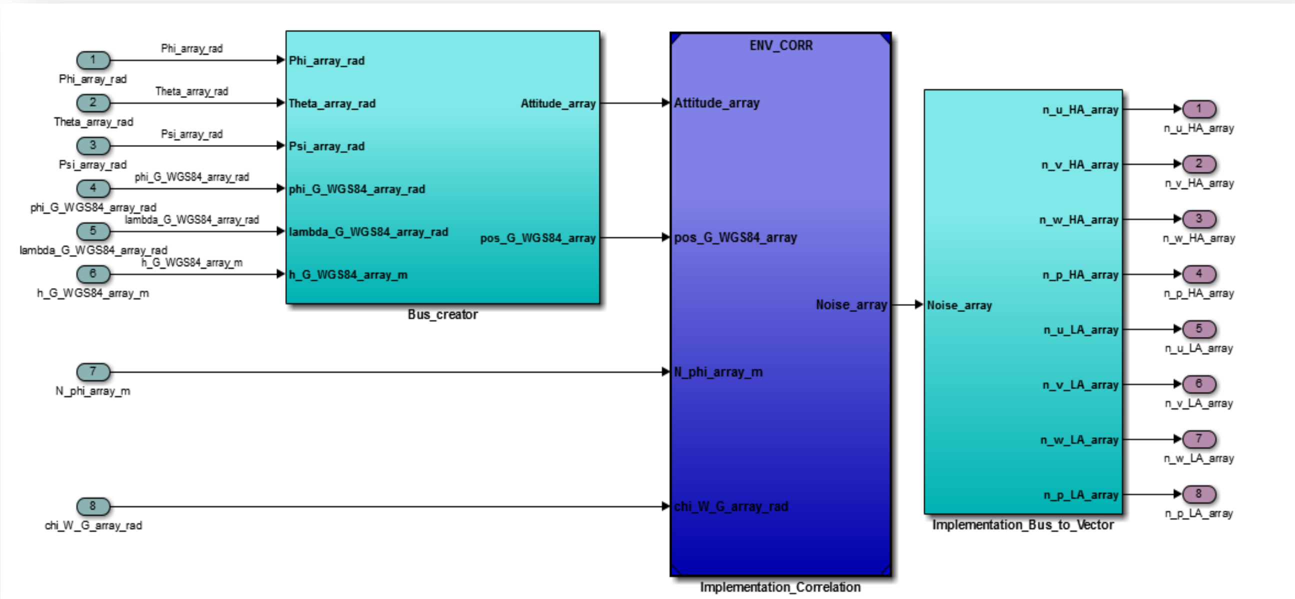


Figure 6-54 ENV_CORR-Nominal_TC1

In order to take into account the correlation between the velocities, the variables are varied in such a way that the twenty aircraft are next to each other.

Test Name:	**Equivalence of Implementation and Dissimilar Implementation**
Test ID:	ENV_CORR-Nominal_TC2
Related Requirements:	R-FUN-ENV_CORR: Computation of Signals
Verification Data:	See section 6.8.2.1

The implemented model and a dissimilar implementation (Matlab script) are both excited with the same input signals. Afterwards, the correlation matrices are checked for deviations. As long as the absolute deviations stay below a certain threshold both implementations are considered equivalent and the test is passed.

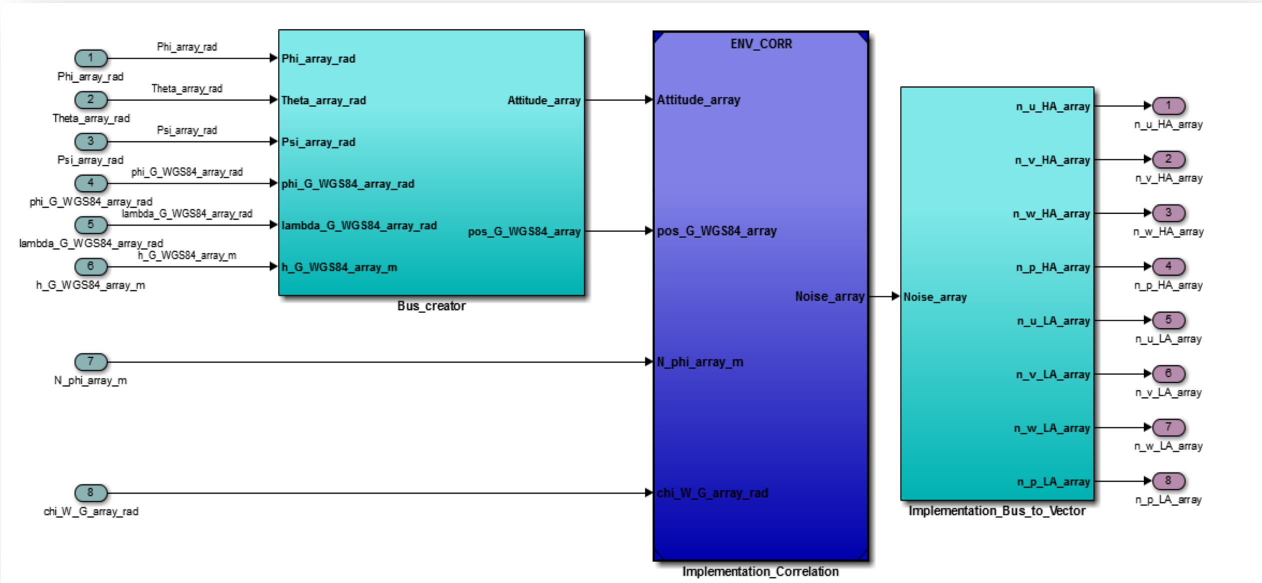


Figure 6-55 ENV_CORR-Nominal_TC2

The inputs to the system are varied in such a way that the aircraft are close to each other, in order to have correlation matrix elements different from zero.

6.7.3 Verification Plan for Operational Requirements

6.7.3.1 Code Generation and Equivalence Testing

Test Name:	**Code Generation and Equivalence Testing**
Test ID:	ENV_CORR-Operational_TC1
Related Requirements:	R-OPS-ENV_CORR: Operation Standard Requirements Matrix
Verification Data:	See section 6.8.3.1

The implemented simulation model running in normal mode and a S-function are excited with inputs signals.

Afterwards the outputs and the correlation matrices are checked for deviations. As long as the deviations stay below a certain threshold the test is passed.

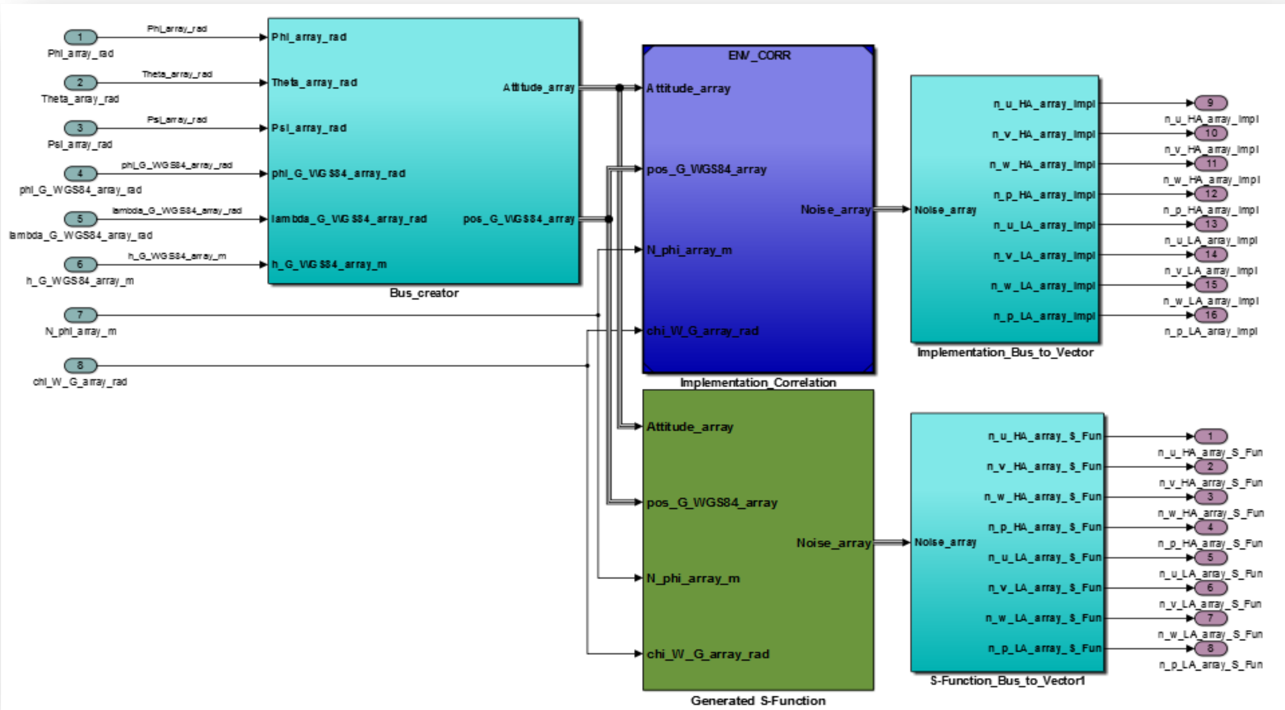


Figure 6-56 ENV_CORR-Operational_TC1

The inputs to the system are varied in such a way that the aircraft are close to each other, in order to have correlation matrix elements different from zero.

6.8 Verification Data

6.8.1 Verification of Implementation Requirements

6.8.1.1 Numeric Efficiency

Requirement Name **Numeric Efficiency**			Requirement ID R-NUM-ENV_CORR
Requirement is violated if the model contains one or more of the following items:			
Unused / Dead Code Branches Description	YES ☐	NO ☒	
Computational Redundancies Description	YES ☐	NO ☒	
Matrix Inversions Description	YES ☐	NO ☒	
Scalar Expansions of Vector/Matrix Math Description	YES ☐	NO ☒	
Circle Computations Description	YES ☐	NO ☒	
Inefficient Lookup Table Programming Description	YES ☐	NO ☒	
Algebraic Loops Description	YES ☐	NO ☒	
Numeric Efficiency met?	**YES ☒**	**NO ☐**	

Table 6-14 R-NUM-ENV_CORR

6.8.1.2 Implementation Compliance to FSD Style Guidelines

Requirement Name	Requirement ID	
Implementation Compliance to FSD Style Guidelines	R-SGC-ENV_CORR	
Requirement is violated if the model contains other blocks than the specified ones.		
Non-Specified Blocks within the Model?	YES ☐	NO ☒
Description		
Style Guide Compliance met?	**YES ☒**	**NO ☐**

Table 6-15 R-SGC-ENV_CORR

6.8.1.3 Implementation Standards Compliance

Requirement Name	Requirement ID	
Implementation Standards Compliance	R-ISC-ENV_CORR	
The coded algorithm must not contain any programming technique listed below unless detailed justification substantiates indispensability.		
*Discrete Switches**	YES ☒	NO ☐
Memory Blocks	YES ☐	NO ☒
Time Delays	YES ☐	NO ☒
Time Dependent / Non-Autonomous Elements	YES ☐	NO ☒
In-lined Integrations	YES ☐	NO ☒
Hysteresis and Quantized Elements	YES ☐	NO ☒
*Stochastic / Random Elements***	YES ☒	NO ☐
Normal atan Blocks	YES ☐	NO ☒
Operations with Sign Loss	YES ☐	NO ☒
Value Flipping and Range Limiting	YES ☐	NO ☒
Math Function out of Range	YES ☐	NO ☒
Division by Zero	YES ☐	NO ☒
Finite State Transition	YES ☐	NO ☒
Implementation Standards Compliance met?	**YES ☒**	**NO ☐**

Table 6-16 R-ISC-ENV_CORR

* The switches in the system do not affect the correct operation.

** Stochastic / Random Elements in the system do not affect the correct operation.

6.8.2 Verification of Functional Requirements

6.8.2.1 Results for Nominal Testing

Test Name:	**Correct Calculation of Correlated Signals**
Test ID:	ENV_CORR-Nominal_TC1
Verification Plan:	See section 6.7.2.1

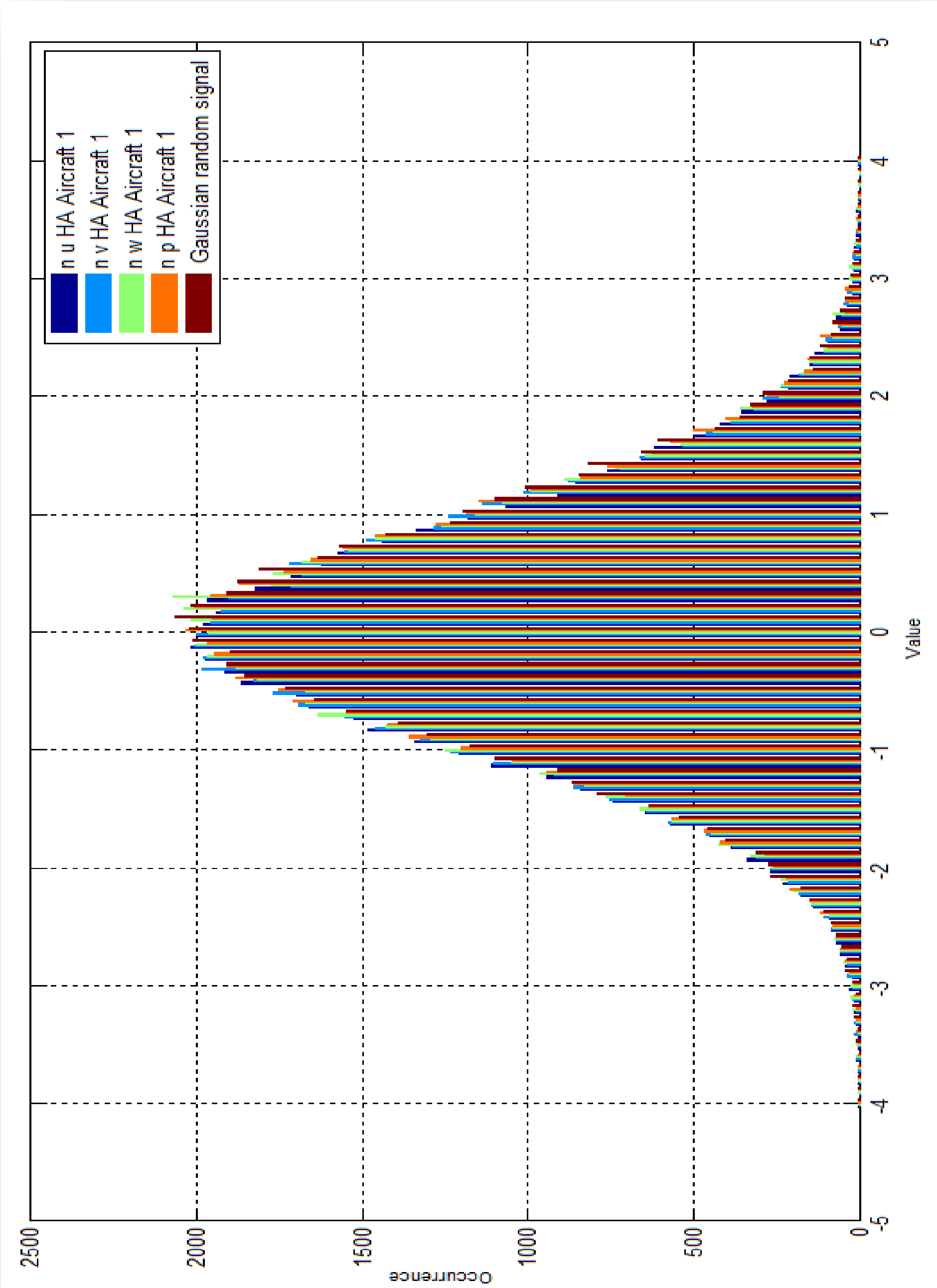


Figure 6-57 Noises Aircraft 1 $n^{HA}_{u_{array}}, n^{HA}_{v_{array}}, n^{HA}_{w_{array}}, n^{HA}_{p_{array}}$

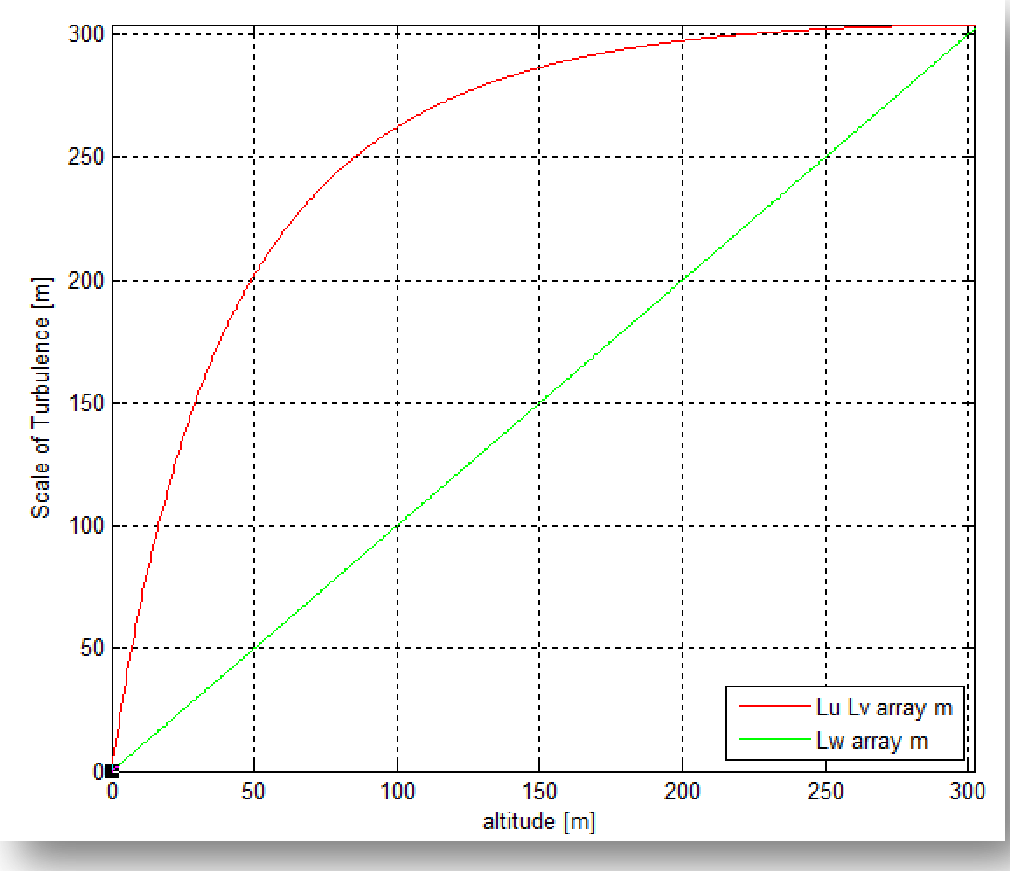


Figure 6-58 Scale of Turbulence $h < 1000\ ft$

Course of Noises and Scale of Turbulence realistic?	**YES ☒**	**NO ☐**

Test Name:	**Equivalence of Implementation and Real Correlation Matrices**
Test ID:	ENV_CORR-Nominal_TC1
Verification Plan:	See section 6.7.2.1
Number of Test Points:	5,000
Execution Time:	60.03s
Abs. Deviation Threshold:	5 %

All Deviations below Threshold	YES ☒	NO ☐
Absolute Deviations : comparison of the correlation matrix relative to the noises $n^{HA}_{u_{array}}$ calculated with the implementation and the real correlation matrix R^{HA}_{u}		
Average Absolute Deviation:	0.0076434	
Maximum Absolute Deviation:	0.039198	
Absolute Standard Deviation:	0.014082	

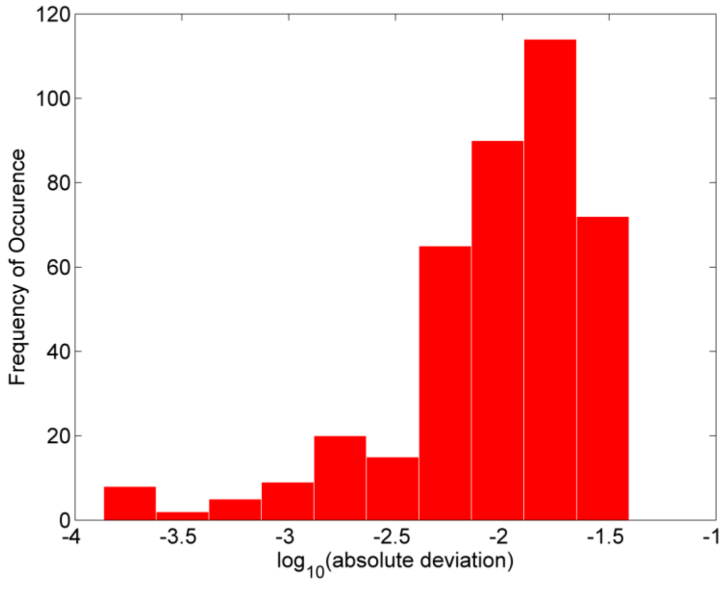


Figure 6-59 ENV_CORR-Nominal_TC1 - 1

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 6-17 ENV_CORR-Nominal_TC1 - 1

All Deviations below Threshold	YES ☒	NO ☐
Absolute Deviations : comparison of the correlation matrix relative to the noises $n_{v_{array}}^{HA}$ calculated with the implementation and the real correlation matrix R_v^{HA}		
Average Absolute Deviation:	0.0072482	
Maximum Absolute Deviation:	0.03823	
Absolute Standard Deviation:	0.014322	

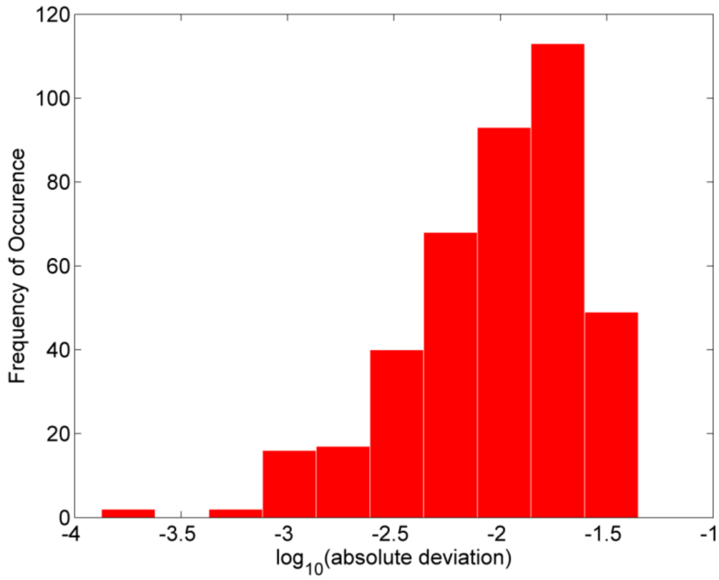


Figure 6-60 ENV_CORR-Nominal_TC1 - 2

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 6-18 ENV_CORR-Nominal_TC1 - 2

All Deviations below Threshold	YES ☒	NO ☐
Absolute Deviations : comparison of the correlation matrix relative to the noises $n_{w_{array}}^{HA}$ calculated with the implementation and the real correlation matrix R_w^{HA}		
Average Absolute Deviation: 0.0026489		
Maximum Absolute Deviation: 0.048145		
Absolute Standard Deviation: 0.015466		

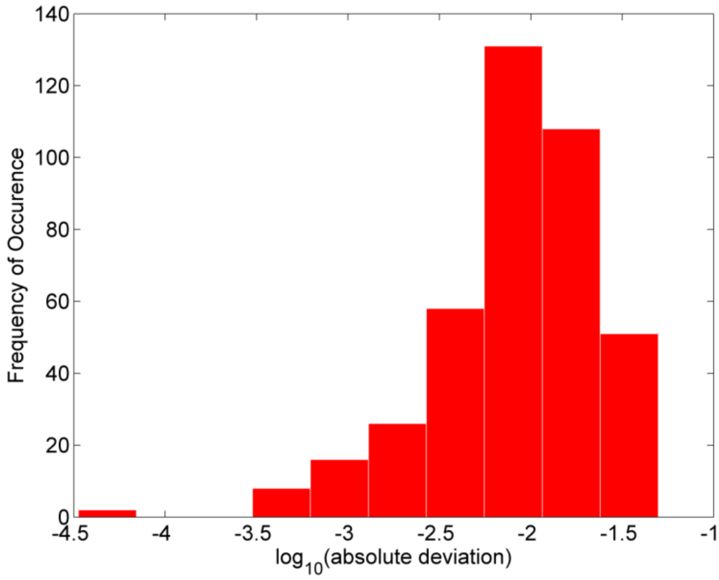


Figure 6-61 ENV_CORR-Nominal_TC1 - 3

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 6-19 ENV_CORR-Nominal_TC1 - 3

Equivalence of Implemented Model and Dissimilar Implementation?	**YES ☒**	**NO ☐**
Correct Nominal Behavior?	**YES ☒**	**NO ☐**

Test Name:	**Equivalence of Implementation and Dissimilar Implementation**
Test ID:	ENV_CORR-Nominal_TC2
Verification Plan:	See section 6.7.2.1
Number of Test Points:	5,000
Execution Time:	62.06
Abs. Deviation Threshold:	5%

All Deviations below Threshold YES ☒ NO ☐

Absolute Deviations : comparison of the correlation matrix relative to the noises $n_{u_{array}}^{HA}$ calculated with the implementation and the correlation matrix calculated with the dissimilar implementation

Average Absolute Deviation:	-0.0086726
Maximum Absolute Deviation:	0.033724
Absolute Standard Deviation:	0.015849

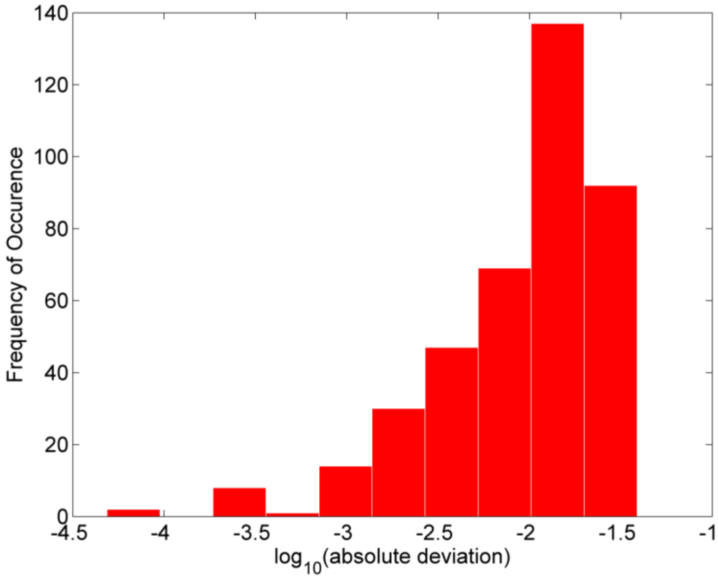


Figure 6-62 ENV_CORR-Nominal_TC2 - 1

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 6-20 ENV_CORR-Nominal_TC2 - 1

All Deviations below Threshold	YES ☒	NO ☐
Absolute Deviations : comparison of the correlation matrix relative to the noises $n_{v_{array}}^{HA}$ calculated with the implementation and the correlation matrix calculated with the dissimilar implementation		
Average Absolute Deviation:	-0.0075074	
Maximum Absolute Deviation:	0.040874	
Absolute Standard Deviation:	0.015464	

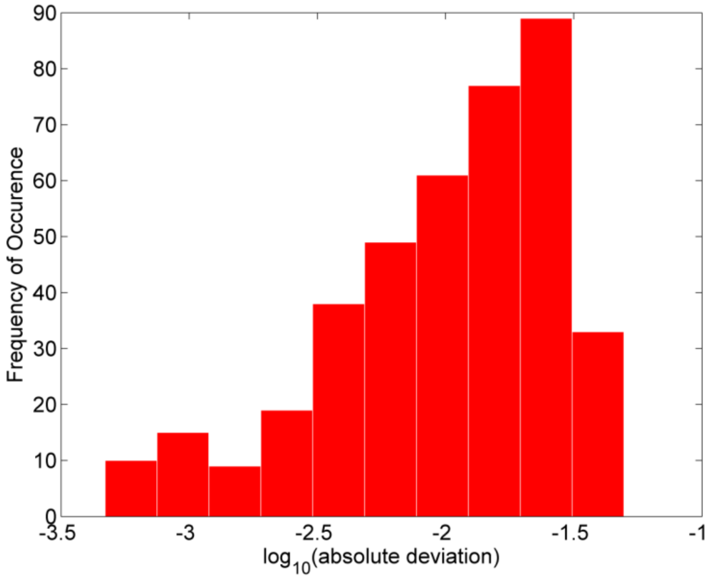


Figure 6-63 ENV_CORR-Nominal_TC2 - 2

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 6-21 ENV_CORR-Nominal_TC2 - 2

All Deviations below Threshold	YES ☒ NO ☐
Absolute Deviations : comparison of the correlation matrix relative to the noises $n_{w_{array}}^{HA}$ calculated with the implementation and the correlation matrix calculated with the dissimilar implementation	
Average Absolute Deviation:	0.0028555
Maximum Absolute Deviation:	0.040569
Absolute Standard Deviation:	0.016747

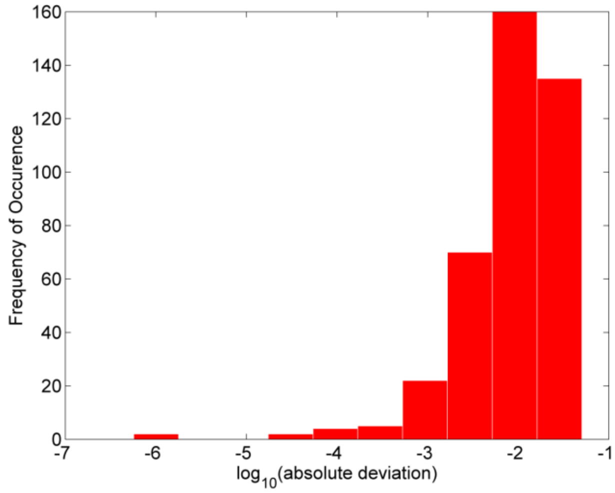


Figure 6-64 ENV_CORR-Nominal_TC2 - 3

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 6-22 ENV_CORR-Nominal_TC2 - 3

Equivalence of Implemented Model and Dissimilar Implementation?	**YES ☒**	**NO ☐**
Correct Nominal Behavior?	**YES ☒**	**NO ☐**

6.8.3 Verification of Operational Requirements

6.8.3.1 Results for Code Generation and Equivalence Testing

Test Name:	**Code Generation and Equivalence Testing**
Test ID:	ENV_CORR-Operational_TC1
Verification Plan:	See section 6.7.3.1
Number of Test Points:	5,000
Execution Time:	55.22s
Abs. Deviation Threshold:	1.00e-13

Equivalence of S-function and Simulation Model: correlation matrix relative to the noises $n_{u_{array}}^{HA}$		
All Deviations below Threshold	YES ☒	NO ☐
Absolute Deviations:		
Average Absolute Deviation:	-1.8476e-16	
Maximum Absolute Deviation:	1.8874e-15	
Absolute Standard Deviation:	6 .3669e-16	

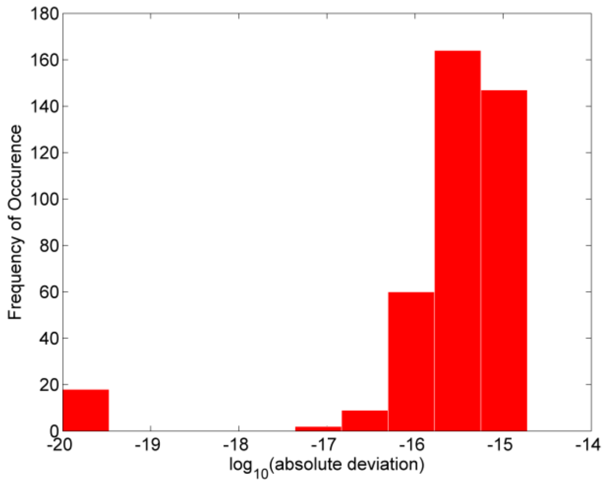


Figure 6-65 ENV_CORR-Operational_TC1 - 1

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 6-23 ENV_CORR-Operational_TC1 - 1

Equivalence of S-function and Simulation Model: **correlation matrix relative to the noises** $n_{v_{array}}^{HA}$		
All Deviations below Threshold	YES ☒	NO ☐
Absolute Deviations:		
Average Absolute Deviation:	4.4107e-17	
Maximum Absolute Deviation:	3.8858e-15	
Absolute Standard Deviation:	1.0543e-15	

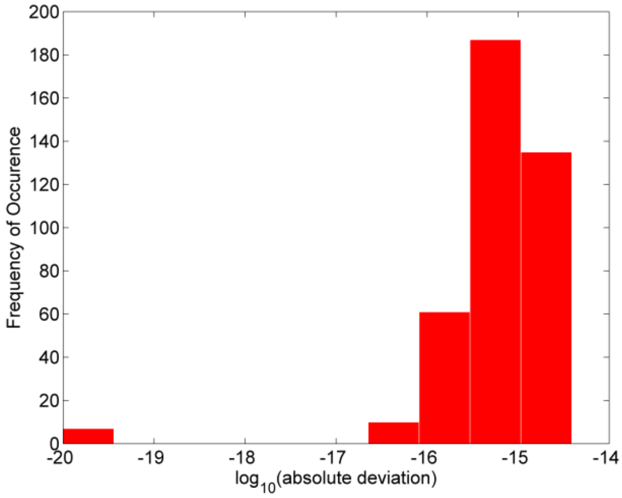


Figure 6-66 ENV_CORR-Operational_TC1 - 2

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 6-24 ENV_CORR-Operational_TC1 - 2

Equivalence of S-function and Simulation Model: **correlation matrix relative to the noises $n_{w_{array}}^{HA}$**		
All Deviations below Threshold	YES ☒	NO ☐
Absolute Deviations:		
Average Absolute Deviation:	-6.937e-17	
Maximum Absolute Deviation:	2.4425e-15	
Absolute Standard Deviation:	5.7207e-16	

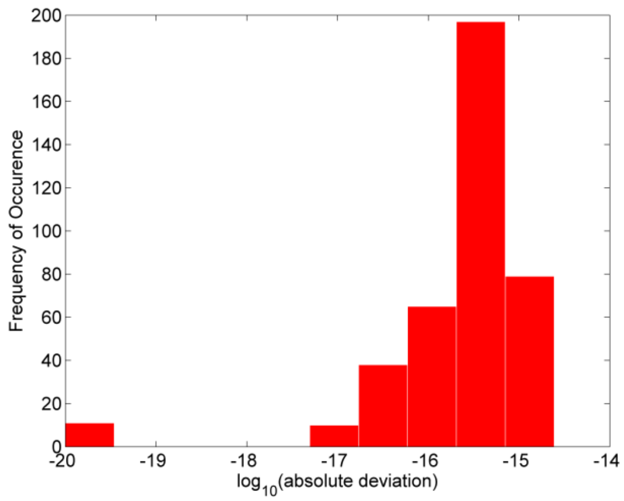


Figure 6-67 ENV_CORR-Operational_TC1 - 3

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 6-25 ENV_CORR-Operational_TC1 - 3

Equivalence of S-function and Simulation Model: **correlation matrix relative to the noises** n_{parray}^{HA}	
All Deviations below Threshold	YES ☒ NO ☐
Absolute Deviations:	
Average Absolute Deviation:	3.0835e-17
Maximum Absolute Deviation:	1.5543e-15
Absolute Standard Deviation:	5.5040e-16

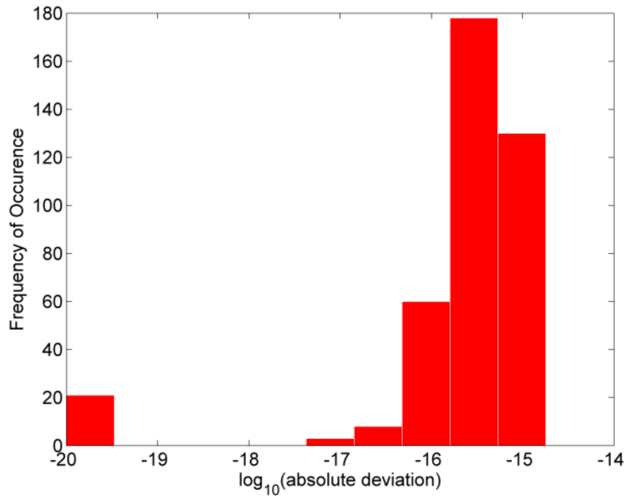


Figure 6-68 ENV_CORR-Operational_TC1 - 4

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 6-26 ENV_CORR-Operational_TC1 - 4

Equivalence of S-function and Simulation Model: **correlation matrix relative to the noises $n_{u_{array}}^{LA}$**		
All Deviations below Threshold	YES ☒	NO ☐
Absolute Deviations:		
Average Absolute Deviation:	-3.5062e-16	
Maximum Absolute Deviation:	1.5543e-15	
Absolute Standard Deviation:	6.7902e-16	

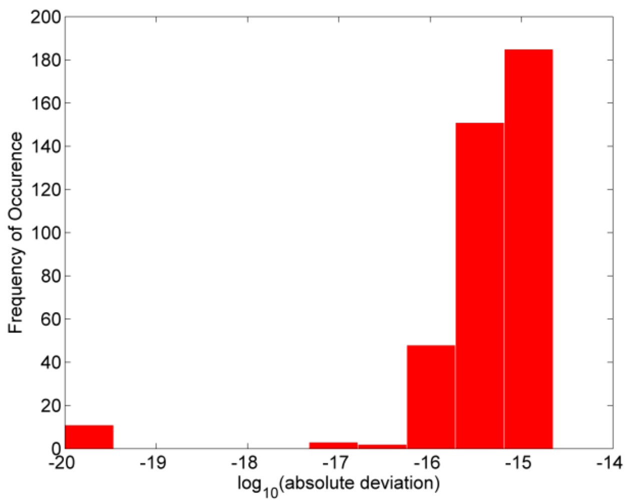


Figure 6-69 ENV_CORR-Operational_TC1 - 5

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 6-27 ENV_CORR-Operational_TC1 - 5

Equivalence of S-function and Simulation Model: **correlation matrix relative to the noises** $n_{v_{array}}^{LA}$		
All Deviations below Threshold	YES ☒	NO ☐
Absolute Deviations:		
Average Absolute Deviation:	5.8784e-17	
Maximum Absolute Deviation:	2.6645e-15	
Absolute Standard Deviation:	6.0747e-16	

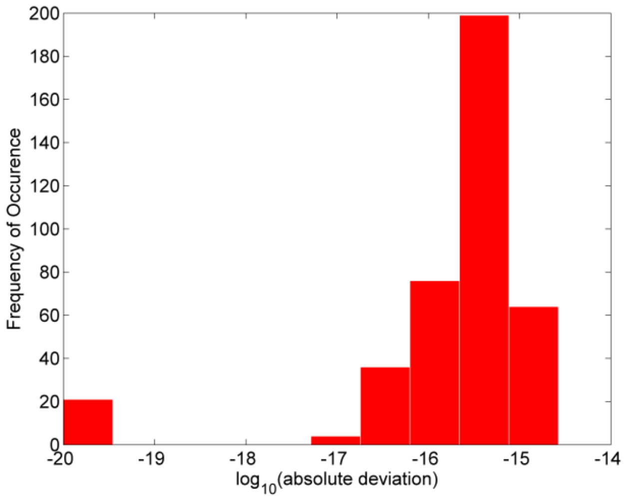


Figure 6-70 ENV_CORR-Operational_TC1 - 6

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 6-28 ENV_CORR-Operational_TC1 - 6

Equivalence of S-function and Simulation Model: **correlation matrix relative to the noises $n^{LA}_{w_{array}}$**		
All Deviations below Threshold	YES ☒	NO ☐
Absolute Deviations:		
Average Absolute Deviation:	3.0835e-17	
Maximum Absolute Deviation:	1.5543e-15	
Absolute Standard Deviation:	5.504e-16	

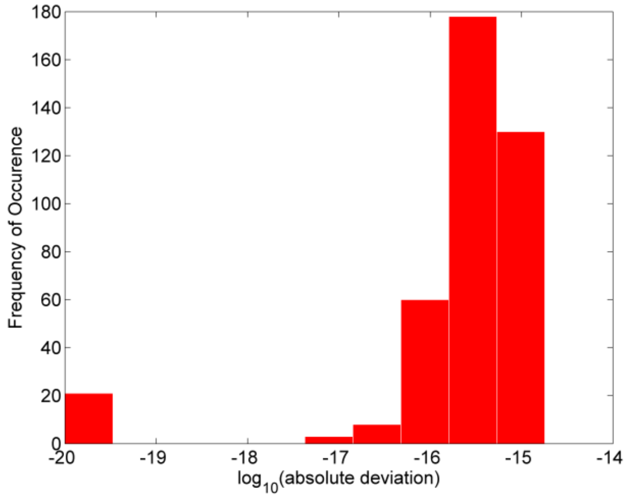


Figure 6-71 ENV_CORR-Operational_TC1 - 7

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 6-29 ENV_CORR-Operational_TC1 - 7

Equivalence of S-function and Simulation Model: **Equivalence of S-function and Simulation Model:** **correlation matrix relative to the noises** n_{parray}^{LA}		
All Deviations below Threshold		YES ☒ NO ☐
Absolute Deviations:		
Average Absolute Deviation:	3.0835e-17	
Maximum Absolute Deviation:	1.5543e-15	
Absolute Standard Deviation:	5.504e-16	

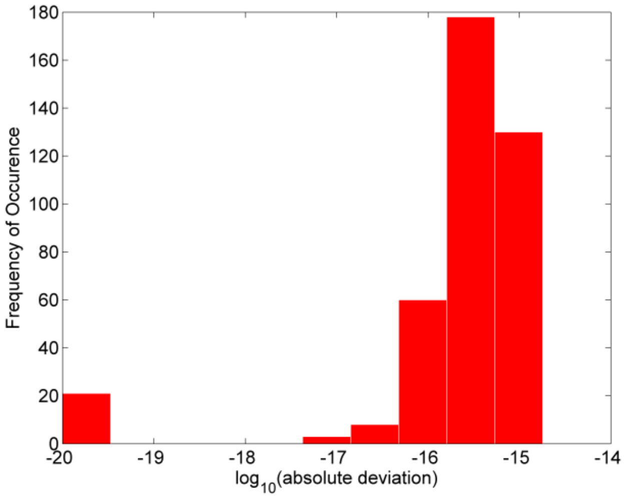


Figure 6-72 ENV_CORR-Operational_TC1 - 8

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 6-30 ENV_CORR-Operational_TC1 - 8

Equivalence of S-function and Simulation Model: **Scale of Turbulence Lu_Lv_array_m**	
All Deviations below Threshold	YES ☒ NO ☐
Absolute Deviations:	
Average Absolute Deviation:	0.0
Maximum Absolute Deviation:	0.0
Absolute Standard Deviation:	0.0

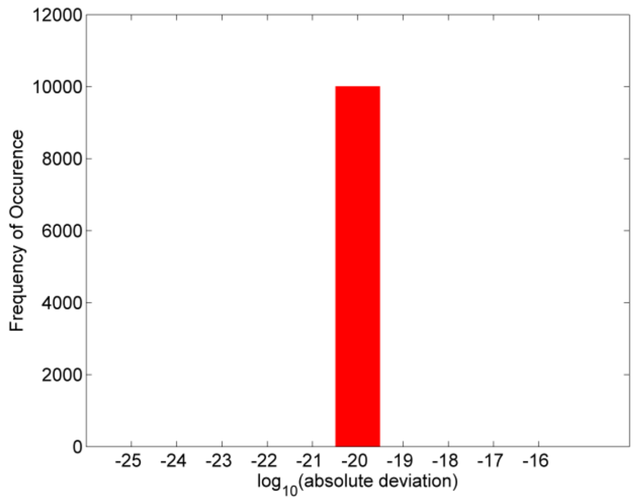


Figure 6-73 ENV_CORR-Operational_TC1 - 9

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 6-31 ENV_CORR-Operational_TC1 - 9

Equivalence of S-function and Simulation Model: **Scale of Turbulence Lw_array_m**		
All Deviations below Threshold	YES ☒	NO ☐
Absolute Deviations:		
Average Absolute Deviation:	0.0	
Maximum Absolute Deviation:	0.0	
Absolute Standard Deviation:	0.0	

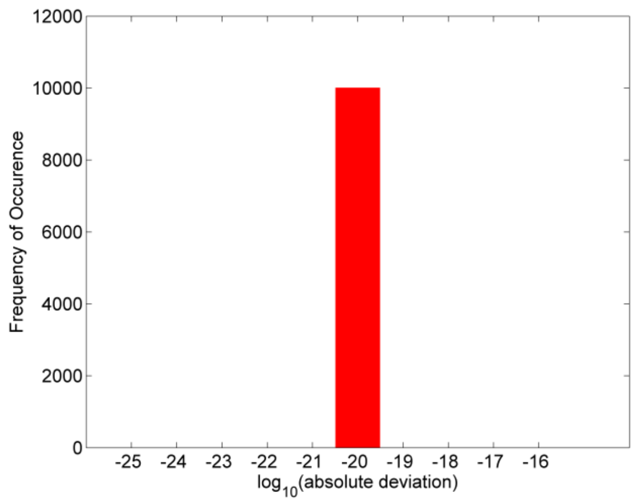


Figure 6-74 ENV_CORR-Operational_TC1 - 10

Description of deviation exceeding the pre-specified threshold and assessment of possible causes

Table 6-32 ENV_CORR-Operational_TC1 - 10

Compilation of S-function successful?	YES ☒	NO ☐
Compilation of standalone executable successful?	YES ☒	NO ☐
Description of Compilation Errors		
Warnings during Compilation?	YES ☐	NO ☒
Description of Compilation Warnings		

Run-Time Errors or Warnings?	YES ☐	NO ☒
Description of Error / Warning Messages		
Code Generation successful and Coded Version equivalent to Simulation Model?	**YES ☒**	**NO ☐**

7 Final Assembly of *Atmosphere Model*

7.1 Description of the Functional and Operational Intent

The *Atmosphere Model* is intended to compute the main proprieties of the Standard Atmosphere (Temperature (T), Pressure (P), Density (ρ), Speed of Sound (a), Dynamic Viscosity (μ) and Kinematic Viscosity (ν)), the wind velocity $\left(\vec{\boldsymbol{v}}_W^G\right)_O^E$, the wind velocity $\left(\vec{v}_{W_{turb}}^G\right)_B^E$ and the wind angular rate $\left(\vec{\omega}_{W_{turb}}^{EW}\right)_B$ in case of turbulence and the wind gust velocity $\left(\vec{v}_{W_{gust}}^G\right)_B^E$.

7.2 Requirements

7.2.1 Functional Requirements

Requirement Name	Requirement ID
Computation of Temperature	R-FUN-ATM_ISA_01
Derived from Purpose of the system.	
Requirement Definition The Temperature of the atmosphere in the troposphere and lower stratosphere, referred to geopotential altitudes, shall be computed.	

Table 7-1 R-FUN-ATM_ISA_01

Requirement Name	Requirement ID
Computation of Pressure	R-FUN-ATM_ISA_02
Derived from Purpose of the system.	
Requirement Definition The Pressure of the atmosphere shall be computed in the troposphere and lower stratosphere, referred to geopotential altitudes.	

Table 7-2 R-FUN-ATM_ISA_02

Requirement Name **Computation of Density**	Requirement ID R-FUN-ATM_ISA_03
Derived from Purpose of the system.	
Requirement Definition The system shall compute the Density of the atmosphere in the troposphere and lower stratosphere, referred to geopotential altitudes.	

Table 7-3 R-FUN-ATM_ISA_03

Requirement Name **Computation of Speed of Sound**	Requirement ID R-FUN-ATM_ISA_04
Derived from Purpose of the system.	
Requirement Definition The Speed of Sound of the atmosphere in the troposphere and lower stratosphere, referred to geopotential altitudes, shall be computed.	

Table 7-4 R-FUN-ATM_ISA_04

Requirement Name **Computation of Dynamic Viscosity**	Requirement ID R-FUN-ATM_ISA_05
Derived from Purpose of the system.	
Requirement Definition The Dynamic Viscosity shall be computed in the troposphere and lower stratosphere, referred to geopotential altitudes.	

Table 7-5 R-FUN-ATM_ISA_05

Requirement Name **Computation of Kinematic Viscosity**	Requirement ID R-FUN-ATM_ISA_06
Derived from Purpose of the system.	
Requirement Definition The system shall compute the Kinematic Viscosity in the troposphere and lower stratosphere, referred to geopotential altitudes.	

Table 7-6 R-FUN-ATM_ISA_06

Requirement Name	Requirement ID
Computation the Wind Velocity $\left(\vec{v}_W^G\right)_O^E$	R-FUN-ATM_WIND_07
Derived from Purpose of the system.	
Requirement Definition The Wind Velocity of the center of gravity *G* defined with respect to the Earth Centered Fixed (E) frame, components written in O frame, shall be computed.	

Table 7-7 R-FUN-ATM_WIND_07

Requirement Name	Requirement ID
Computation of Air Velocity	R-FUN-ATM_TURB_08
Derived from Purpose of the system.	
Requirement Definition The system shall compute the Wind Velocity of the center of gravity *G* defined with respect to the Earth Centered Fixed (E) frame, components written in Body frame (B) .	

Table 7-8 R-FUN-ATM_TURB_08

Requirement Name	Requirement ID
Computation of Angular Velocity of the Air	R-FUN-ATM_TURB_09
Derived from Purpose of the system.	
Requirement Definition The system shall compute the Wind angular rate of the Wind frame (W) relative to the Earth Centered Fixed (E) frame, components written in Body frame (B) .	

Table 7-9 R-FUN-ATM_TURB_09

Requirement Name	Requirement ID
Computation of Wind Gust Velocity	R-FUN-ATM_GUST_10
Derived from Purpose of the system.	
Requirement Definition The Wind Gust Velocity shall be computed.	

Table 7-10 R-FUN-ATM_GUST_10

7.2.2 Operational Requirements

Requirement Name	Requirement ID
Incorporation into Flight Simulator Simulation Model	R-OPS-ATM
Derived from Usage intents	
Requirement Definition The model shall be incorporated into the simulation model of an (atmospheric) flight simulation device. Therefore it is necessary that all components support code generation.	

Table 7-11 R-OPS-ATM

7.2.3 Implementation Requirements

Requirement Name	Requirement ID
Numeric Efficiency	R-NUM-ATM
Derived from Global Implementation Guidelines	
Requirement Definition The coded algorithm must not contain any of the numerical inefficient programming techniques listed below unless detailed justification substantiates indispensability.	
Programming Techniques to be Avoided:	
Unused / Dead Code Branches	
Computational Redundancies	
Matrix Inversions	
Scalar Expansion of Vector / Matrix Math	
Circle Computations	
Inefficient Lookup Table Programming	
Algebraic Loops	

Table 7-12 R-NUM-ATM

Requirement Name	Requirement ID
Input / Output Interface Compliance to Parent System and Child Systems	R-IOC-ATM
Derived from	
Global Implementation Guidelines	
Requirement Definition	
Input and Output interface must comply with parent system.	
Compliance required to:	
Global bus object definitions	
I/O signal name matching to parent system	
I/O signal unit matching to parent system	
I/O signal data type matching to parent system	
I/O signal data range compatibility matching to parent system	

Table 7-13 R-IOC-ATM

Requirement Name	Requirement ID
Implementation Compliance to FSD Style Guidelines	R-SGC-ATM
Derived from	
Global Implementation Guidelines	
Requirement Definition	
Only a subset of SIMULINK blocks is allowed to be implemented.	
Allowed Libraries and Toolboxes:	
FSD Compliant Base	
Use of other Libraries and Toolboxes is Forbidden!	

Table 7-14 R-SGC-ATM

Requirement Name **Implementation Standards Compliance**	Requirement ID R-ISC-ATM
Derived from Global Implementation Guidelines	
Requirement Definition The coded algorithm must not contain any programming technique listed below unless detailed justification substantiates indispensability.	
Forbidden Programming Techniques:	
Discrete Switches*	
Memory Blocks	
Time Delays	
Time Dependent / Non-Autonomous Elements	
In-lined Integrations	
Hysteresis and Quantized Elements	
Stochastic / Random Elements	
Normal atan Blocks	
Operations with Sign Loss	
Value Flipping and Range Limiting	
Math Function out of Range	
Division by Zero	
Finite State Transition	

Table 7-15 R-ISC-ATM

* The switches in the system do not affect the correct operation

7.3 Usage Analysis

Domain of Use:	Atmospheric Flight

Description of Domain

The implemented model is used for atmospheric flight, especially in the troposphere and lower stratosphere. The implemented physical relations are valid as long as the assumptions made in the WGS84 model are valid.

Definition of Domain by Inputs

Name	Unit	Range	
		Minimum	**Maximum**
delta_T_K	K	-100	100
delta_P_Pa	Pa	-5000	5000
H_Vector_Shear_m	m	0	20000
Dchi_Vector_Shear_rad	rad	-pi	+pi
H_Wind_Shear_m	m	0	20000
DV_Wind_Shear_mDs	mDs	0	Inf
h_array_WGS84_m	m	-500	20000
vel_W_array_E_W_mDs	mDs	0	Inf
chi_W_array_rad	rad	-pi	+pi
h_GND_m	m	0	20000
Psi_rad	rad	-pi	pi
Theta_rad	rad	-pi/2	pi/2
Phi_rad	rad	-pi	pi
phi_G_WGS84_rad	rad	-pi/2	pi/2
lambda_G_WGS84_rad	rad	-pi	pi
h_G_WGS84_m	m	-500	2e4
Probability_of_Exceedance	-	1	7
V_TAS_mDs	mDs	-Inf	Inf
b_m	m	0	Inf
n_u_HA	-	-Inf	Inf
n_v_HA	-	-Inf	Inf
n_w_HA	-	-Inf	Inf
n_p_HA	-	-Inf	Inf
n_u_LA	-	-Inf	Inf
n_v_LA	-	-Inf	Inf
n_w_LA	-	-Inf	Inf

	n_p_LA	-	-Inf	Inf
	Airspeed_mDs	m/s	-Inf	Inf
	Gust_Ampitude_u_mDs	m	-Inf	Inf
	Gust_Ampitude_v_mDs	m	-Inf	Inf
	Gust_Ampitude_w_mDs	m	-Inf	Inf
	Gust_Length_x_m	m	0	Inf
	Gust_Length_y_m	m	0	Inf
	Gust_Length_z_m	m	0	Inf
	flg_start_gust_x	-	0	1
	flg_start_gust_y	-	0	1
	flg_start_gust_z	-	0	1

Finite state transitions in considered domain?	YES ☐	NO ☒
Description of finite state transitions		
Known algorithm validity bounds considered domain?	YES ☐	NO ☒
Description of known validity bounds		
Known problems / exceptions in considered domain?	YES ☐	NO ☒
Description of known problems / exceptions		
Non-algebraic I/O-relationships?	YES ☐	NO ☒
Description of non-algebraic I/O relationships		
Intentional non-deterministic I/O-behavior?	YES ☐	NO ☒
Description of intentional non-deterministic I/O-behavior		

Table 7-16 Usage Analysis

7.4 Architecture Specification

7.4.1 Parent / Child Systems

7.4.1.1 Parent System

The system is a top level system.

7.4.1.2 Child Systems

This system contains child systems.

7.4.2 Signal Definitions

7.4.2.1 Inputs

Inputs							
Symbol	**Name**	**Size**	**Components**	**Data Type**	**Min**	**Max**	**Description**
ΔT	delta_T_K	[1 1]	-	double	-100	100	Temperature change at MSL to take into account climatic conditions
ΔP	dellta_P_Pa	[1 1]	-	double	-5000	5000	Pressure change at MSL to take into account climatic conditions
$DV_{WindShear}$	DV_Wind_Shear_mDs	[1 1]	-	double	0	Inf	External input for Wind Intensity: velocity variation
$H_{WindShear}$	H_Wind_Shear_m	[1 1]	-	double	0	2e4	External input for Wind Intensity: height variation
$D\chi_{VectorShear}$	Dchi_Vector_Shear_rad	[1 1]	-	double	-pi	+pi	External input for Initial Wind Orientation: direction variations
$H_{VectorShear}$	H_Vector_Shear_m	[1 1]	-	double	0	2e4	External input for Initial Wind Orientation: height variations
h^{array}	h_array_WGS84_m	[5 1]	-	double	-500	2e4	External input for interpolation: heights
$\left(v_W^{array}\right)_W^E$	vel_W_array_E_W_mDs	[5 1]	-	double	0	Inf	External input for interpolation: velocities
χ_W^{array}	chi_W_array_rad	[5 1]	-	double	-pi	-pi	External input for interpolation: Wind Direction referred to *O* frame (North)
h_{GND}	h_GND_m	[1 1]	-	double	0	2e4	Height of the aircraft's center of gravity above the ground.
b	B_m	[1 1]	-	double	0	Inf	Wingspan

Table 7-17 Inputs

Inputs							
Symbol	**Name**	**Size**	**Components**	**Data Type**	**Min**	**Max**	**Description**
Ψ	Psi_rad	[1 1]	-	double	-pi	pi	Yaw angle of the aircraft
Θ	Theta_rad	[1 1]	-	double	-pi/2	pi/2	Pitch angle of the aircraft
Φ	Phi_rad	[1 1]	-	double	-pi	pi	Roll angle of the aircraft
φ^G	phi_G_WGS84_rad	[1 1]	-	double	-pi/2	pi/2	Geodetic Latitude of the aircraft's center of gravity.
λ^G	lambda_G_WGS84_rad	[1 1]	-	double	-pi	pi	Geodetic Longitude of the aircraft's center of gravity.
h^G	h_G_WGS84_m	[1 1]	-	double	-500	20000	Height of the aircraft's center of gravity above the WGS84 reference ellipsoid.
$Prob_{Exc}$	Probability_Exceedence	[1 1]	-	double	1	7	Probability of the turbulence intensity being exceeded.
V_{TAS}	V_TAS_mDs	[1 1]	-	double	-pi/2	pi/2	True Air Speed perceived by the aircraft.
n_u^{HA}	n_u_HA	[1 1]	-	double	-Inf	Inf	Random signal required to calculate u_{turb} in the model of high altitude.
n_v^{HA}	n_v_HA	[1 1]	-	double	-Inf	Inf	Random signal required to calculate v_{turb} in the model of high altitude.
n_w^{HA}	n_w_HA	[1 1]	-	double	-Inf	Inft	Random signal required to calculate w_{turb} in the model of high altitude.

Table 7-17 Inputs Part II

Inputs							
Symbol	**Name**	**Size**	**Components**	**Data Type**	**Min**	**Max**	**Description**
n_p^{HA}	n_p_HA	[1 1]	-	double	-Inf	Inf	Random signal required to calculate p_{turb} in the model of high altitude.
n_u^{LA}	n_u_LA	[1 1]	-	double	-Inf	Inf	Random signal required to calculate u_{turb} in the model of low altitude.
n_v^{LA}	n_v_LA	[1 1]	-	double	-Inf	Inf	Random signal required to calculate v_{turb} in the model of low altitude.
n_w^{LA}	n_w_LA	[1 1]	-	double	-Inf	Inf	Random signal required to calculate w_{turb} in the model of low altitude.
n_p^{LA}	n_p_LA	[1 1]	-	double	-Inf	Inf	Random signal required to calculate p_{turb} in the model of low altitude.
V_{air}	Airspeed_mDs	[1 1]	-	double	-Inf	Inf	Velocity of the Air.
v_x	Gust_Amplitude_u_mDs	[1 1]	-	double	-Inf	Inf	Gust Amplitude in direction of x
v_y	Gust_Amplitude_v_mDs	[1 1]	-	double	-Inf	Inf	Gust Amplitude in direction of y
v_z	Gust_Amplitude_w_mDs	[1 1]	-	double	-Inf	Inf	Gust Amplitude in direction of z
d_x	Gust_Length_x_m	[1 1]	-	double	0	Inf	Gust Length in direction of x

Table 7-17 Inputs Part III

Inputs							
Symbol	**Name**	**Size**	**Components**	**Data Type**	**Min**	**Max**	**Description**
d_y	Gust_Length_y_m	[1 1]	-	double	0	Inf	Gust Length in direction of y
d_z	Gust_Length_z_m	[1 1]	-	double	0	Inf	Gust Length in direction of z
flg_x	flg_start_gust_x	[1 1]	-	double	0	1	Flag to start the gust component x
flg_y	flg_start_gust_y	[1 1]	-	double	0	1	Flag to start the gust component y
flg_z	flg_start_gust_z	[1 1]	-	double	0	1	Flag to start the gust component z

Table 7-17 Inputs Part IV

7.4.2.2 Outputs

Outputs							
Symbol	**Name**	**Size**	**Components**	**Data Type**	**Min**	**Max**	**Description**
T	T_K	[1 1]	-	double	0	-	Atmospheric temperature relative to the geopotential altitude.
P	P_Pa	[1 1]	-	double	0	-	Atmospheric pressure relative to the geopotential altitude.
ρ	rho_kgDm3	[1 1]	-	double	0	-	Atmospheric density relative to the geopotential altitude.
a	a_mDs	[1 1]	-	double	0	-	Atmospheric speed of sound relative to the geopotential altitude.
μ	mu_sPa	[1 1]	-	double	-	-	Atmospheric dynamic viscosity relative to the geopotential altitude.
ν	nu_m3sPaDkg	[1 1]	-	double	-	-	Atmospheric kinematic viscosity relative to the geopotential altitude.
$(\vec{v}_W^G)_O^E$	vel_W_G_E_O_mDs	[3 1]	u_W_G_E_O_mDs v_W_G_E_O_mDs 0	double	-Inf -Inf 0	Inf Inf 0	Wind velocity of the center of gravity G defined with respect to the Earth Centered Fixed (E) frame, components written in O frame

Table 7-18 Outputs

Outputs							
Symbol	**Name**	**Size**	**Components**	**Data Type**	**Min**	**Max**	**Description**
$Error_{Negative_{DV}}$	ERROR_Negative_DV_Wind_Shear_mDs	[1 1]	-	double	0	1	Error signal: $DV_{WindShear}$ negative
$Error_{Wrong_{vel_{20ft}}}$	ERROR_Wrong_vel_20ft_mDs	[1 1]	-	double	0	1	Error signal: wrong value of $\left(v_W^{20ft}\right)_W^E$
$\left(\vec{v}_{W_{turb}}^G\right)_B^E$	Vel_W_turb_G_E_B_mDs	[3 1]	u_W_turb_G_E_B_mDs v_W_turb_G_E_B_mDs u_W_gust_G_E_B_mDs	double	-Inf -Inf -Inf	Inf Inf Inf	Wind Velocity of the center of gravity *G* defined with respect to the Earth Centered Fixed (E) frame, components written in Body frame (B) .
$\left(\vec{\omega}_{W_{turb}}^{EW}\right)_B$	rot_W_turb_EW_B_radDs	[3 1]	v_W_gust_G_E_B_mDs w_W_gust_G_E_B_mDs u_W_turb_G_E_B_mDs	double	-Inf -Inf -Inf	Inf Inf Inf	Wind angular rate of the Wind frame (W) relative to the Earth Centered Fixed (E) frame, components written in Body frame (B) .
$\left(\vec{v}_{W_{gust}}^G\right)_B^E$	Vel_W_gust_G_E_B_mDs	[3 1]	u_W_gust_G_E_B_mDs v_W_gust_G_E_B_mDs w_W_gust_G_E_B_mDs	double	-Inf -Inf -Inf	Inf Inf Inf	Wind Velocity of the Center of Gravity *G* defined with respect to the Earth Centered Fixed Frame *(E)*, components written in Body Frame *(B)*.

Table 7-18 Outputs Part II

7.4.3 Bus Structure

Inputs

L	Bus Name	Elements	Element Types
0	*External_Inputs_ISA_Bus*	delta_T_K delta_P_Pa	double double
0	*pos_G_WGS84_Bus*	phi_G_WGS84_rad lambda_G_WGS84_rad h_G_WGS84_m	double double double
0	*att_euler_Bus*	Psi_rad Theta_rad Phi_m	double double double
0	*Noise_Bus*	Noise_HA_Bus Noise_LA_Bus	Noise_HA_Bus Noise_LA_Bus
1	*Noise_HA_Bus*	n_u_HA n_v_HA n_w_HA n_p_HA	double double double double
1	*Noise_LA_Bus*	n_u_LA n_v_LA n_w_LA n_p_LA	double double double double
0	*External_Inputs_Wind_Shear_Bus*	DV_Wind_Shear_mDs H_Wind_Shear_m	double double
0	*External_Inputs_Vector_Shear_Bus*	Dchi_Vector_Shear_rad H_Vector_Shear_m	double double

Table 7-19 Inputs Bus Structure

0	*Wind_External_Inputs_Bus*	h_array_WGS84_m vel_W_array_E_W_mDs chi_W_array_rad	double double double
0	*Gust_Amplitude_Bus*	Gust_Amplitude_u_mDs Gust_Amplitude_v_mDs Gust_Amplitude_w_mDs	double double double
0	*Gust_Length_Bus*	Gust_Length_x_m Gust_Length_y_m Gust_Length_z_m	double double double
0	*flg_start_gust_Bus*	flg_start_gust_x flg_start_gust_y flg_start_gust_z	double double double

Table 7-19 Inputs Bus Structure Part II

<u>Outputs</u>

L	Bus Name	Elements	Element Types
0	*ISA_Variables_H_G_Bus*	T_K P_Pa rho_kgDm3 a_mDs mu_sPa nu_ m3sPaDkg	double double double double double double
0	*vel_W_G_E_O_Bus*	u_W_G_E_O_mDs v_W_G_E_O_mDs w_W_G_E_O_mDs	double double double
0	*Vel_W_turb_G_E_B_Bus*	u_W_turb_G_E_B_mDs v_W_turb_G_E_B_mDs w_W_turb_G_E_B_mDs	double double double
0	*rot_W_turb_EW_B_Bus*	p_W_turb_EW_B_radDs q_W_turb_EW_B_radDs r_W_turb_EW_B_radDs	double double double
0	*Vel_W_gust_G_E_B_Bus*	u_W_gust_G_E_B_mDs v_W_gust_G_E_B_mDs w_W_gust_G_E_B_mDs	double double double

Table 7-20 Outputs Bus Structure

8 Conclusions

The Atmosphere Model and the Correlation Model implemented in MATLAB® and Simulink® have been developed.

The models already available in Simulink® have represented the starting point for the modeling and implementation of the subsystems of the Atmosphere Model. Due to the exclusive use of Libraries and Toolboxes property of the FSD Institute and especially because of the requirements imposed by the client, the models developed have many differences compared to such models.

The most innovative part of the whole work involved the construction of the Correlation Model: a long time has been spent on the study of the correlation of signals and, above all, on the study of the correlation between the air velocities perceived in different points of the space. Once it has been established such correlation, the greater difficulty was represented by the determination of a general procedure that allows to calculate random signals correlated in a known manner in order to provide the input signals to the Turbulence Model.

These difficulties have depended primarily by the fact that it has not been found a reference that would provide clearly that procedure; for this reason, the development of the Correlation Model has been very challenging and hardworking.

The entire work has therefore proved to be very interesting even if the study of the mathematical models used turned out to be quite complex.

Finally, after completing the implementation, all systems made have been checked carefully and compared, where possible, with existing models or with different computational procedures; the results obtained were satisfactory.

For each subsystem and for the assembled system the reports have been made.

9 References

[1] U.S: Government Printing Office, " U.S. Standard Atmosphere 1976", Washington D.C., 1976

[2] International Organization for Standardization, *Standard Atmosphere*, ISO 2533:1975, 1975

[3] U.S. Military Specification *MIL-F-8785C*, *5 November 1980*

[4] David T. Sawdy, "On the two-dimensional atmospheric turbulence response of an airplane", University of Wichita, 1960.

[5] T. Von Karman and L. Howarth, "On the Statistical Theory of Isotropic Turbulence", Turbulence - Classic Paper on Statistical Theory, S.K.Friedlander and Leonard Topper, eds. Interscience Publications, New York,1961

[6] www.mathworks.com

[7] Julian Praceus, "Modellierung und Implementierung der Dynamischen Atmosphäre für Ein High-Fidelity Flugsimulationsmodell", TUM, München, 2010

[8] Prof. Dr.-Ing. Florian Holzapfel, "Lecture Flight System Dynamics II", TUM, München

[9] "FSD Naming Convention", FSD TUM, München

[10] Giovanni Mengali, "Elementi di Dinamica del Volo con Matlab", Edizioni ETS, Pisa, 2003

[11] P.A. van Gastel and W.H.J.J. van Staveren, "Aircraft Responses to Atmospheric Turbulence. A Comparative Study", TU Delft, 1994

10 Appendix A : Integration of the Correlation Model

10.1 Integration with other systems: verifications

The following is the verification of correct operation of the functions used to integrate the *Correlation Model* with other systems.

The verification was carried out by repeating the data of only three aircraft and sending the results to Matlab Workspace to make a comparison.

As a test, were selected only three signals.

The test shows that the system operates correctly.

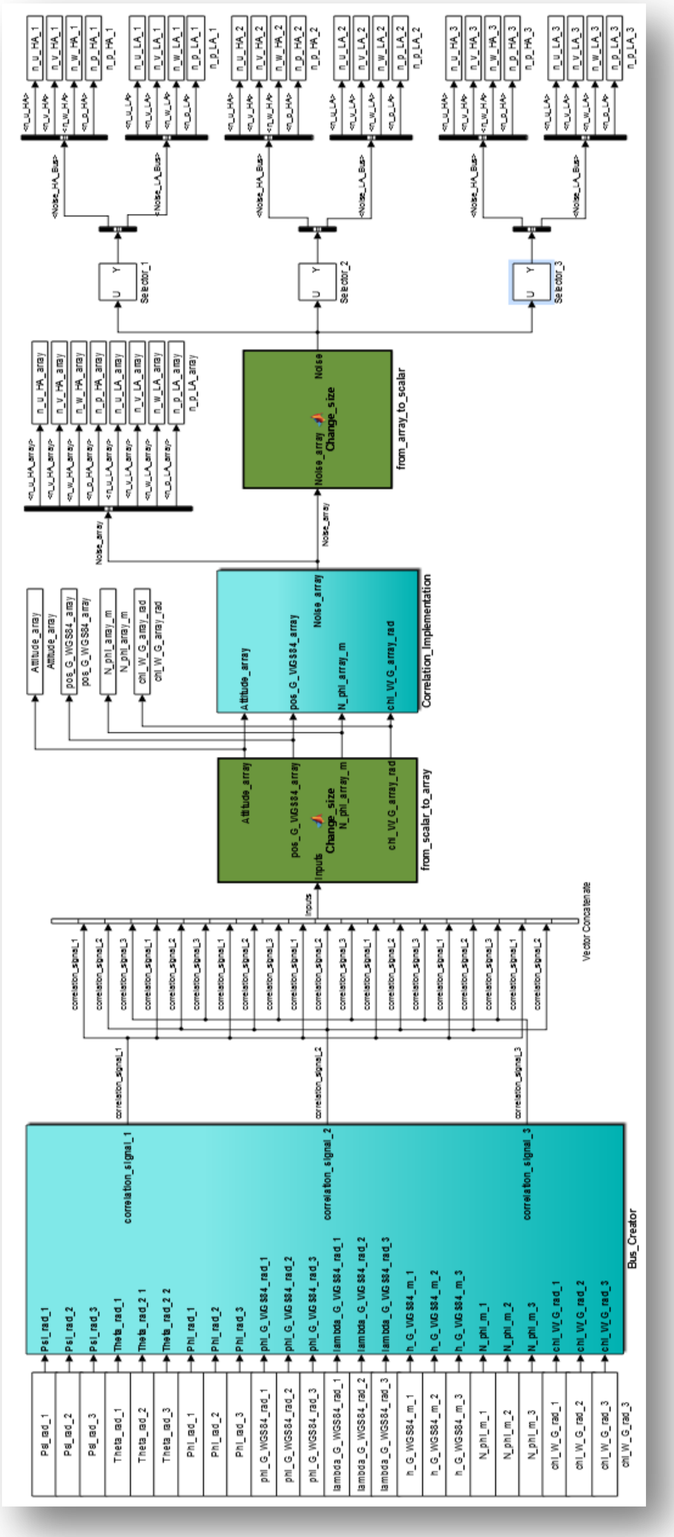


Figure 10-1 Appendix A:Test

11 Appendix B: Nomenclature and Reference Frames

The following information are taken from references [8] and [9] and are useful to clarify the nomenclature and the reference systems used in the previous chapters.

11.1 Nomenclature and Designation Principles

Below only the information of interest for the understanding of the nomenclature used in the previous chapters are listed:

- Frames (for more information see section 11.2)

Name	Symbol
- ECI Frame	*I*
- ECEF Frame	*E*
- WGS 84 Coordinates	WGS84
- NED Frame	*O*
- Body Frame	*B*

- Points

Name	Symbol
- Center of gravity	G
- Center of the Earth	O

- Signal Reference

Name	Symbol
- Kinematic	K
- Aerodynamic	A
- Wind	W

❖ Components

Name	Symbol
- Positions	x, y, z
- Velocities	u, v, w
- Angular Rates	p, q, r

11.1.1 Nomenclature and Notation

For each quantity, the point and the frame against which it was calculated and the frame with respect to which it is expressed, is indicated.

❖ Position $\vec{\boldsymbol{r}}$

Position of point G relative to the center of the Earth O:

Nomenclature	$(\vec{r}^G)_B$	$\begin{bmatrix} x^G \\ y^G \\ z^G \end{bmatrix}_B$
Notation	pos_G_B_m	x_G_B_m y_G_B_m z_G_B_m

B-Frame is the notation frame

Position are always specified in [m].

❖ Velocities $\vec{\boldsymbol{v}}$

Velocity (Wind Velocity) of point G with respect to the Earth Center O:

Nomenclature	$(\vec{v}_W^G)_B^E$	$\begin{bmatrix} u_W^G \\ v_W^G \\ w_W^G \end{bmatrix}_B^E$
Notation	vel_W_G_E_B_mDs	u_W_G_E_B_mDs v_W_G_E_B_mDs w_W_G_E_B_mDs

Velocity is defined with respect to E-Frame

B-Frame is the notation frame

❖ Angular Rates $\vec{\omega}$

Angular Rates (Wind Angular Rates):

Nomenclature	$(\vec{\omega}_W^{OB})_B$	$\begin{bmatrix} p_W \\ q_W \\ r_W \end{bmatrix}_B = \begin{bmatrix} \omega_{W,x}^{OB} \\ \omega_{W,y}^{OB} \\ \omega_{W,z}^{OB} \end{bmatrix}_B$
Notation	rot_W_OB_B_radDs	p_W_OB_B_radDs q_W_OB_B_radDs r_W_OB_B_radDs

Rotation of the B-Frame relative to the O-Frame

B-Frame is the notation frame

For the notation used for basic constants and other parameters, see reference [9].

11.2 Frames and Transformations

❖ ECI (Earth-Centered Inertial)

Index	I
Origin	Center of the Earth
Translation	With solar system, around the sun
Rotation	None
x_I-Axis	In equatorial plane, in vernal equinox direction
y_I-Axis	In equatorial plane, to form a right hand system with x-axis and z-axis
z_I-Axis	Rotation axis of the Earth

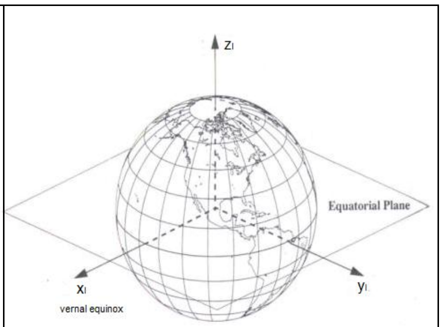


Figure 11-1 ECI Frame

Table 11-1 ECI Frame

❖ ECEF (Earth-Centered Earth-Fixed)

Index	E
Origin	Center of the Earth
Translation	With ECI-System
Rotation	Earth Rotation (around z-axis with Earth angular rate $\vec{\boldsymbol{\omega}}^{IE}$)
x_E-Axis	In equatorial plane, points through Greenwich Meridian
y_E-Axis	In equatorial plane, to form a right hand system with x-axis and z-axis
z_E-Axis	Rotation axis of the Earth

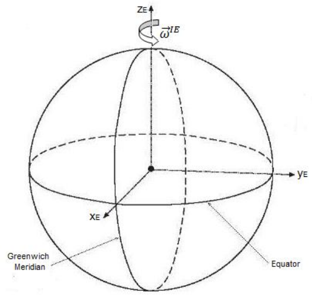


Figure 11-2 ECEF Frame

Table 11-2 ECEF Frame

❖ WGS 84 (World Geodetic System 1984)

The WGS84 Coordinates are defined by two angles and the height above ellipsoid

Index	WGS84
Geodetic Longitude $\boldsymbol{\lambda}$	Angle measured in the meridian plane between the zero meridian plane and the meridian plane of point P. Range: $-\boldsymbol{\pi} \leq \boldsymbol{\lambda} \leq \boldsymbol{\pi}$
Geodetic Latitude $\boldsymbol{\varphi}$	Angle measured in the meridian plane of the point P between the equatorial plane and the surface normal of point P. Range: $-\boldsymbol{\pi}/\mathbf{2} \leq \boldsymbol{\varphi} \leq \boldsymbol{\pi}/\mathbf{2}$
Geodetic Height $\boldsymbol{h}$	Height above the WGS84 ellipsoid measured along the surface normal

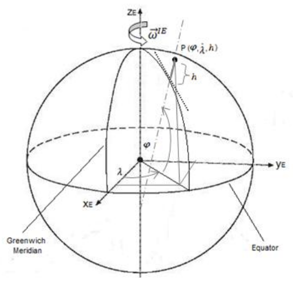


Figure 11-3 WGS84 Coordinates

Table 11-3 WGS84 Frame

❖ O-Frame (NED: North-East-Down Frame)

Index	O (NED)
Origin	Reference point of the aircraft
Translation	With Aircraft reference point
Rotation	Rotate with transport rate to keep NED alignment $\vec{\omega}^{EO}$
x_O-Axis	Parallel to the local geoid surface, pointing to the geographic north pole
y_O-Axis	Parallel to the local geoid surface, pointing east to form a right hand system with x-axis and z-axis
z_O-Axis	Pointing downwards perpendicular to the local geoid surface

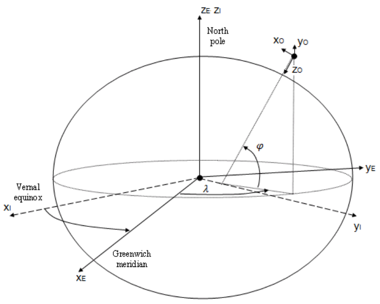


Figure 11-4 NED Frame

Table 11-4 NED Frame

❖ B-Frame (Body Fixed Frame)

Index	B
Origin	Reference point of the aircraft
Translation	With Aircraft reference point
Rotation	With the rigid body aircraft
x_B-Axis	Pointing towards the aircraft nose in the symmetry plane
y_B-Axis	Pointing to the right (starboard) wing to form an orthogonal right hand system
z_B-Axis	Pointing downwards in the symmetry plane of the aircraft, perpendicular to the x- and y-axis

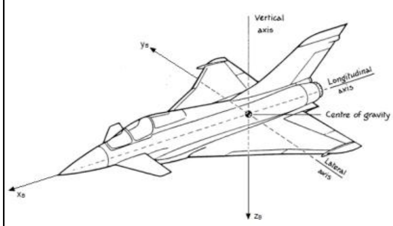


Figure 11-5 Body Frame

Table 11-5 Body Frame

Special Thanks

Voglio ringraziare prima di tutti il Prof. Eugenio Denti per avermi seguita ed aiutata durante il lavoro di tesi, superando spesso le distanze, e per il tempo che mi ha pazientemente dedicato in queste ultime settimane.

Ringrazio il Prof. Giovanni Mengali per la sua disponibilità e per aver spesso chiarito i miei dubbi.

Ringrazio il Prof. Florian Holzapfel per avermi accolta nel suo istituto e per avermi spronata ad integrarmi rapidamente.

Ringrazio l'Ing. Stefan Hager per la sua infinita pazienza e per il suo incomparabile aiuto.

Grazie all'amico Paolo Antonelli per avermi dedicato il suo tempo quasi senza conoscermi.

Un sentito ringraziamento va a tutta la mia famiglia: grazie per il sostegno economico senza il quale non avrei potuto intraprendere questo percorso ma soprattutto per aver sempre partecipato alla mia vita con estrema discrezione, senza mai interferire con le scelte più importanti; grazie per aver gioito con me per ogni piccolo successo e per avermi sempre ricordato, quando era necessario, che nella mia vita ci sono tante altre cose oltre allo studio matto e disperato. Sentirvelo dire mi è sempre servito.

Ringrazio gli amici "pisani" con i quali ho condiviso la maggior parte di questi anni: grazie a chi è sempre stato al mio fianco, a chi ha fatto in modo che avessi sempre qualcosa da mangiare, a chi mi ha aiutata a traslocare, a chi mi ha tirata su di morale, a chi mi ha convinta a prendermi meno sul serio, a chi non ha mai fatto un passo indietro. Grazie per la vostra disponibilità, per il vostro calore e per il vostro affetto; in questi anni siete stati per me una seconda famiglia e sarete insostituibili.

Un ringraziamento speciale va alle ragazze con le quali ho convissuto in questi anni ma soprattutto a chi ha fatto in modo che la nostra casa non fosse un luogo di passaggio ma un posto in cui potessi trovare sempre un confronto o una semplice parola di conforto: grazie alle ex coinquiline alle quali oggi sono legata da amicizia e alla carovana delle nuove coinquiline. Grazie a chi tra voi ha avuto la pazienza di

volermi conoscere e capire a fondo. Conserverò un bel ricordo di ognuna o almeno ci proverò.

Grazie a tutte le persone che ho incontrato in questi anni, a chi è ancora nella mia vita o a chi è stato solo di passaggio. Ognuno, in fondo, mi ha insegnato qualcosa.

Ringrazio soprattutto chi ha reso unici questi ultimi anni con le sue attenzioni, con la sua pazienza, con la sua discrezione e con il suo instancabile sostegno: grazie perché se non ho cambiato strada ed oggi sono qui lo devo soprattutto a te.

Ringrazio questa città, tanto odiata e tanto amata, per aver fatto da cornice ad alcuni dei momenti più belli ed importanti della mia vita. In ogni caso, resterà nel mio cuore.

Inoltre ringrazio me stessa per aver avuto il coraggio di lanciarmi una sfida difficile e per averla vinta.

Infine ringrazio tutte le persone che oggi sono qui a condividere con me questo momento e chi, invece, non ha potuto esserci ma l'avrebbe voluto.