Politecnico di Torino

Facoltà di Ingegneria
Corso di Laurea Magistrale in Ingegneria Aerospaziale

Tesi di Laurea

Development of atmosphere and altimeter models for Mars mission simulation and integration in dSPACE real-time environment

Relatori:
Prof. Giorgio Guglieri
Prof. Paolo Maggiore
Ing. Antonio Tramutola

Candidato:
Elio Miscio

Luglio 2018

Development of atmosphere and altimeter models for Mars mission simulation and integration in dSPACE real-time environment

Elio Miscio

Contents

List of Figures

List of Tables

List of Acronyms

AU	Astronomical Unit
DEM	Digital Elevation Model
DM	Descent Module
DOF	Degrees Of Freedom
DVV	Development, Verification & Validation
EDL	Entry, Descent & Landing
EIP	Entry Interface Point
EOT	Equation Of Time
EUV	Extreme Ultraviolet
FB	Body Axis System
FEIC	Feature Extraction and Integrated circuit
FI	Planet Inertial Frame
FPGA	Field Programmable Gate Array
FR	Planet Fixed Rotating Frame
FS	Flight Segment
GC	Guidance and Control
GCM	General Circulation Model
GNC	Guidance, Navigation & Control
GS	Ground Segment

HDA	Hazard Detection and Avoidance
HIL	Hardware In Loop
HW	Hardware
IMU	Inertial Measurement Unit
JD	Julian Date
Lidar	Light Detection and Ranging
LMT	Local Mean Time
Ls	Solar Longitude
LTF	Local Terrain Frame
LTST	Local True Solar Time
LUT	Look-Up Table
MCD	Mars Climate Database
MGM	Goddard Mars Model
MGS	Mars Global Surveyor
MOLA	Mars Orbiter Laser Altimeter
MVM	Mode and Vehicle Management
NASA	National Aeronautics and Space Administration
NAV	Navigation
NetCDF	Network Common Data Form
OBC	On-Board Computer
PANGU	Planet and Asteroid Natural scene Generation Utility
RAMSES	Rule-Based Analytic Asset Management for Space Exploration Systems
RWEE	Real World Environment Emulator
S/C	Spacecraft

SDF	System Description File
SPW	SpaceWire
SW	Software
TASI	Thales Alenia Space Italy
TM/TC	Telemetry/Telecommand
VBNAV	Visual Based Navigation
VL1	Viking Lander 1

Chapter 1

Abstract

This thesis deals with the development of Mars atmosphere model and altimeter sensor model in Matlab/Simulink environment, using both S-Function and Embedded Matlab function technics. They have been integrated in a model-based simulator framework together with guidance, navigation and control (GNC) functions and other sensors, actuators models and spacecraft dynamics setting up the simulation environment suitable for planetary Entry, Descent and Landing missions.

The main objective of the thesis is: to generate a C code from the above mentioned models using the Real-Time Workshop tool, to cross compile it into an executable application for the real-time hardware and to download the application into the dSPACE platform in order to perform closed loop simulations with an avionics test bench. Results of the test campaign led to identify models' performance, requirements and constraints.

Chapter 2

Introduction

This thesis has been developed in the Avionic Unit of Thales Alenia Space, Turin (Italy), from October 2017 to June 2018. It has been carried out working on a flight simulator development project in collaboration with the Data Systems and Communication group. All documents and datafiles used to create the two models have been provided by the company and shall not be divulged to any third party without its permission.

2.1 Mission Scenario

The Mars mission scenario consists in an Entry, Descent and Landing mission designed to perform landing at any latitude between 15 degrees South and 45 degrees North, and at any longitude. According to mission requirements, the landing of Descent Module is set to occur during Martian daylight and far away from the global dust storm season. In order to fulfill lander safety constraints, the maximum vertical and horizontal velocity at landing required to be respectively equal to 2.5 m/s and 2 m/s, while maximum angle of the longitudinal axis with respect to the local vertical has to be within 2 deg half cone.

The EDL manoeuvre begins when the Entry Interface Point (EIP), nominally expected at an altitude of 120 km, is detected through acceleration measurement by IMU. During this phase the navigation is based only on gyroscopes and accelerometers. The atmospheric drag gradually reduces the velocity of the DM, acting on its front shell, while the thermal protection absorbs the heat load. No guided entry actuation is foreseen during this phase due to lack of controllability, although the DM is expected to be passively stabilized by the increasing aerodynamic torques to the equilibrium null angle of attack.

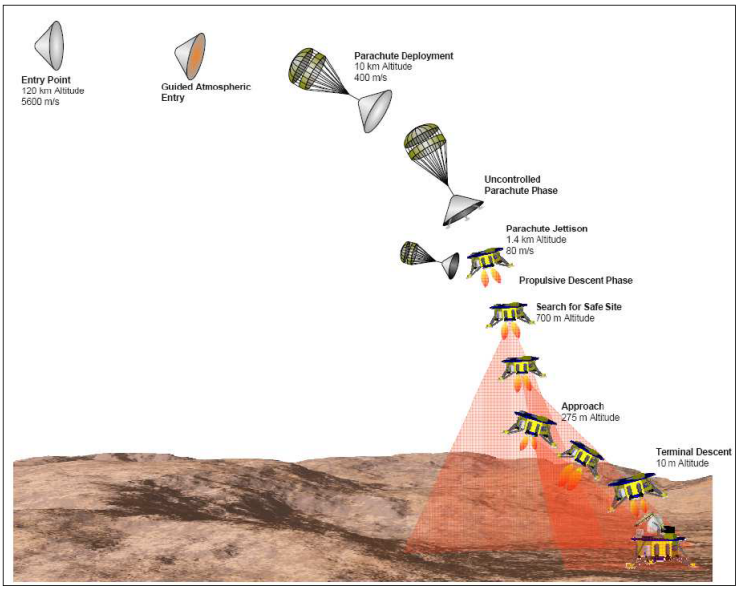


Figure 2.1: Mission Entry, Descent and Landing scenario (*Courtesy of* TASI)

The next phase is driven by Guided Entry algorithms, which follows through bank angle control the defined trajectory minimising dispersion due to atmosphere uncertainty. This phase is triggered by the detection of a set lift value, corresponding to a sufficient authority control, that represents the start of controllability window during Entry phase, i.e. the beginning of the bank angle reversal manoeuvres. By controlling the bank angle, thus the component of the lift vector in the vertical direction, the spacecraft is able to fly through denser atmosphere in order to reduce its velocity, taking advantage of atmospheric friction. By adequately driving the capsule bank angle through the atmosphere, the parachute deployment velocity conditions can be met at a specific location in terms of altitude, longitude and latitude.

Parachute deployment algorithm, based on accelerometer measurements and timers, is designed to trigger at about 10 km altitude and 400 m/s velocity. Parachute phase is a passive phase with the objective to decrease DM velocity and start the controlled landing. The Lander follows a first uncontrolled trajectory, reaching a steady-state descent velocity of approximately 80 to 100 m/s. After a defined elapsed time from parachute deployment (almost 40s), the heat shield is released in order to use sensors for altitude estimation such as camera, LIDAR and altimeter.

The final descent phase includes both the functionalities of Hazard Detection and Avoidance function and the GNC function for attitude control. When the terrain relative altitude computed by sensors reaches the trigger value of 2 km, the backshell with the attached parachute is jettisoned and 2 seconds later the

propulsion system is turned on. The main task of the propulsive descent phase is performing the Hazard Detection and Avoidance function in order to assure the landing in a safe area in terms of presence of hazards, fuel cost, illumination and slope. At 275 m of altitude, the spacecraft commits to choose a landing site and updates its trajectory to aim for that site. Right above the target, at 10 m altitude, Terminal Descent (TD) is triggered and the S/C follows a constant vertical velocity trajectory until touchdown.

The following table summarises each phases of the manoeuvre:

Mission Phase	**Description**	**Triggering Principle**
Atmospheric Entry	"Unguided" Atmospheric Entry	Acceleration based, EIP detection (120 km altitude)
Guided Entry	Guided Entry (bank reversal)	Acceleration based, deceleration peak detection (80 km altitude)
Parachute Phase	Uncontrolled Parachute Phase	G-Activated Timer based (10 km altitude)
Descent Phase	Powered Descent Phase	Relative Altitude based (2 km altitude)
	Safe Site Search	Relative Altitude based (700 m altitude)
	Approach (Safety Retargeting)	Safe Site Identified (275 m altitude)
	Terminal Descent	Altitude-based (10 m altitude)

Table 2.1: Description of the mission phases

2.2 Brief History of Mars Exploration and Technology Improvements

The landing missions on Mars has experienced a high failure rate due to the complexity in performing a self-guided soft landing. According to a NASA historical log paper [6], only the 50% of landing procedures has been carried out.

This section provides a brief description of the successful landing missions on Mars surface and the related technology improvements. All these missions had the common goal to find an answer to two key questions:

1. Is there life on Mars?

2. Is Mars potentially habitable?

Exploration missions, beginning in the late 20th century, have collected a large quantity of scientific data, focused mainly on characterising Mars' climate and geology, determining whether life ever arose and preparing for future human exploration.

MARS 3 (1971)

Figure 2.2: Illustration of MARS 3 lander

Mars 3 was an unmanned space probe of the Soviet Mars program, launched from Tyuratam (U.S.S.R.) in 1971 [1]. Mars 3 lander is well-known to became the first spacecraft to achieve soft landing on Mars surface on 2 December 1971, five days after its twin spacecraft Mars 2 crashed on martian surface. The spacecraft consisted of a bus/orbiter module and an attached descent/lander module. The primary scientific objective of the Mars 3 descent module was to perform a soft landing on Mars, return images from the surface, and return data on meteorological conditions, atmospheric composition, and mechanical and chemical properties of the soil. It was equipped with a spherical landing capsule, a conical aerodynamic braking shield, a parachute system and retrorockets. the attitude control was provided by an automatic control system consisting of gas micro-engines and pressurized nitrogen containers. The main and auxiliary parachutes, the engine to initiate the landing, and the radar altimeter were mounted on the top section of the lander.

The instrumentation consisted of two television cameras with a 360 degree view of the surface, a spectrometer to study atmospheric composition, sensors to evaluate temperature, pressure and wind, and devices to measure mechanical and chemical properties of the surface, searching for organic materials and signs of life. The communication with the orbiter was provided by four aerials, via an on-board radio system. The equipment was powered by batteries which were charged by the orbiter before the separation. After the landing, the transmission between the capsule and the orbiter lasted only 20 seconds, as it was stopped for unknown

reasons and no further signals were received at Earth from the martian surface. The probable cause of the failure may have been related to the extremely powerful martian dust storm taking place that time. This storm could have damaged the communications system.

Viking 1 & 2 (1975)

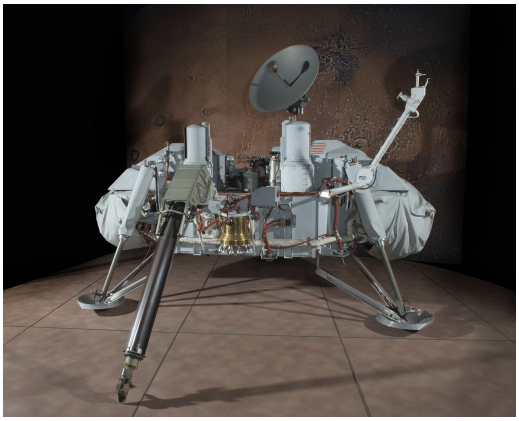


Figure 2.3: Illustration of Viking lander

NASA's Viking Project was the first U.S. mission to land a spacecraft safely on the surface of Mars and return images of the surface. Viking 1 and Viking 2 were two identical spacecrafts, consisting of a lander and an orbiter, launched both from Cape Canaveral (Florida) in 1975 and landed on Mars surface respectively on 20 July 1976 and 3 September 1976 [12]. Viking landing sites were chosen after studying orbiter photos in order to find safe areas for landers. The primary mission objectives were to obtain high resolution images of the Martian surface, characterize the structure and composition of the atmosphere and surface, and search for evidence of life.

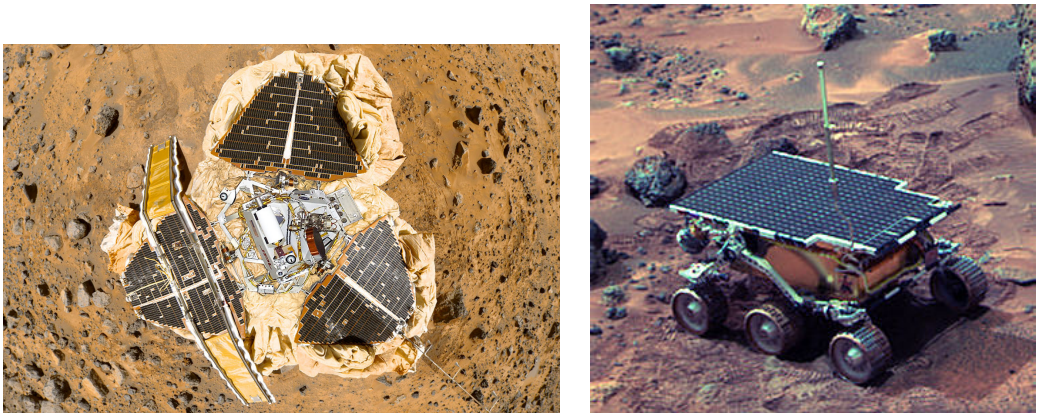
The lander spacecraft was composed of five basic systems: the lander body, the bioshield cap and base, the aeroshell, the base cover and parachute system, and lander subsystems. The body was a basic platform for science instruments and operational subsystems, supported by three landing legs, 1.3 meters long, attached to the short-side bottom corners of the body. The two-piece bioshield, white thermal painted, was a pressurized cocoon that completely sealed the lander from any possibility of biological contamination until Viking left Earth's atmosphere. The aeroshell was an aerodynamic heat shield made of aluminium with an ablative material bonded to its exterior, that burned away to protect the lander from aerodynamic heating at entry temperatures. The interior of the aeroshell contained

twelve small reaction control engines, which maintains the correct attitude during entry phase, an upper atmosphere mass spectrometer and sensors to measure pressure and temperature. The parachute was packed inside a mortar mounted into the base cover. The lander subsystems were divided into six major categories: descent engines, communications equipment, power sources, landing radars, data storage and guidance and control.

Basic power for the lander was provided by two radioisotope thermoelectric generators (RTGs) which provided a long-lived source of electricity and heat on Mars, where sunlight is half as strong as on Earth, and it is non-existing during the Martian night. Besides taking photographs and collecting other science data on the Martian surface, the two landers conducted three biology experiments that discovered unexpected chemical activity in the Martian soil, but provided no clear evidence for the presence of living microorganisms in soil near the landing sites. Furthermore, the landers continuously monitored weather at the landing sites, examining atmospheric cyclic variations and climate change.

Because of the variations in available sunlight, each orbiter and lander operated far beyond their 90 days design lifetime: Viking Lander 1 made its final transmission to Earth on 11 November 1982, while the last data from Viking Lander 2 arrived at Earth on 11 April 1980.

Mars Pathfinder (1996)



(a) *Pathfinder lander* (b) *Sojourner rover*

Figure 2.4: Illustration of Mars Pathfinder

Mars Pathfinder was a NASA robotic spacecraft launched from Cape Canaveral (Florida) in 1996 and landed on Mars' Ares Vallis on July 4, 1997 [7]. It was designed as a technology demonstration of a new way to deliver an instrumented

lander, formally named the Carl Sagan Memorial Station, and the first-ever robotic rover, named Sojourner, to the surface of the red planet. Pathfinder used an innovative method of directly entering the Martian atmosphere, assisted by a parachute to slow its descent through the thin Martian atmosphere and a giant system of airbags to cushion the impact. It was the first time this airbag technique had been used. First transmission received from lander's antenna contained preliminary information about the health of the spacecraft and rover, a first look at the density and temperatures of the Martian atmosphere, and first images of the landing area taken by the lander's camera.

Mars Pathfinder lander includes heat shield, back shell, solar panels, propulsion system, low- and high-gain antennas, parachute, airbags and atmospheric sensors and instruments. The Pathfinder rover measured only 65 centimeters long by 48 centimeters wide by 30 centimeters tall, with a maximum traveling speed on Mars equal to 1 centimeter per second. It was powered by a 0.25-square-meter solar array on its top surface. The rover carried two finger-sized black-and-white cameras in front, a color camera in back, an alpha proton X-ray spectrometer (APXS) for determining the elemental composition of rocks and soil, and a set of experiments for testing material adherence and wheel abrasion. A laser system worked in conjunction with the two forward cameras to detect and avoid obstacles. The robotic rover, capable of autonomous navigation and performance of tasks, communicated with Earth via the lander.

Both the lander and rover carried instruments for scientific observations, able to analyze the Martian atmosphere, climate, geology and the composition of its rocks and soil. During its activities, ended on 27 September 1997, Mars Pathfinder demonstrated the efficacy of new available technologies and returned an unprecedented amount of scientific data, including chemical analysis of rocks and soil, images of water ice clouds in lower atmosphere, and atmospheric statistics.

Spirit & Opportunity (2003)

Spirit and Opportunity were two twin robotic geologists launched from Cape Canaveral (Florida) in 2003 and landed on opposite sides of Mars in January 2004 [11]. They are the two rovers of NASA's Mars Exploration Rover (MER) mission. These robotic explorers, with far greater mobility than the 1997 Mars Pathfinder rover, have trekked for miles across the Martian surface, conducting field geology and making atmospheric observations. Opportunity landed close to a thin outcrop of rocks, where mineral deposits suggested that Mars had a wet history. Spirit explored a plain strewn with volcanic rocks and pocked with impact craters, finding indications that small amounts of water may have leaked into cracks in the rocks and may also have affected some of the rocks' surfaces.

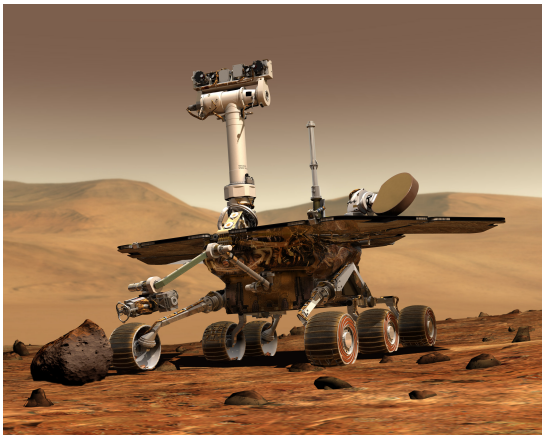


Figure 2.5: Illustration of Spirit rover

The descent maneuver was performed using a parachute fastened to the backshell portion of the aeroshell, opened about two minutes before landing.

About 20 seconds later, the spacecraft jettisoned the heat shield to allow a downward-pointing camera on the lander to take pictures during the final half-minute of the flight in order to estimate horizontal motion. In the final eight seconds before impact, gas generators inflated the lander's airbags. Retro rockets on the backshell fired to halt descent speed, and transverse rockets fired to reduce horizontal speed.

The rovers included some of the design elements of Mars Pathfinder Sojourner: six wheels and a rocker-bogie suspension for driving over rough terrain, solar panels and rechargeable batteries for power, and radioisotope heater units for protecting batteries through extremely cold martian nights. However, each Mars Exploration Rover is more than 17 times as heavy as Pathfinder and more than twice as long and tall. In fact, Spirit and Opportunity housed in addiction mission's main communications, camera and computer functions, as they are able to operate without a lander platform.

Equipped with an innovative geology toolkit, the twin rovers have sent hundreds of thousands of spectacular, high-resolution, full-color images of Martian terrain as well as detailed microscopic images of rocks and soil surfaces to Earth. Their four spectrometers and special rock abrasion tools, never sent before to another planet, have collected plenty of information about the chemical and mineralogical makeup of Martian rocks and soil.

Both rovers exceeded their planned 90-day mission lifetimes by many years. Spirit lasted 20 times longer than its original design until its final communication to Earth on 22 March 2010, while Opportunity still continues to operate more than a decade after launch.

Phoenix (2007)

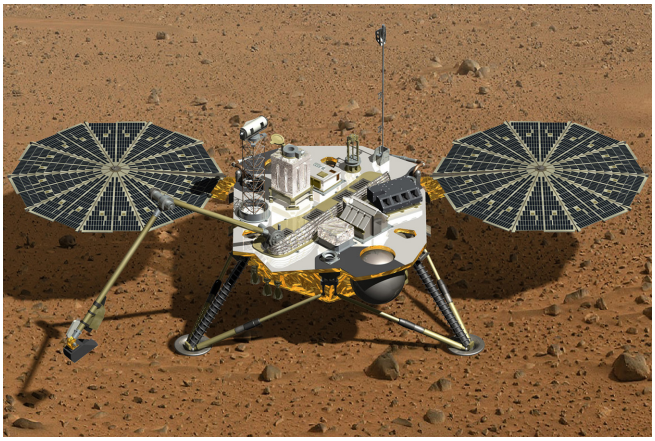


Figure 2.6: Illustration of Phoenix lander

Phoenix was a NASA's robotic spacecraft launched from Cape Canaveral (Florida) in 2007. The Phoenix lander landed on the artic plains of Mars on 25 May 2008 [10].

The main objective of Phoenix mission was the exploration of far northern plains of Mars in order to analyze components of the surface, subsurface and atmosphere.

For what concerns the entry, descent and landing phase, the craft used a heat-generating atmospheric friction, a parachute and descent thrusters. Although air bags add a safety margin for landing on slopes or rocky ground, they were not utilized for Phoenix as that advantage is not relevant for the flat and relatively unrocky terrain targeted for the landing. Therefore, the landing systems weighed less than the systems for earlier Mars missions, helping Phoenix to have a higher ratio science-instrument payload to total launch weight (59/664 kilograms) than any spacecraft that had previously landed on Mars. During the final minutes of planet approach, Phoenix used radar to continually assess and adjust the spacecraft's vertical and horizontal motion. The thrusters then shut off when sensors on the footpads detected contact with the ground.

The lander was set to use a trenchdigging arm and analytical tools to study water believed to be frozen into the soil just below the surface. It checked for the presence of organic compounds as part of an evaluation of whether the site has been a favorable environment for microbial life. On the surface of Mars, the power was guaranteed by a two-wing solar array converting solar radiation to electricity while a pair of rechargeable lithiumion batteries provided power storage. Regarding telecommunications, Phoenix used a UHF radio band signal, relayed by Mars

orbiters, to communicate with Earth. The thermal control was performed using a combination of electrical heaters, thermostats, temperature sensors, blanketing and thermal coatings.

Phoenix lander carried different scientific instruments, including a robotic arm camera, which provided close-up colour images of Martian soil at the landing site, a Surface Stereo Imager, which recorded panoramic views of the surroundings, and a Thermal and Evolved-Gas Analyzer for temperature and heating monitoring.

The lander completed its mission in August 2008, after it had ended all planned science experiments and observations.

Curiosity (2011)

Figure 2.7: Illustration of Curiosity rover

Curiosity was a car-sized rover designed to explore Gale Crater on Mars as part of NASA's Mars Science Laboratory mission (MSL). It was launched from Cape Canaveral (Florida) in 2011 and landed successfully on the floor of Gale Crater on 26 November 2012 [8]. The rover was a real mobile laboratory with the objective of studying the geology and environment of the crater, analyzing samples drilled from rocks or scooped from the ground.

Curiosity carried a payload more than 10 times as massive as those of earlier Mars rovers, being about twice as long and five times as heavy as NASA's twin Mars Exploration Rovers, Spirit and Opportunity, launched in 2003. The spacecraft was designed to steer itself during descent through Mars' atmosphere with a series of S-curve maneuvers similar to those used by astronauts piloting NASA space shuttles. During the three minutes before touchdown, the spacecraft slowed its descent with a parachute, then used retrorockets mounted around the rim of its upper stage. In the final seconds, the upper stage acted as a sky crane, lowering the upright rover on a tether to land on its wheels.

The rover inherited many design elements from earlier rovers, such us six-wheel drive and cameras, but unlike them, it carried equipment to gather and process samples of rocks and soil, distributing them to onboard test chambers inside analytical instruments. Curiosity performed analyses on the first samples of material ever drilled from rocks on Mars, providing the evidence of conditions favorable for life in Mars' early history. On a subsequent drill sample, it was able to determine the age of the rock, showing that the drilled material was 4.2 billion years old and yet had been exposed at the surface for only 80 million years. The instrument for sample analysis included a gas chromatograph, a mass spectrometer and a laser spectrometer. The wide view of its Navigation Camera was used to aid targeting of other instruments and to survey the sky for clouds and dust.

Curiosity is still operational after 6 years of activities.

2.3 Atmosphere and Climate of Mars

This section provides some statistics of Martian atmosphere in order to better understand its physical properties and climatic processes. The atmosphere of Mars is about 100 times thinner than Earth's, and it is composed mostly of carbon dioxide. According to a NASA fact sheet [13], it consists of:

- Carbon dioxide: 95.32 percent
- Nitrogen: 2.7 percent
- Argon: 1.6 percent
- Oxygen: 0.13 percent
- Carbon monoxide: 0.08 percent
- Minor amounts of: water, nitrogen oxide, neon, hydrogen-deuterium-oxygen, krypton and xenon

The average atmospheric pressure on Martian surface is equal to 600 Pa, about 0.6% of Earth's mean sea level pressure, and ranges from a low of 30 pascals on Olympus Mons (21287.4 m) to over 1155 pascals in the depths of Hellas Planitia (-7152 m). The thin air is responsible for the wide, daily swings in temperature, almost 100 degrees.

Temperature depends also on the altitude: the lower atmosphere is a relatively warm region affected by heat from airborne dust and from the ground, the middle atmosphere, containing Mars's jet streams, is instead relatively cold, and the upper

atmosphere (or thermosphere) is a region with very high temperatures, caused by heating from the Sun.

Despite its very low density, 0.02 kg/m^3 on average at surface level, the atmosphere of Mars is still thick enough to support weather, clouds and winds. The current Martian climate is regulated by seasonal changes of the carbon dioxide ice caps, the movement of large amounts of dust by the atmosphere and the exchange of water vapor between the surface and the atmosphere.

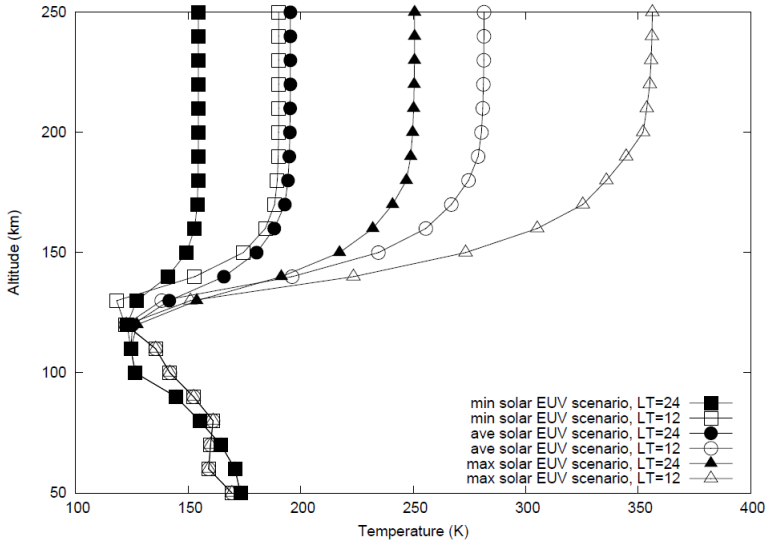


Figure 2.8: Example of atmospheric temperatures as a function of EUV conditions and local time LT (LT=12 midday, LT=24 midnight)

The weather is pretty much the same each day: cold and dry with small daily and seasonal changes in temperature and pressure, plus a chance of dust storms and dust devils. During their activities, Viking landers have measured values of wind equal to 2-7 m/s during summer season, 5-10 m/s during fall season and 17-30 m/s during winter season, where dust storms are likely to happen. As the Earth, seasons on Mars are 3 months long: each of them is defined as spanning 30 degrees in solar longitude [4]. The solar longitude Ls is the Mars-Sun angle, measured from the Northern Hemisphere spring equinox where Ls=0 (figure 2.9).

Ls=90 thus corresponds to northern summer solstice, Ls=180 marks the northern autumn equinox and Ls=270 the northern winter solstice. A martian year is 668.6 sols (martian solar days) long and a sol is 88775.245 seconds long. Due to the eccentricity of Mars' orbit, martian months are thus from 46 to 67 sols long, as shown in the table below.

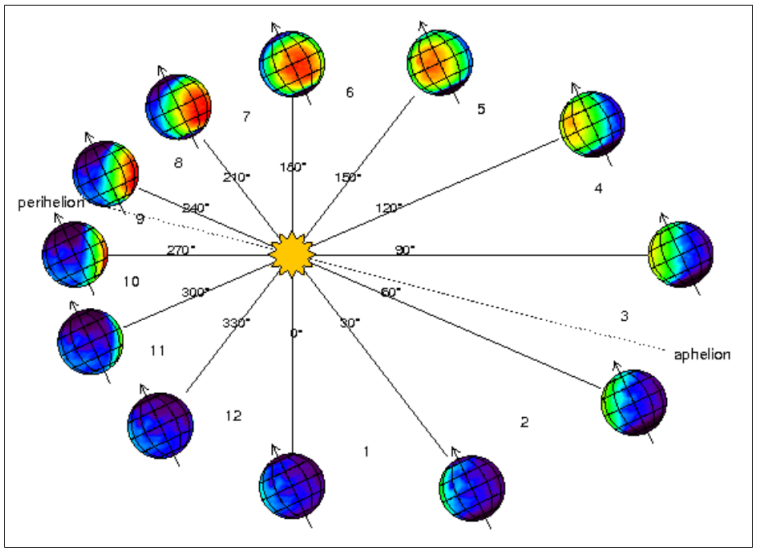


Figure 2.9: Illustration of Martian seasons

Month number	**Ls range (degrees)**	**Duration (sols)**	**Peculiarity**
1	0-30	61.2	Northern Hemisphere Spring Equinox at Ls=0
2	30-60	65.4	
3	60-90	66.7	Aphelion (largest Sun-Mars distance) at Ls=71
4	90-120	64.5	Northern Hemisphere Summer Solstice at Ls=90
5	120-150	59.7	
6	150-180	54.4	
7	180-210	49.7	Northern Hemisphere Summer Solstice at Ls=180 Dust Storm Season begins
8	210-240	46.9	Dust Storm Season
9	240-270	46.1	Perihelion (smallest Sun-Mars distance) at Ls=251 Dust Storm Season
10	270-300	47.4	Northern hemisphere Winter Solstice at Ls=270 Dust Storm Season
11	300-330	50.9	Dust Storm Season
12	330-360	55.7	Dust Storm Season ends

Table 2.2: Martian months

Chapter 3

Description of the Simulator Architecture and Environment

3.1 Simulator Overview

The application software used to reproduce the mission scenario and operations is a model-based simulator design framework developed in Matlab/Simulink environment. It includes different kinds of models, such as environment, Guidance Navigation & Control systems, sensors and dynamics, able to simulate and analyze properly an Entry, Descent and Landing mission on a planet. A mathematics library containing models of Vector and Quaternion manipulation, Coordinate conversions, Matrix manipulation and Source model blocks such as random number and noise generators, is also available in the simulator model repository.

Within the simulator, the user can easily build models and system templates or modify and customize existing models and templates to meet the project needs. Furthermore, Simulink allows code written in C or other languages to be implemented as a block using S-functions, enabling the simulator to run code ported from another environment with a minimum of rework or integration effort. The automatic generation of C code is based on TargetLink from dSPACE.

The software and hardware components used to run appropriately the simulator are listed below.

Software:

- Windows 10 (64-bit)
- Matlab R2009b (32-bit)
- Simulink 7.4
- Microsoft Visual C++ 2008 Express, version 9.0

- Java version 8

Hardware:

- CPU Intel quad core i7-8550U 1.80 GHz
- 16 GB RAM (minimum RAM required: 2 GB)
- 2 TB Hard Disk (minimum space required: 2 GB)

The simulator consists of several layers which offer a methodical and clear structure to the user, allowing him to simply identify models and their connections within it. At the top layer of the simulator Simulink model, we can find two main templates: the *Body Template* and the *Environment Template*. The first contains all the model components relevant to the main elements of the space vehicle such as dynamics, deployables structure, sensors, actuators and GNC, while the second contains all the model components relevant to the environment of the planet such as atmosphere and gravity field.

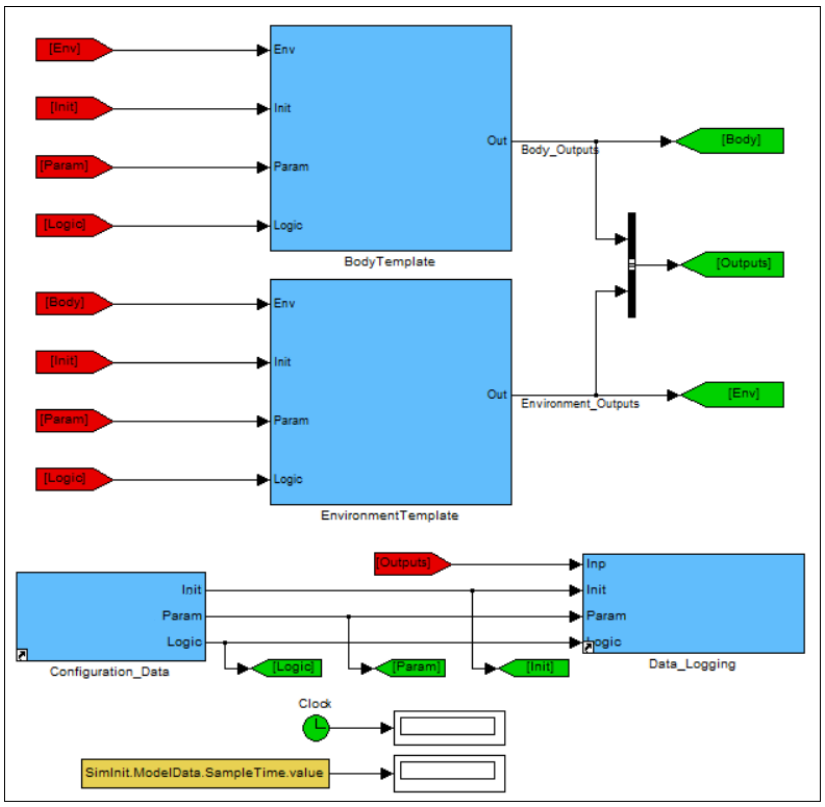


Figure 3.1: Simulator top level structure

In addiction, the *Configuration_ Data* block is set to include the values of all parameters and initial conditions of models, while the *Data_ Logging* is introduced to define a location from where the entire simulation model inputs and outputs can be accessed and post-processed. Communication between the various blocks is realized using *From/Goto* blocks from the Signal Routing library of Simulink.

3.2 Simulator Architecture

Blocks included inside the simulator implement a physical and mathematical representation of:

- ENVIRONMENT
  - Atmosphere
  - Gravity field
- LANDER
  - Actuators
  - Deployables
  - Dynamics and structure
  - Touchdown and Simulation Ending Condition
  - Altimeter
  - Camera
  - Inertial Measurement Unit (IMU)
  - LIDAR
- ON BOARD SOFTWARE
  - Feature Extraction and Integrated circuit (FEIC)
  - Vision Based Navigation (VBNAV)
  - Navigation (NAV)
  - Mode and Vehicle Management (MVM)
  - Guidance and Control (GC)
  - Hazard Detection and Avoidance (HDA)

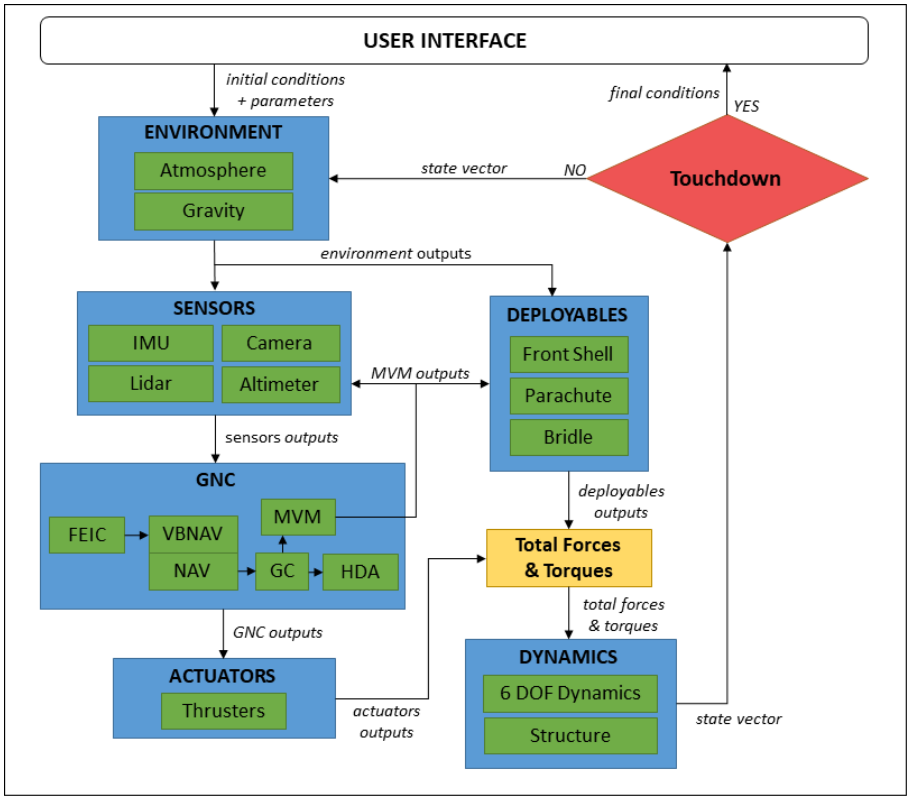


Figure 3.2: Block diagram of simulator architecture

In order to perform simulations, the user must provide as inputs the initial conditions (entry point, Julian Date,...) and parameters (planet constants, orbital elements, vehicle dimensions,...) to the simulator. At the end of simulation, it is possible to observe the final conditions (landing point, state vector of lander,...) and the values of variables logged during the simulation.

The path of signal flow through the simulator models, shown in figure 3.2, is described below.

First of all, the state vector of S/C, i.e. position, velocity and acceleration, is passed to the Environment template, which contains Gravity field model and Atmosphere model. From the coordinates of S/C and planet parameters, the Gravity model evaluates the direction and the module of gravitational acceleration vector while the Atmosphere model computes the climatic conditions for a given position and time (Julian Date) using the Mars Climate Database.

Subsequently, outputs of sensors are calculated: the IMU model, composed of accelerometers and gyrometers, provides measured body linear and angular acceleration using lander state and gravity vector, the spacecraft Camera model takes a perfect image source as input and adds lensing effects, dead pixels, noise

and quantisation, the Lidar model measures the distance to the ground taking into account the unit orientation, the boresight and scan pattern, and the radar Altimeter model estimates the spacecraft-ground distance by simulating the radar beam and reflected signal.

The Deployables template consists of Front Shell, Parachute and Bridle models. The aerodynamic forces and torques of each of these models are computed from the outputs of Environment template.

The GNC template contains all the logic and navigation blocks able to control the movement of the lander. The FEIC model simulates the detection and tracking of interest points from camera sensor images, the VBNAV and NAV models take as inputs the generated interest points and outputs of sensors to provide an estimation of the current state of the lander, the GC model implements guidance laws based on navigation measurements in order to send a torque command to the thrusters for attitude control, and the HDA model uses the sensor data and the outputs of guidance to determine the current safe landing site. The MVM block takes into account the GNC outputs to manage the mission logic flags which enable/disable sensors and deployables as they are required over the trajectory.

The Actuators model takes as input the command from the GNC block and parameters such as number and geometry of thrusters, and computes the resultant force and torque vector.

The Dynamics template contains 6 DOF Dynamics and Structure models. The first takes as inputs the total forces and torques vector, the mass and the inertia matrix of the lander, and gives in output the position and the attitude of S/C. It makes use of the classical 6 d.o.f. equations of rigid body motion:

Translation

$$\begin{cases} F = ma \\ \dot{v} = a \\ \dot{x} = v \end{cases}$$

Rotation

$$\begin{cases} T = \frac{d(I\omega)}{dt} = I\frac{d(\omega)}{dt} + \omega \times I\omega \\ \frac{d(\omega)}{dt} = I^{-1}(T - \omega \times I\omega) \\ \frac{dq}{dt} = \frac{d}{dt}\begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = -\frac{1}{2}\begin{bmatrix} 0 & \omega_1 & \omega_2 & \omega_3 \\ -\omega_1 & 0 & -\omega_3 & \omega_2 \\ -\omega_2 & \omega_3 & 0 & -\omega_1 \\ -\omega_3 & -\omega_2 & \omega_1 & 0 \end{bmatrix}\begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \end{cases}$$

where F and T are the Forces and Torques applied on the body, expressed respectively in an Inertial Reference Frame and Body Fixed Reference Frame, m and I are the body mass and inertia matrix, x and v are the position and velocity

vectors expressed in an Inertial Reference Frame, ω is the angular velocity vector of the Body Fixed Reference Frame with respect to the Inertial Reference Frame, q is the attitude quaternion expressing the orientation of the Body Fixed Reference Frame with respect to the Inertial Reference Frame.

The Structure model calculates the evolving Centre of Mass and Inertia of the vehicle taking into account the mass evolution and the fuel sloshing masses.

Once the new position and attitude of S/C are obtained, the Touchdown model verifies if each landing leg extremity is above or below the surface, using the local surface height and normal vector. The touchdown represents the loop termination condition of the simulator.

Regarding the two models discussed in this thesis, it is appropriate to clarify their influence on other models within the simulator. The Altimeter range is furnished to the Navigation block to estimate the altitude of the lander. The Atmosphere model provides the atmospheric conditions to the Front Shell model and Parachute model. Wind is taken into account in the computation of the aerodynamic angle while density, temperature, viscosity and speed of sound are needed to evaluate the heat flux acting on the aeroshell and the related ablation rate. Pressure and dynamic pressure $\frac{1}{2}\rho v^2$ are used to compute both front shell and parachute aerodynamic forces.

3.3 Reference Frames

All coordinate systems used by the two models are defined in this section. The coordinates of S/C expressed in every frame and the related quaternions are provided by the simulator for each simulation step. The models need basically 3 kinds of reference frames:

1. **Inertial Frame (FI)**, inertial coordinate frame of planet Mars;

2. **Planet Fixed Frame (FR)**, rotating coordinate frame of planet Mars;

3. **Body Axis System (FB)**, body fixed frame of the Descent Module;

With regard to the Inertial Reference Frame, the Reference Epoch is the 12 p.m. (midday) of 1 January 2000 (JD 2451545.0, or J2000).

The Inertial Frame (FI) is defined by taking a "snapshot" of the orientation of a particular set of orthogonal axes at the Reference Epoch. It is formed by attaching the Planet Mean Equator and Prime Meridian of Epoch. The origin is located at Mars' centre of mass and the axes are defined as follow:

- X_I axis: from the Mars centre along the direction of the prime meridian[1] of epoch, in the Mars mean equatorial plane
- Y_I axis: completes the right-handed system
- Z_I axis: points to Mars' spin axis (North Pole)

Like the Inertial Frame, the origin of Planet Fixed Frame (FR) is at Mars' centre of mass. The reference plane is the planet mean equator and the reference direction is the intersection of Mars' prime meridian with the reference plane. The frame wanders with respect to the instantaneous rotation axis of the planet. The axes are defined as follow:

- X_R axis: from the Mars centre along the direction of Mars' prime meridian in the mean equatorial plane
- Y_R axis: completes the right-handed system
- Z_R axis: points to Mars' spin axis (North Pole)

Knowing the angle α_0 between FI and FR at the beginning of simulation, the rotation α about the axis Z, for a given simulation time t, can be computed as:

$$\alpha = \alpha_0 + \omega_M \cdot t \tag{3.1}$$

where ω_M (rad/s) is the rotational rate of planet Mars. The quaternion between FI and FR can be then easily obtained:

$$q_{FI/FR} = [\cos(\alpha/2),\ 0,\ 0,\ \sin(\alpha/2)] \tag{3.2}$$

The Body Axis System (FB) describes the vehicle motion during its flight. The origin is located at the centre of mass of the Descent Module. The reference plane is the plane that passes through the origin and parallel to the plane of symmetry of the Descent Module. The axes are defined as follow:

- X_B axis: lies in the reference plane, parallel to the Descent Module axis of symmetry and positive backward to the front aeroshell
- Y_B axis: normal to the reference plane and positive to the right of the X_B axis
- Z_B axis: lies in the reference plane, completes the right-handed system

The relative position and orientation of Body Axis System with respect to planet frames are evaluated during the simulation by the dynamics model.

[1] The prime meridian of Mars is defined by the crater Airy-0.

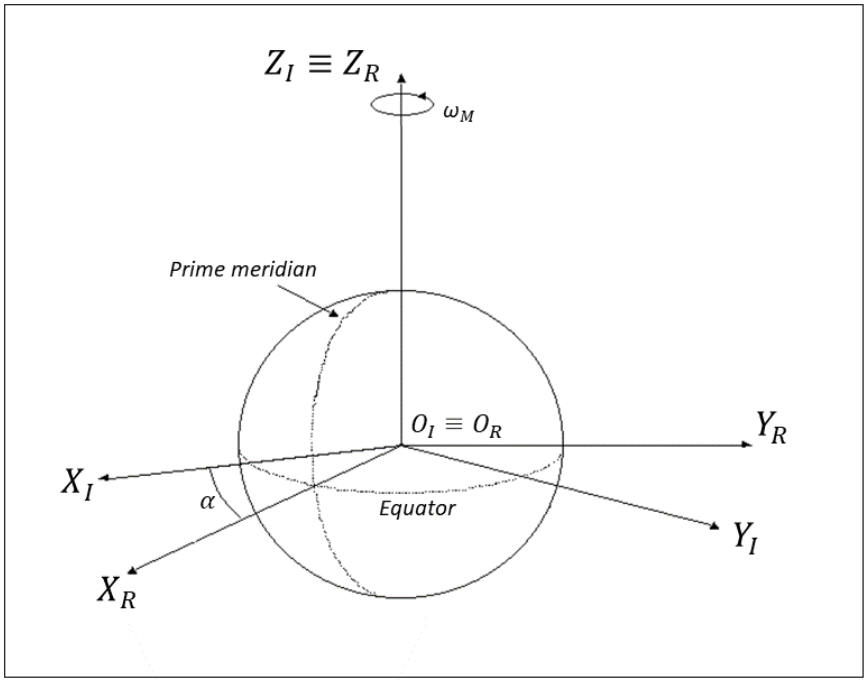


Figure 3.3: Inertial Frame (FI) and Fixed Frame (FR) of Mars

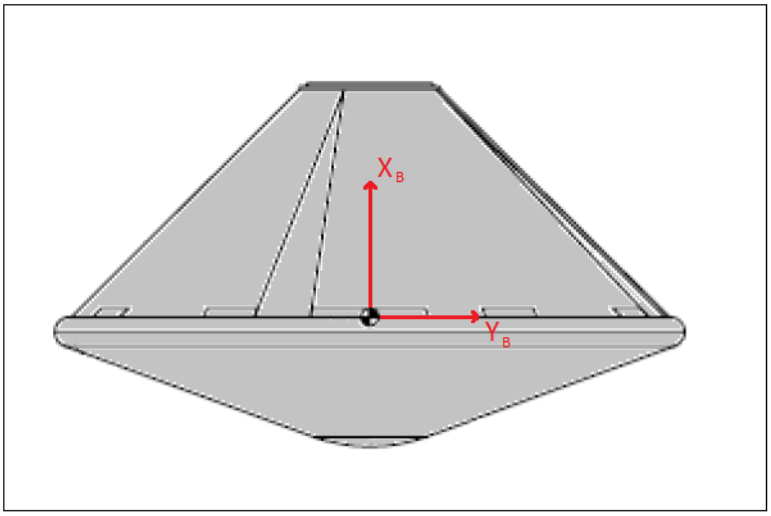


Figure 3.4: Body Axis System (FB) of Descent Module

Chapter 4

Atmosphere Model

This chapter provides a description of the development process of Mars Atmosphere model. It is divided into four sections, representing the main phases of the project: the first section describes the Mars Climate Database, used by the Atmosphere model to retrieve meteorological variables, the second section discusses mathematical methods and physical models contained in it and the last two sections explain the model creation in Simulink environment and its validation.

4.1 Mars Climate Database

Before discussing the mathematical formulation of the model, it is appropriate to give a description of the available Mars Climate Database (MCD), focusing on its contents and how data are stored within it. The MCD contains the atmosphere statistics compiled from state-of-art General Circulation Model (GCM)[1] simulations of the Martian atmosphere, validated using available observational data.

The GCM is developed at Laboratoire de Météorologie Dynamique du CNRS (Paris, France) in collaboration with the Open University (UK), the Oxford University (UK) and the Instituto de Astrofisica de Andalucia (Spain) with support from the European Space Agency (ESA) and from the Centre National d'Etudes Spatiales (CNES) [4].

4.1.1 Description of Database Contents

The statistics included in MCD are stored on a $5.625° \times 3.75°$ longitude-latitude equispaced grid, from the surface up to an approximate altitude of 300 km.

[1] A General Circulation Model is a type of climate model, representing physical processes in the atmosphere and land surface.

Therefore, the grid dimension is 64 × 49 in longitude × latitude: longitudes range from -180.0° to 174.375° in steps of 5.625° (from West to East) and latitudes from 90° to -90° in steps of 3.75° (from North to South). The main data are: temperature, density, wind, pressure, radiative fluxes, atmosphere composition and gases concentration, CO_2 ice surface layer, turbulent kinetic energy, dust optical depth.

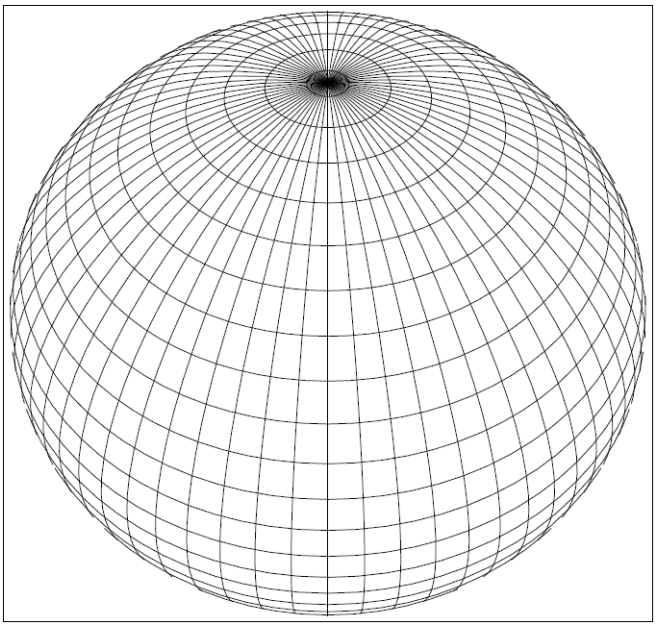


Figure 4.1: Database grid

To store variables vertically, a hybrid coordinate is utilized, in which vertical levels l are at pressure P:

$$P(l) = aps(l) + bps(l) \cdot Ps$$

where Ps is the surface pressure, coefficients *aps(l)* and *bps(l)* are respectively hybrid pressure and hybrid sigma levels. The vertical profile of atmospheric variables consists of 50 levels divided in lower atmosphere (l=1,...,30) and thermosphere (l=30,...,50). This is due to the fact that only the thermosphere is affected by solar EUV input, unlike the lower part of the atmosphere.

As shown in figure 4.2, *aps* and *bps* are prescribed coefficients such that near the surface (small values of l) levels are basically terrain-following sigma coordinates, while at high altitude (large values of l) the vertical levels are pressure levels.

Regarding the temporal structure, data are stored along 12 martian months: each of these is defined as a span of 30° in solar longitude.

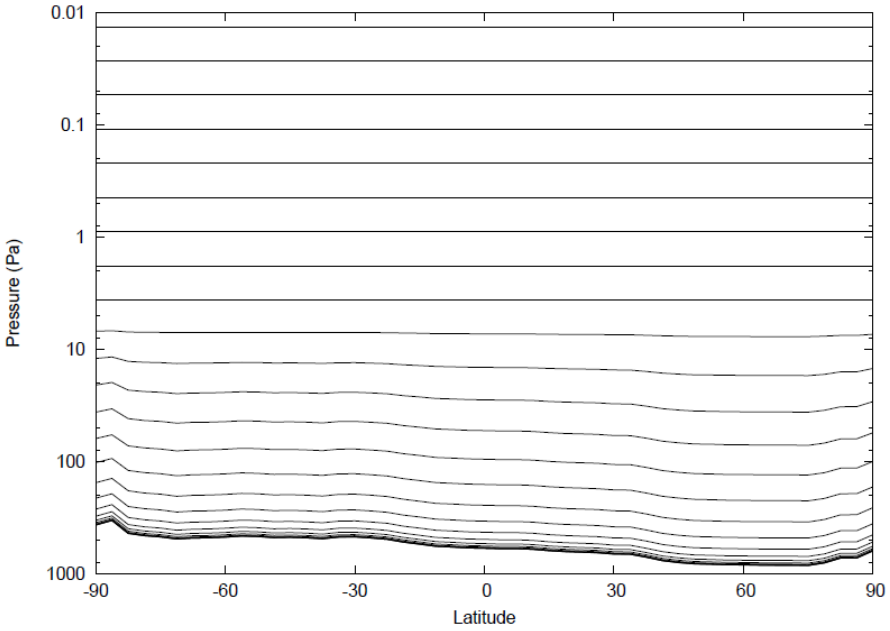


Figure 4.2: Illustration of the vertical hybrid coordinate

The value of every variable is collected 12 times a day, computed in the centre of the month (Ls=15°, 45°, 75°...). In other words, at every grid-point, the database contains 12 "typical" days, one for each month, giving a comprehensive representation of the annual and diurnal cycles. The database reference time is Mars Universal Time (which is simply "prime meridian time", i.e. the local time at 0° longitude) and data is stored every 2 martian hours, from 2 to 24 hours.

Furthermore, the MCD provides eight different combinations of dust and solar scenarios. There are two main reasons which affect the behaviour of Mars atmosphere: the irregular amount and distribution of suspended dust and the EUV input from the Sun, which varies significantly on an 11 year cycle. The database proposes 4 dust scenarios:

1. The **Mars Year 24 (MY24)** scenario is designed to mimic Mars as observed by Mars Global Surveyor from 1999 to 2001, a martian year supposed to be representative of one without global dust storm. The MY24 scenario includes 3 solar EUV conditions: solar minimum, solar average, solar maximum.

2. The **cold scenario** is characterized by an extremely clear atmosphere and a solar minimum thermosphere.

3. The **warm scenario** corresponds to a dusty atmosphere scenario (but not a global dust storm) and a solar maximum thermosphere.

4. The **dust storm scenario** represents Mars during a global dust storm. It is available only when such storms are likely to happen, i.e. during northern fall and winter (Ls=180-360), and includes 3 solar EUV conditions: solar minimum, solar average, solar maximum.

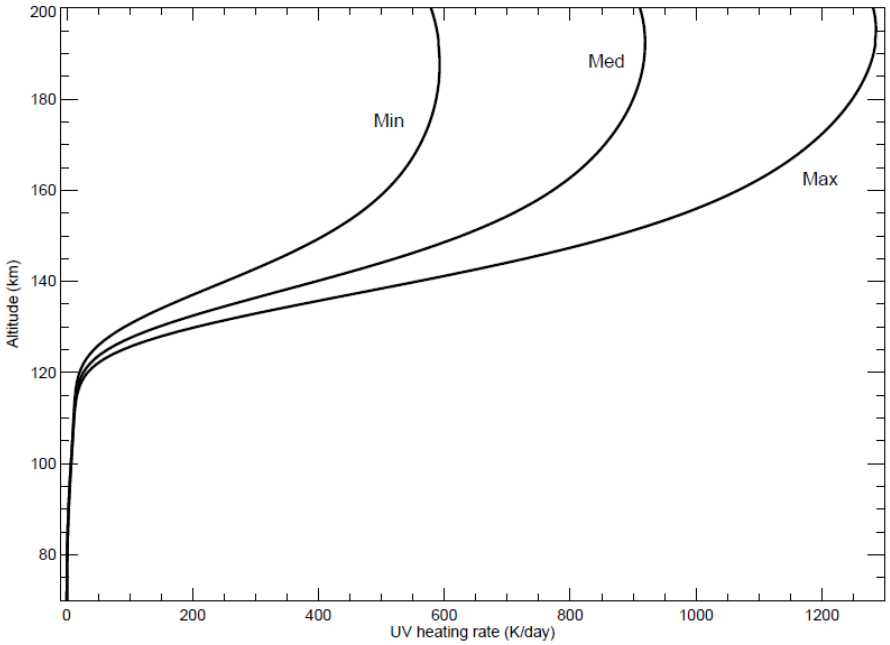


Figure 4.3: Illustration of typical heating rates as a function of altitude for the 3 EUV conditions

In addition to meteorological variables, the MCD provides different datafiles useful to compute the altitude of S/C above the surface and high resolution surface pressure. These files are the following:

- File **mountain.nc**, contains maps (at GCM resolution) of topography;
- File **mola32.nc**, contains a high resolution (32 pixels per degree) map of Martian topography;
- File **mgm1025**, contains the spherical harmonic expansion coefficients necessary to compute the Martian areoid with high accuracy;
- File **VL1.ls**, contains a diurnaly averaged and smoothed record of surface pressure at Viking Lander 1 site.

4.1.2 Data Used by Atmosphere Model

The task of Mars Atmosphere Model in mission simulation environment consists in computing 6 different atmosphere properties (density, pressure, temperature, viscosity, speed of sound, wind) which affect other components of simulator, as discussed in section 3.2. As a matter of fact, the model does not required the whole database to obtain its outputs, but only a part of it.

The objective of this section is to list all variables of MCD used by Mars Atmosphere Model. Basically, they can be divided into 5 categories in terms of variable dimension (scalar, array, matrix):

- **scalar data**: parameters with a set value;
- **1-D data**: arrays storing different classes of statistics, such as atmospheric component masses or longitude coordinates;
- **2-D data**: variables depending on 2-D position (e.g. longitude-latitude);
- **3-D data**: variables depending on 2-D position and time, such as surface pressure;
- **4-D data**: variables depending on 3-D position (longitude-latitude-altitude) and time, such as temperature and density.

The data in the MCD are written using various types of format, principally NetCDF (.nc). Therefore, every file has been converted to .mat format, through Matlab 2017b functions, in order to manage them in Simulink as LUTs.

Table 4.1 summarises all the variables included in model inputs. The symbol column displays the names of variables in Matlab environment.

Variable	Symbol	Unit	Dimension
VL1 longitude	longVL	deg	scalar
VL1 latitude	latVL	deg	scalar
VL1 longitude index	iVL	—	scalar
VL1 latitude index	jVL	—	scalar
Altitude of VL1 (GCM DEM)	AltGCM_VL	m	scalar
Altitude of VL1 (MOLA DEM)	AltMOLA_VL	m	scalar
Longitudes of GCM grid	longitude	deg	1-D
Latitudes of GCM grid	latitude	deg	1-D
Altitudes of pressure layers	altitude	km	1-D
Hybrid pressure level	aps	Pa	1-D
Hybrid sigma level	bps	—	1-D
Times of day	time	martian hours	1-D
Longitudes of MOLA grid	lonmola	deg	1-D
Latitudes of MOLA grid	latmola	deg	1-D
Molecular masses of components	mass	kg mol^{-1}	1-D
Orography (GCM DEM)	oro	m	2-D
Orography (MOLA DEM)	altmola	m	2-D
VL1 surface pressure records	psVL	Ls | Pa	2-D
Spherical harmonic coefficient C	C	—	2-D
Spherical harmonic coefficient S	S	—	2-D
Surface temperature	tsurf	K	3-D
Surface pressure	ps	Pa	3-D
Atmospheric temperature	temp	K	4-D
Atmospheric density	rho	kg m^{-3}	4-D
Zonal (East-West) wind	u	m/s	4-D
Meridional (North-South) wind	v	m/s	4-D
Vertical (down-up) wind	w	m/s	4-D
O volume mixing ratio	vmr_o	mol/mol	4-D
CO_2 volume mixing ratio	vmr_co2	mol/mol	4-D
CO volume mixing ratio	vmr_co	mol/mol	4-D
N_2 volume mixing ratio	vmr_n2	mol/mol	4-D

Table 4.1: Variables of MCD used by Atmosphere Model

4.2 Description of Mathematical Model

The Mars Atmosphere model must be able to compute climatic conditions for a given position and time. To accomplish this task, the knowledge of Mars physical characteristics, orbital elements and topography is essential. As a matter of fact, the model makes use of several mathematical methods derived from different branches of physics, such as gravity models, astrodynamics and fluid mechanics.

First of all, the altitude of S/C above the surface is calculated using MOLA DEM and Martian areoid. Subsequently, Solar Longitude and Local True Solar Time of Mars can be obtained through Mars orbital parameters and Julian Date value. Once position (longitude, latitude, altitude) and time are known, the algorithm is able to extract the required atmospheric variables from Mars Climate Database LUTs, performing interpolation between encompassing grid points, Martian months and Martian hours. Viking Lander 1 pressure records are used to correct atmospheric mass and create a new vertical profile of variables by implementing a hydrostatic equation. The air viscosity and the speed of sound, which are not included in the database, are computed as a function of temperature, volume mixing ratio of atmospheric components and heat ratio.

The following paragraphs provide a description of all numerical methods and equations used by the model.

4.2.1 Altitude of S/C Above Local Surface

In order to acquire the altitude of lander above the local surface, the MCD provides a high resolution map of Martian topography, called MOLA DEM, that represents the terrain surface elevation from Mars areoid. The areoid[2] is defined as the gravitational and rotational equipotential surface of Mars, the analogous to the terrestrial geoid. It is employed as a reference surface for Mars topography, in the absence of a convenient "sea level". The MOLA resolution is 32 pixels per degree, that means a grid dimension of 11520 $\times$ 5760 in longitude $\times$ latitude, a much higher resolution than GCM grid.

Figure 4.4 shows the MOLA DEM height information as a function of longitude and latitude. We can notice how lands of southern hemisphere are overall higher than the ones of northern hemisphere, with the exception of *Hellas Planitia* crater (the large blue spot at the bottom left). The white spots in the centre of the figure represent the highest mountains of Mars.

[2] From Ancient Greek word "*Ares*", which is the personification of planet Mars in Greek mythology.

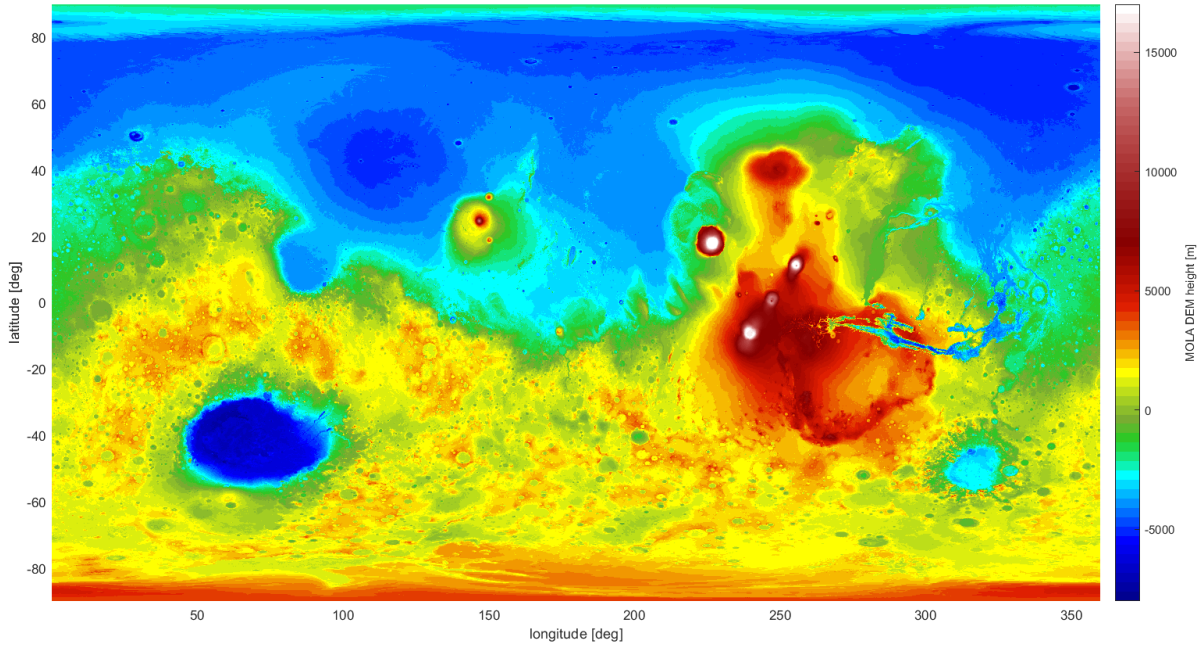


Figure 4.4: Mars topography (MOLA DEM)

On the other hand, the distance between Mars areoid and the center of the planet is not computed using datasets but it is estimated through a spherical harmonics representation. The latter is based on an available gravitational field model called Goddard Mars Model (MGM1025) [5], that provides a spherical harmonic solution of the Mars gravity field to degree and order 80, using X band tracking data of the MGS Mars Orbiter Laser Altimeter.

The gravitational/rotational potential V at a generic point in Mars body-fixed rotating frame can be defined as:

$$V = \frac{GM}{r}\left\{1 + \sum_{l=2}^{N}\sum_{m=0}^{l}\left(\frac{r_M}{r}\right)^l \bar{P}_{lm}(sin\phi)[\bar{C}_{lm}\cos(m\lambda) + \bar{S}_{lm}\sin(m\lambda)]\right\} + \frac{1}{2}\omega^2 r^2\cos^2\phi \tag{4.1}$$

where r is the radial distance from the centre of mass of Mars to the point, ϕ and λ are respectively the aerocentric latitude and longitude of the point, r_M is the mean radius of the reference ellipsoid of Mars, GM is the product of the universal gravitational constant and the mass of Mars, ω is the angular velocity of the planet, $\bar{P}_{lm}$ are the normalized associated Legendre functions of degree l and order m, $\bar{C}_{lm}$ and $\bar{S}_{lm}$ are the normalized spherical harmonic coefficients estimated by the tracking observations to determine the gravitational model, and N is the maximum degree representing the size (or resolution) of the field.

The first term of the equation is the gravitational potential, defined as the potential energy per unit mass in a gravitational field whereas the second term represents the centrifugal potential. The values of constants contained in Eq.(4.1) are listed in table 4.3.

Constant	Definition	Unit	Value
GM	Standard gravitational parameter	$m^3\ s^{-2}$	$42828.37{\cdot}10^9$
r_M	Mean radius of the reference ellipsoid	m	$3396{\cdot}10^3$
ω	Angular velocity of Mars	$rad\ s^{-1}$	$0.70882{\cdot}10^{-4}$

Table 4.2: Mars gravity field constants

The normalized associated Legendre functions $\bar{P}_{lm}$ can be written as:

$$\bar{P}_{lm}(\sin\phi) = \sqrt{\frac{(2l+1)k(l-m)!}{(l+m)!}} P_{lm} \tag{4.2}$$

where P_{lm} is the associated Legendre functions:

$$P_{lm}(sin\phi) = \frac{(1-\sin^2\phi)^{m/2}}{2^l l!} \frac{d^{l+m}(\sin^2\phi - 1)^l}{d(\sin\phi)^{l+m}} \tag{4.3}$$

and k is a parameter depending on the order m:

$$k = \begin{cases} 1 & \text{if } m = 0 \\ 2 & \text{if } m \neq 1 \end{cases}$$

The normalized spherical harmonic coefficients $\bar{C}_{lm}$ and $\bar{S}_{lm}$ are provided by the MCD. The gravitational/rotational potential of the areoid, on which MOLA height information is based, corresponds to the value computed in the intersection between martian prime meridian ($\lambda = 0°$) and martian equator ($\phi = 0°$), where $r = r_M$. Therefore, the only unknown of Eq.(4.1) is the areoid radius, $r = r_{ae}$, that may be computed by a suitable iterative procedure for a given longitude and latitude. Because the position of S/C (x_{FR}, y_{FR}, z_{FR}) in Mars body-fixed rotating frame is known, radius from planet centre, longitude and latitude can be calculated:

$$\begin{cases} r_{S/C} = \sqrt{x_{FR}^2 + y_{FR}^2 + z_{FR}^2} & \text{radius} \\ \lambda = \tan^{-1}\left(\frac{y_{FR}}{x_{FR}}\right) & \text{longitude} \\ \phi = \sin^{-1}\left(\frac{z_{FR}}{r}\right) & \text{latitude} \end{cases}$$

The value of MOLA height h_{MOLA} at a given location (which is not on MOLA grid) can be acquired by horizontal bilinear interpolation of encompassing grid values. In conclusion, the altitude of spacecraft $h_{S/C}$ above the local surface can be easily obtained:

$$h_{S/C} = r_{S/C} - h_{MOLA} - r_{ae} \tag{4.4}$$

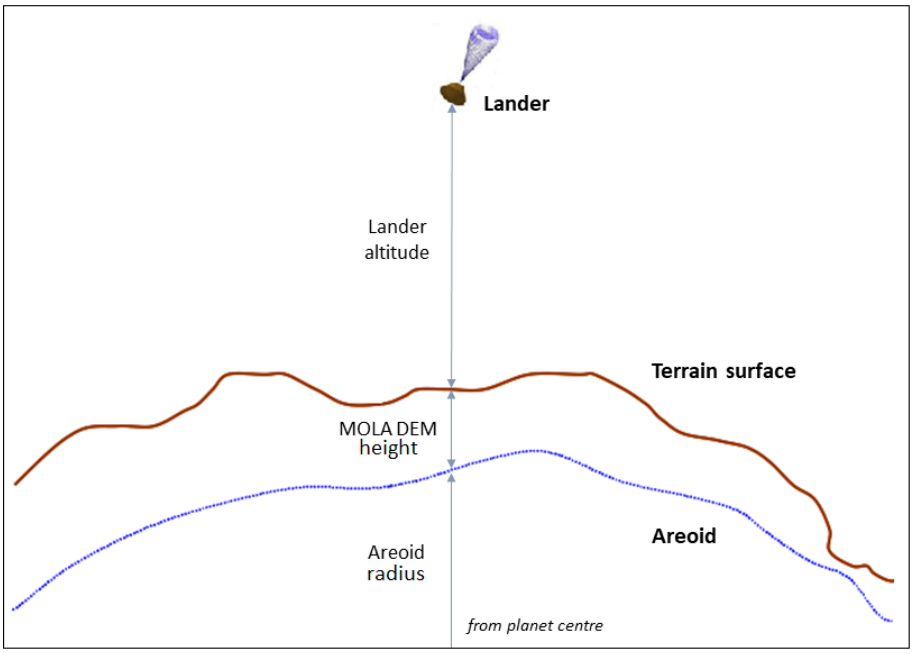


Figure 4.5: Illustration of vertical coordinates

4.2.2 Martian Date and Solar Longitude

This paragraph describes how aerocentric solar longitude of Mars and the corresponding Local True Solar Time at longitude 0 can be computed for a given Julian date [4]. This process can be summarised in 3 main steps:

1. Converting Julian date JD to Martian sol number D_s;
2. Converting sol number D_s date to Ls;
3. Computing Local True Solar Time $LTST$.

The following table displays the values of the parameters used by equations:

Constant	Definition	Unit	Value
N_s	Number of sols in a martian year	sols	668.6
Ls_{peri}	Perihelion solar longitude	deg	250.99
t_{peri}	Perihelion date	sols	485.35
a	Semi-major axis of orbit	AU	1.52368
e	Orbital eccentricity	—	0.09340
ϵ	Obliquity of equator to orbit	deg	25.1919

Table 4.3: Mars orbit parameters

The martian sol date, $D_s \in [0; N_s]$, is the number of sols elapsed since beginning of martian year defined by $Ls = 0$. It can be obtained from a given Julian date:

$$D_s = (JD - JD_{ref})\frac{86400}{88775.245} \quad (mod\ N_s) \tag{4.5}$$

where 86400 and 88775.245 are respectively the number of seconds in an earth day and a martian sol. JD_{ref} is a reference Julian date corresponding to an $Ls = 0$ event[3]. The symbol *mod* indicates a modulo operation.

Once the martian sol date is acquired, the aerocentric solar longitude Ls can be obtained from values of the three anomalies. The mean anomaly M corresponds to the angular distance from the perihelium which a fictitious body would have if it moved in a circular orbit, with constant speed, in the same orbital period as the planet in its elliptical orbit.

It is equal to (in radians):

$$M = 2\pi\frac{D_s - t_{peri}}{N_s} \tag{4.6}$$

To compute eccentric anomaly E, the angular position of planet projected onto the reference circle as shown in figure 4.6, the nonlinear relation between eccentric and mean anomaly can be used:

$$M = E - e\sin E \tag{4.7}$$

which can be solved by the classical Newton iterative procedure.

[3]The reference date used in the model for a $Ls = 0$ event is on on 19-12-1975 at 4:00:00, which corresponds to Julian date 2442765.667.

The true anomaly ν, defined as the angle between the planet and the perihelion in orbital plane, is calculated, in radians, by the following equation:

$$\nu = 2\tan^{-1}\left[\sqrt{\frac{1+e}{1-e}}\tan\left(\frac{E}{2}\right)\right] \tag{4.8}$$

Finally, the solar longitude Ls can be obtained as the sum between true anomaly and solar longitude at perihelion:

$$Ls = \left(\nu\frac{180}{\pi} + Ls_{peri}\right) \quad (mod\ 360) \tag{4.9}$$

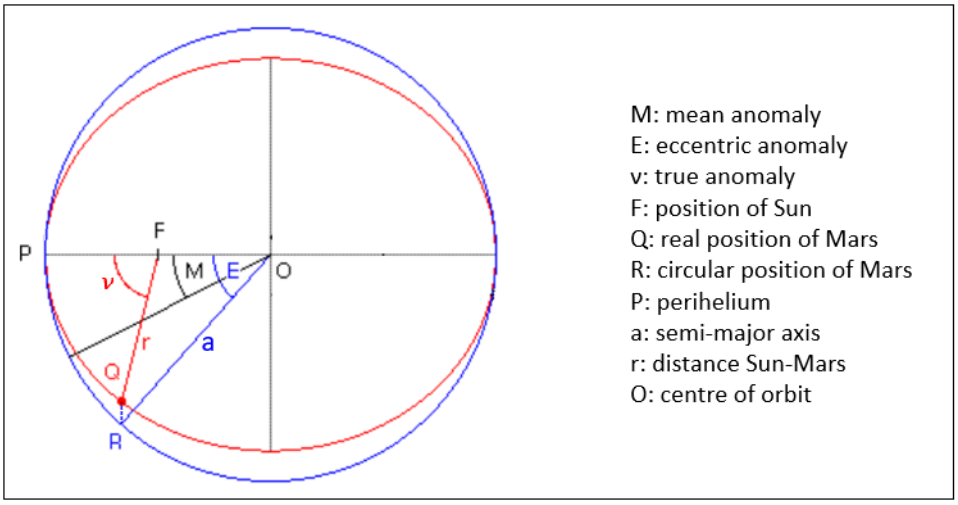


Figure 4.6: Illustration of the three anomalies in orbital plane

Once the solar longitude is determined, it is possible to compute the Local Mean Time at longitude 0 LMT_0, in martian hours, for a given JD:

$$LMT_0 = \left(LMT_{ref} + 24(JD - JD_{ref})\frac{86400}{88775.245}\right) \quad (mod\ 24) \tag{4.10}$$

where JD_{ref} is a reference Julian date, and LMT_{ref} the Local Mean Time at longitude 0, in martian hours, at that date[4]. Due to Mars' orbital eccentricity and obliquity, the LMT_0 and the Local True Solar Time at longitude 0, $LTST_0$, are not the same.

[4]The reference date used in the model is: 01-01-1975 at 00:00:00 was such that $JD_{ref} = 2442778.5$ and $LMT_{ref} = 16.1725$.

The Equation Of Time EOT provides the difference between Mean Time and True Solar Time as a function of orbital eccentricity, obliquity and Ls (in martian hours):

$$EOF = \frac{24}{2\pi}\left[2e\sin(Ls - Ls_{peri}) - \tan^2\left(\frac{\epsilon}{2}\right)\sin(2Ls)\right] \tag{4.11}$$

where Ls_{peri} is the perihelion date, e the orbital eccentricity and ϵ the obliquity of equator to orbit.

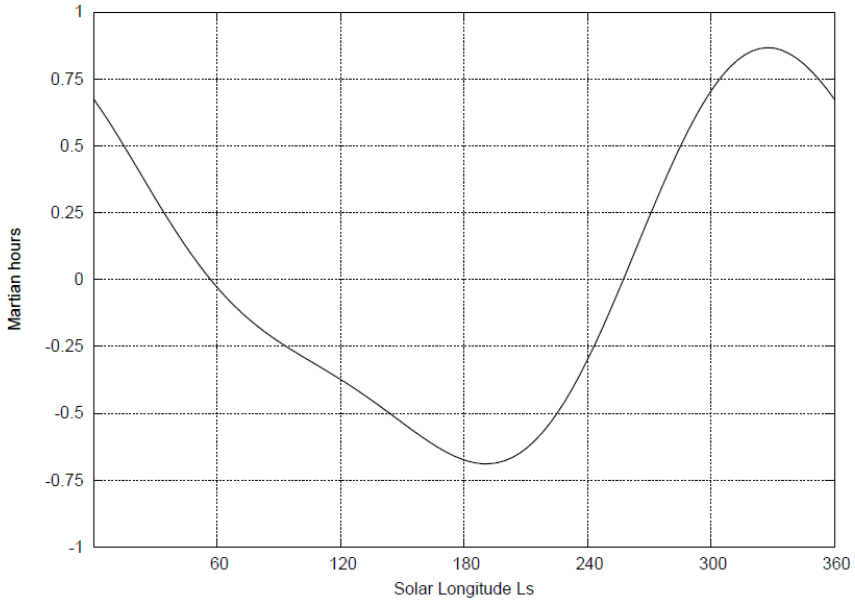


Figure 4.7: Equation Of Time (EOT) for Mars as a function of Solar Longitude

Finally, $LTST_0$, i.e. the reference time of Mars Climate Database, can be calculated in martian hours:

$$LTST_0 = LMT_0 - EOT \tag{4.12}$$

At longitude lon (in degrees east), Local True Solar Time $LTST$, in martian hours, is then:

$$LTST = LTST_0 + \frac{lon}{15} \quad (mod\ 24) \tag{4.13}$$

4.2.3 Computation of Mars Atmosphere Properties

This paragraph provides a description of how Atmosphere model manages climatic variables in order to obtain high resolution outputs, for a given position and time. Firstly, to compute the values of variables which are not on GCM grid, a virtual column of data has to be built along the GCM vertical levels. This profile is obtained by 3 different interpolations:

1. bilinear interpolation between encompassing grid values in terms of longitude and latitude;

2. linear interpolation between encompassing martian hours;

3. linear interpolation between encompassing seasons, defined by solar longitude.

This is an example of linear interpolation between seasons:

$$var = var_L + \frac{(Ls - Ls_L)}{(Ls_H - Ls_L)}(var_H - var_L)$$

where var is the required variable (e.g. temperature, density, wind), Ls is the solar longitude, and subscripts "L" and "H" indicate the values of variable stored respectively in seasons "low" and "high". However, the Mars Climate Database has been compiled from the output of a general circulation model in which the topography is very smoothed due to its low resolution. Therefore, a combination of high resolution MOLA topography and accurate pressure records of Viking Lander 1, used as a reference to correct the atmospheric mass, with MCD surface pressure is utilized to compute surface pressure as accurately as possible. The latter is then employed to reconstruct vertical pressure levels and thus high resolution interpolated values of atmospheric variables.

The new value of surface pressure is given by:

$$Ps = Ps_{GCM}\frac{< P_{VL1_{MOLA}} >}{< P_{VL1_{GCM}} >}e^{-(z-z_{GCM})/H} \tag{4.14}$$

where Ps_{GCM} is the pressure predicted by the GCM at a given location and time, $< P_{VL1_{MOLA}} >$ is the diurnal average of VL1 surface pressure records corrected by MOLA grid, $< P_{VL1_{GCM}} >$ is the diurnal average of GCM surface pressure, z is the altitude of local surface retrieved from MOLA dataset, and z_{GCM} is the altitude at the location interpolated from the coarse GCM topography grid. H is the scale height, in meters, used in the hydrostatic equation to vertically interpolate the pressure: $H = \frac{RT}{g}$, with $R = 191\ m^2s^{-2}K^{-1}$ the gas constant,

$g = 3.72\ ms^{-2}$ the acceleration of gravity, and T the atmospheric temperature extracted from the GCM at about 1 km above the surface.

Consequently, a new vertical profile of "high resolution" pressure levels $P_{HR}(l)$ can be built using an empirical function based on available atmospheric data and designed to ensure a steady transition from the near-surface conditions to the free atmosphere (high altitudes) environment:

$$P_{HR}(l) = P_{GCM}(l)\left[f(l) + 0.5(1 - f(l))(1 + \tanh(6(-10\log\left(\frac{P_{GCM}(l)}{Ps_{GCM}}\right) - z)/z))\right] \tag{4.15}$$

where $P_{GCM}(l)$ is the original pressure level in database. z is a parameter depending on the altitude difference Δz, derived from pressure, between the high and low resolution grid (km):

$$\Delta z = -10\log\left(\frac{Ps_{HR}}{Ps_{GCM}}\right) \tag{4.16}$$

and it is set to:

$$z = \begin{cases} \Delta z + 3 & \text{if } \Delta z > 0 \text{ (local mountain)} \\ 3 & \text{if } \Delta z \leq 0 \text{ (local depression)} \end{cases}$$

$f(l)$ is another parameter planned to ensure that model levels are compressed closer to surface, to mimic the observed behaviour in high resolution GCM simulations, as previously shown in figure 4.2. In particular, $f(l)$ is computed as:

$$f(l) = \frac{Ps_{HR}}{Ps_{GCM}}\left(\frac{P_{GCM}(l)}{Ps_{GCM}}\right)^{x} \tag{4.17}$$

where x is the parameter which controls the compression or extension of the layers near the surface: compression if $x < 0$, extension if $x > 0$, no effect if $x = 0$. The value of x is derived from that of Δz:

$$x = \begin{cases} 0 & \text{if } -1 < \Delta z < 1 \\ -0.12(|\Delta z| - 1) & \text{if } \Delta z > 1 \\ +0.12(|\Delta z| - 1) & \text{if } \Delta z < 1 \\ 0.8 & \text{if formulas above yield } x > 0.8 \\ -0.8 & \text{if formulas above yield } x < -0.8 \end{cases}$$

Finally, the heights $h(l)$ of new pressure levels can be obtained through an iterative procedure, whose initial conditions correspond to those of surface level:

$$h(l) = h(l-1) - \frac{R(l)T_m}{g} \log\left(\frac{\sigma(l)}{\sigma(l-1)}\right) \tag{4.18}$$

The various terms represent:

- Gas constant ($m^2 s^{-2} K^{-1}$):

$$R(l) = \frac{p_{GCM}(l)}{\rho_{GCM}(l) T_{GCM}(l)}$$

- Average temperature (K):

$$T_m = (T_{GCM}(l) - T_{GCM}(l-1)) \cdot \log\left(\frac{T_{GCM}(l)}{T_{GCM}(l-1)}\right)$$

- Acceleration of gravity (ms^{-2}):

$$g = g_0 \frac{a_0}{a_0 + h_{MOLA} + h(l-1)}$$

  where $g_0 = 3.7257964\ ms^{-2}$ is the gravity at the areoid $a_0 = 3396 \cdot 10^3\ m$

- Pressure ratio:

$$\sigma(l) = \frac{P_{HR}(l)}{Ps_{HR}}$$

With the new vertical profile, it is now possible to calculate the "high resolution" properties using vertical interpolation between pressure layers. Two kinds of interpolation are adopt in line with variable's behaviour:

1. linear interpolation for variables which have a vertical steady trend, i.e. temperature and wind;

2. logarithmic interpolation for variables which take into account the change in pressure, i.e. pressure and density.

The values of viscosity and speed of sound are not included in MCD and they must be computed as a function of other parameters. Initially, atmosphere's molar mass M(kg mol^{-1}), isobaric heat capacity Cp (J kg^{-1} K^{-1}) and thermal conductivity coefficient A (W m^{-1} K^{-1}) are evaluated:

$$M = \frac{\sum_{i=1}^{5} M_i \cdot vmr_i}{\sum_{i=1}^{5} vmr_i} \tag{4.19}$$

$$Cp = \frac{1}{M} \cdot \frac{\sum_{i=1}^{5} Cp_i \cdot vmr_i}{\sum_{i=1}^{5} vmr_i} \tag{4.20}$$

$$A = \frac{\sum_{i=1}^{5} A_i \cdot vmr_i}{\sum_{i=1}^{5} vmr_i} \tag{4.21}$$

where vmr_i is the volume mixing ratio of the i-th component of atmosphere provided by MCD. The following table displays the properties of each constituent:

Name	M $(kgmol^{-1})$	Cp $(Jmol^{-1}K^{-1})$	A $(Wm^{-1}K^{-1})$
CO_2	$44.01{\cdot}10^{-3}$	37	$3.07{\cdot}10^{-4}$
CO	$28.01{\cdot}10^{-3}$	29	$4.87{\cdot}10^{-4}$
N_2	$28.01{\cdot}10^{-3}$	29	$5.60{\cdot}10^{-4}$
O	$16.00{\cdot}10^{-3}$	21	$7.60{\cdot}10^{-4}$
Ar	$39.95{\cdot}10^{-3}$	20	$3.40{\cdot}10^{-4}$

Table 4.4: Properties of Mars atmosphere components

The heat ratio γ is equal to:

$$\gamma = \frac{1}{1 - R/Cp} \tag{4.22}$$

Finally, the viscosity μ $(Pa{\cdot}s)$ and the speed of sound c (ms^{-1}) can be obtained:

$$\mu = \frac{AT^{0.69}}{0.25(4Cp + 5R)} \tag{4.23}$$

$$c = \sqrt{\gamma \frac{p}{\rho}} \tag{4.24}$$

4.3 Modeling and Simulation in Simulink Environment

This paragraph provides a description of the development of Mars Atmosphere Model in Matlab/Simulink environment. The model takes as inputs 2 dynamic states and 4 parameters, and is able to compute 8 different outputs: density, pressure, temperature, speed of sound, viscosity, wind in X, Y and Z direction (in planet fixed rotating frame).

Tables 4.5, 4.6 and 4.7 summarize the input/output interface of the model.

Input	Name	Type	Unit
Time from simulation start	simulation_time	Scalar	s
Position in Planet Fixed frame (FR)	Position_vec_FR	Vector	m

Table 4.5: Inputs of Mars atmosphere model

Parameter	Name	Type	Unit
Julian Date at the beginning of simulation	SimStartJD	Scalar	JD
MOLA orography	MOLA LUTs	Struct	m
Climatic variables of season "low"	SeasonLow LUTs	Struct	—
Climatic variables of season "high"	SeasonHigh LUTs	Struct	—

Table 4.6: Parameters of Mars atmosphere model

Output	Name	Type	Unit
Atmospheric density	Density	Scalar	kgm^{-3}
Atmospheric pressure	Pressure	Scalar	Pa
Atmospheric temperature	Temperature	Scalar	K
Sonic velocity "high"	SpeedOfSound	Scalar	ms^{-1}
Air dynamic viscosity	Viscosity	Scalar	$Pa \cdot s$
Wind in X direction (FR)	WindX	Scalar	ms^{-1}
Wind in Y direction (FR)	WindY	Scalar	ms^{-1}
Wind in Z direction (FR)	WindZ	Scalar	ms^{-1}

Table 4.7: Outputs of Mars atmosphere model

Downstream of Atmosphere model, named *LIB_ATMcomplete* in simulation environment, there is a block called *Atmospheric Velocity*, that converts wind values from Mars Fixed Rotating frame (FR) to Mars Inertial frame (FI) and computes the velocity of lander relative to the atmosphere, and a division block able to evaluate Mach number.

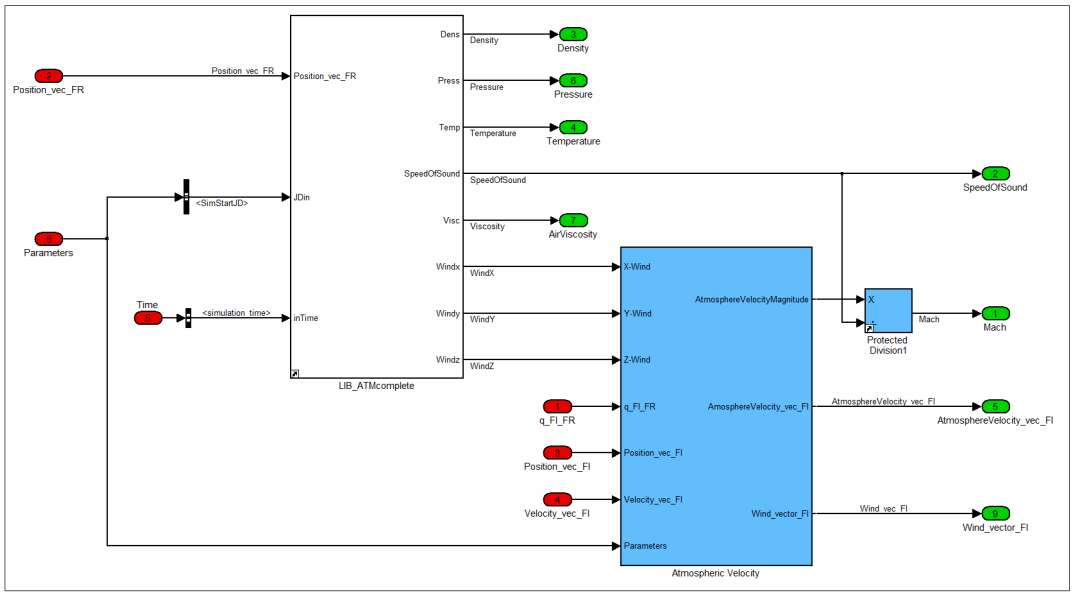


Figure 4.8: Atmosphere model (*LIB_ATMcomplete*) in simulation environment

The Mars Atmosphere Model consists of two submodels, as shown in figure 4.9:

1. **Altitude Model**, which takes as inputs the orography LUTs based on MOLA DEM and the position of spacecraft in Planet Rotation Frame, and computes local orography and height above local surface;

2. **Atmosphere Model**, which takes as inputs the position of the spacecraft in Planet Rotation Frame, Mars Climate Database LUTs, height above local surface, local orography, Julian Date and simulation time, and computes density, pressure, temperature, viscosity, speed of sound and wind.

The first model has been developed using S-Function Builder block, able to generate the C source code and the executable MEX file. The C standard libraries used by the S-function are: *simstruc.h*, *stdlib.h* and *math.h*. A file named *constants.h*, containing spherical harmonic expansion coefficients of the Mars gravity field, has been included in the project. The code makes also use of 2 different external functions:

1. *extern double inter(double orogr[ ][ ], double Long, double Lat, double longitude[ ], double latitude[ ], int i, int j)*: performs linear interpolation between MOLA DEM heights for a given longitude and latitude;

2. *extern double *lgndr(int LMAX, int M, double X)*: computes the associated Legendre function of X for a given polynomial's degree L and order M.

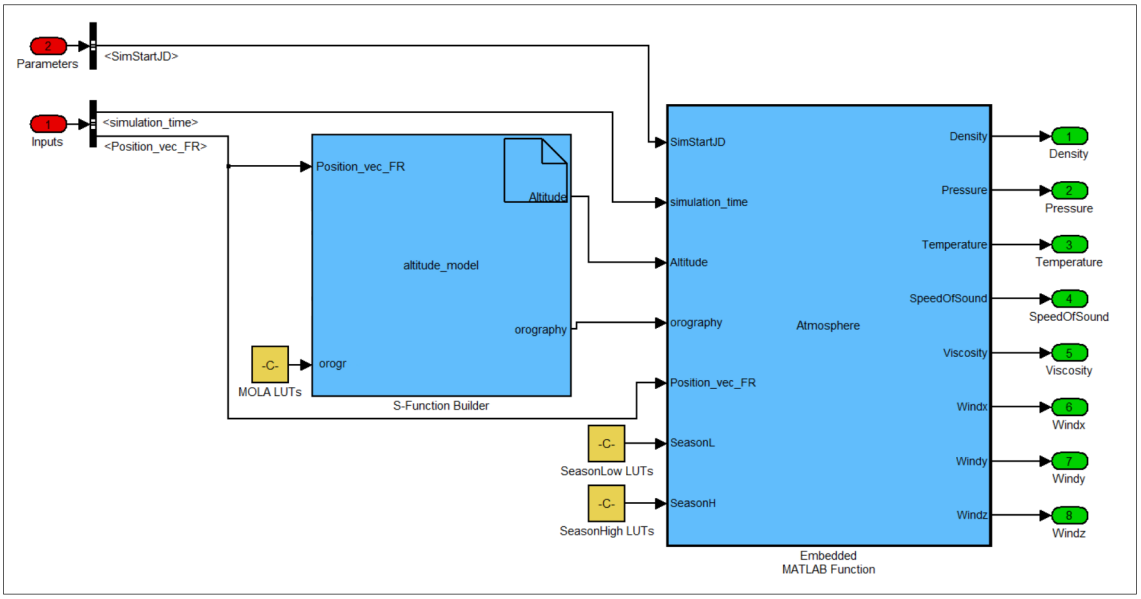


Figure 4.9: Atmosphere submodels in simulation environment

The second model has been developed using Embedded Matlab Function block, followed by an automatic generation of C source code through Real-Time Workshop tool. It includes three different interpolation functions:

1. *inter2D(var,Long,Lat,long,lat,i,j)*: performs linear interpolation between values of atmospheric variable *var* for a given grid position;

2. *inter3D(var,Long,Lat,Time,long,lat,time,i,j,t,logflag)*: performs linear ($logflag = 0$) or logarithmic ($logflag = 1$) interpolation between values of atmospheric variable *var* for a given grid position and time;

3. *inter4D(var,Long,Lat,Alt,Time,long,lat,altit,time,i,j,k,t,logflag)*: performs linear ($logflag = 0$) or logarithmic ($logflag = 1$) interpolation between values of atmospheric variable *var* for a given grid position, altitude and time.

Because the model requires to run in real-time, the use of a high speed performance code, like C language, has been necessary. The model execution time

has been examined through Matlab profiler tool, which generates at the end of simulation a HTML file containing all the execution time measurements of each simulator block, as shown in figure 4.10.

Function List

Name	Time		Calls	Time/call	Self time	
sim	2441.98437500	100.0%	1	2441.9843750000000	0.00000000	0.0%
ModelExecute	2258.25000000	92.5%	1	2258.2500000000000	40.35937500	1.7%
MajorUpdate	1614.92187500	66.1%	3415	0.4728907393851	0.01562500	0.0%
EAGLE_Simulator (Update)	1614.90625000	66.1%	3415	0.4728861639824	0.81250000	0.0%
EAGLE_Simulator/BodyTemplate/GNC/FEIC/FEIC Functional Model/FEIC/FEIC simulation (Update)	1611.92187500	66.0%	194	8.308875643299	1611.92187500	66.0%
EAGLE_Simulator/BodyTemplate/GNC/FEIC/FEIC Functional Model/FEIC (Update)	1611.92187500	66.0%	3415	0.4720122620791	0.00000000	0.0%
MajorOutputs	370.01562500	15.2%	3415	0.1083501098097	0.01562500	0.0%
EAGLE_Simulator (Output)	370.00000000	15.2%	3415	0.1083455344070	116.48437500	4.8%
Integrate	232.95312500	9.5%	3414	0.0682346587581	0.15625000	0.0%
MinorOutputs	231.03125000	9.5%	17070	0.0135343438781	0.07812500	0.0%
EAGLE_Simulator (MinorOutput)	230.95312500	9.5%	17070	0.0135297671353	85.39062500	3.5%
ModelInitialize	183.62500000	7.5%	1	183.6250000000000	183.59375000	7.5%
EAGLE_Simulator/BodyTemplate/GNC/FEIC/FEIC Feature Point Viewer/FEIC viewer (Output)	64.03125000	2.6%	3415	0.0187500000000	0.09375000	0.0%
EAGLE_Simulator/BodyTemplate/GNC/FEIC/FEIC Feature Point Viewer/FEIC viewer/S-Function (Output)	63.46875000	2.6%	388	0.1635792525773	63.46875000	2.6%
EAGLE_Simulator/BodyTemplate/Sensors/Inertial Measurement Unit/Inertial Measurement Unit 1/IMUmodel (MinorOutput)	21.46875000	0.9%	17070	0.0012576889279	0.43750000	0.0%
EAGLE_Simulator/BodyTemplate/Sensors/Camera/PANGU Camera/PANGU CAMERA INTERFACE/Camera Interface (Output)	20.28125000	0.8%	3415	0.0059388726208	0.01562500	0.0%
EAGLE_Simulator/BodyTemplate/Sensors/Camera/PANGU Camera/PANGU CAMERA INTERFACE/Camera Interface/ProcessAndPlotData (Output)	18.76562500	0.8%	193	0.0972312176166	0.01562500	0.0%
EAGLE_Simulator/BodyTemplate/Sensors/Camera/PANGU Camera/PANGU CAMERA INTERFACE/Camera Interface/ProcessAndPlotData/Embedded MATLAB Function/ SFunction (Output)	18.75000000	0.8%	386	0.0485751295337	18.75000000	0.8%
EAGLE_Simulator/BodyTemplate/Sensors/Camera/PANGU Camera/PANGU CAMERA INTERFACE/Camera Interface/ProcessAndPlotData/Embedded MATLAB Function (Output)	18.75000000	0.8%	193	0.0971502590674	0.00000000	0.0%
EAGLE_Simulator/EnvironmentTemplate/AtmosphereModel/Atmosphere/Atmosphere_TASI/S-Function Builder (Output)	18.45312500	0.8%	40970	0.0004504057847	18.45312500	0.8%
EAGLE_Simulator/BodyTemplate/GNC/HDA/HDA/HDA_HAZARDMAPS/HDA_HAZARDMAPS_sfcn (Output)	16.31250000	0.7%	3415	0.0047767203514	16.31250000	0.7%
EAGLE_Simulator/EnvironmentTemplate/AtmosphereModel/Atmosphere/Atmosphere_TASI (MinorOutput)	15.71875000	0.6%	17070	0.0009208406561	0.18750000	0.0%

Figure 4.10: Illustration of block profiling layout for Atmosphere model simulation

Once real-time execution of the model has been verified, the validation activity has begun.

4.4 Validation Activity

The main goal of validation activity is the production of an accurate and credible Atmosphere model. To accomplish this task, an available model of Mars atmosphere has been taken as reference in order to compare its outputs with those of the new Atmosphere model. The flight simulation chosen to test the two models has consisted in a classical entry, descent and landing maneuver, whose duration has been 336.8 seconds. All the data have been acquired using '*To file*' Simulink blocks so as to save the outputs of both models into .mat files. The values of every variables and the relative absolute errors have been then plotted as a function of simulation time. For convenience, the outputs of the reference model are labelled as "reference simulation" and the outputs of the new atmosphere model as "TASI model".

As shown in figures below, output values of the new atmosphere model overlap data of reference simulation and absolute errors of each variables remain under 1% threshold. According to these results, the validation test has been considered satisfactory.

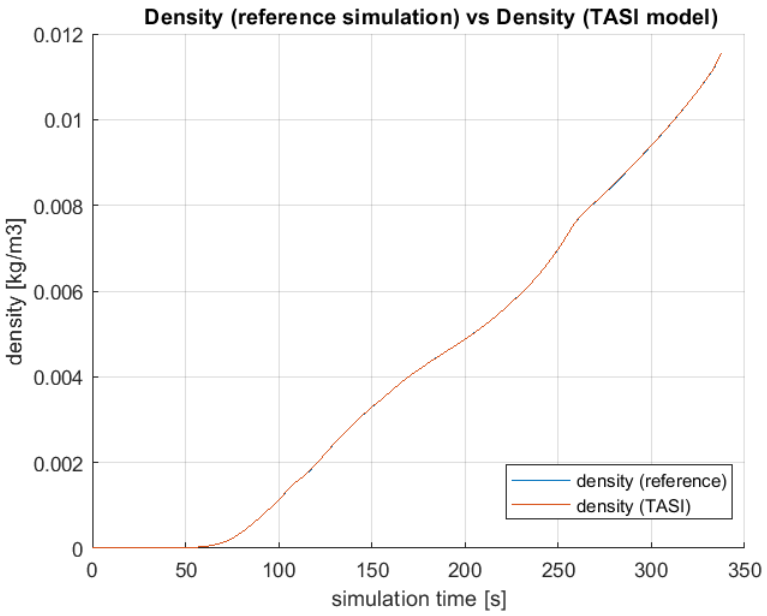


Figure 4.11: Comparison of density values

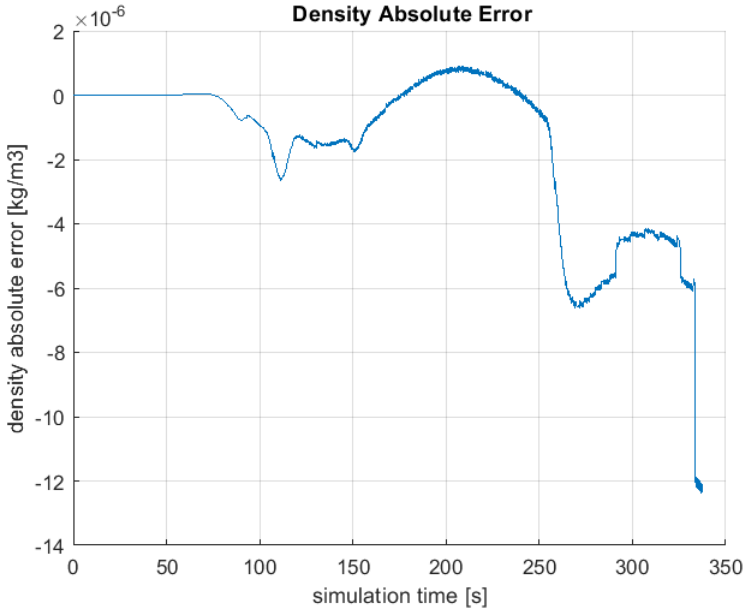


Figure 4.12: Absolute error of density

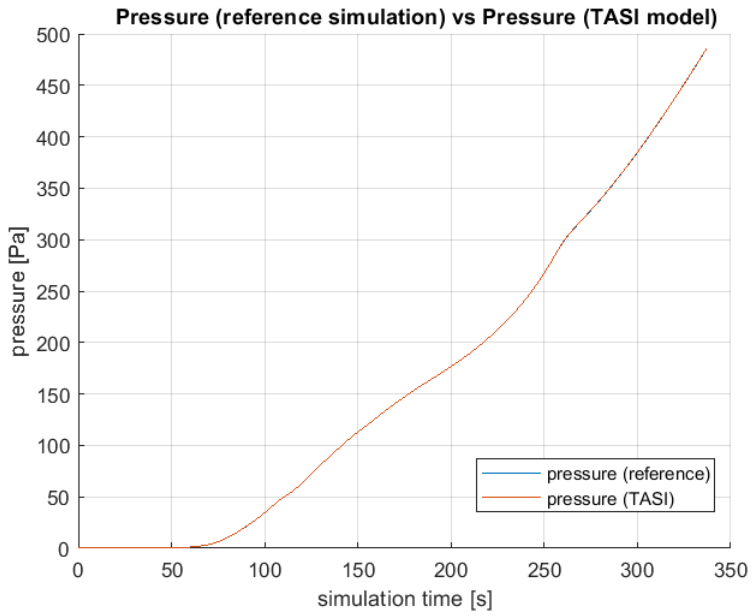


Figure 4.13: Comparison of pressure values

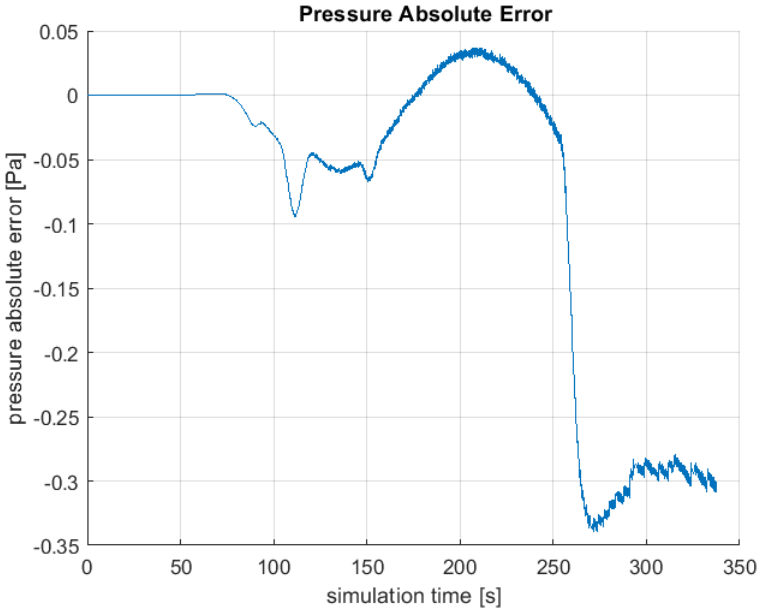


Figure 4.14: Absolute error of pressure

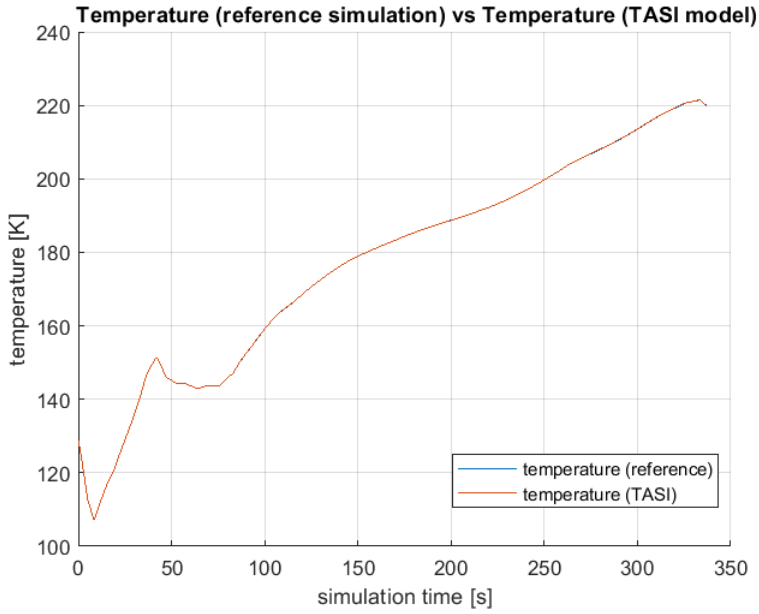


Figure 4.15: Comparison of temperature values

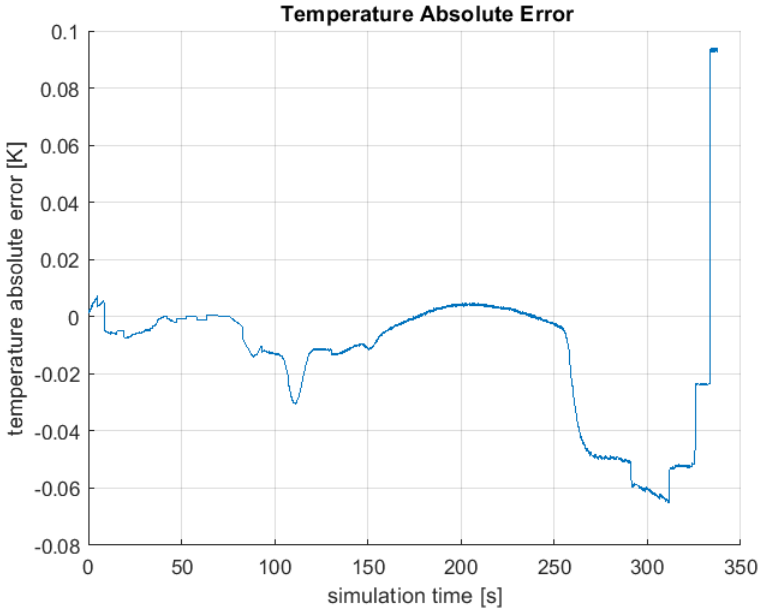


Figure 4.16: Absolute error of temperature

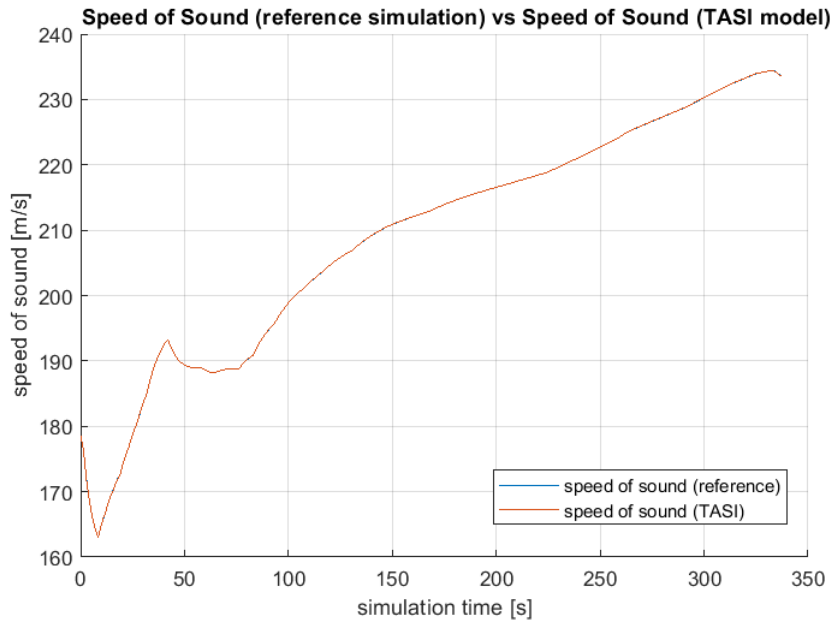


Figure 4.17: Comparison of speed of sound values

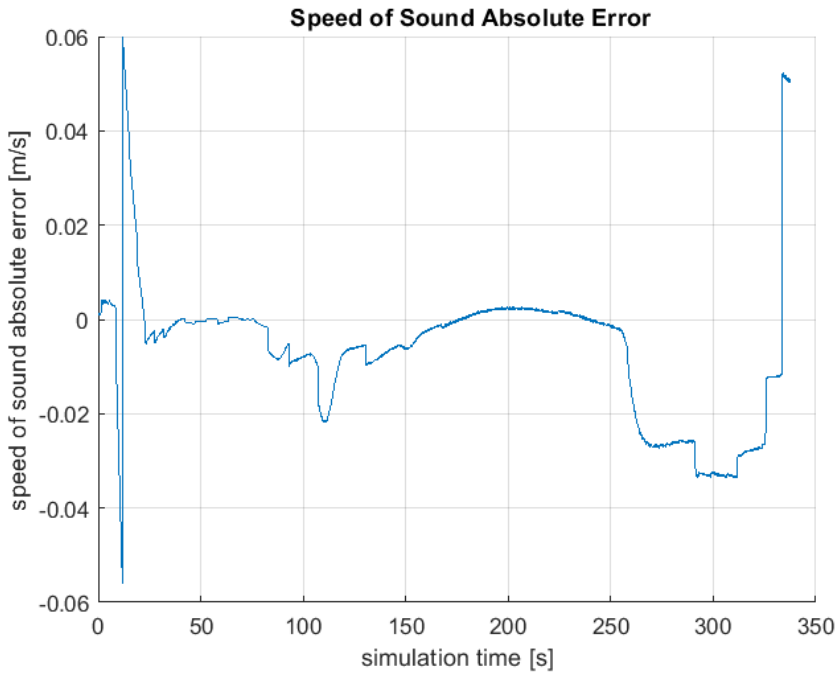


Figure 4.18: Absolute error of speed of sound

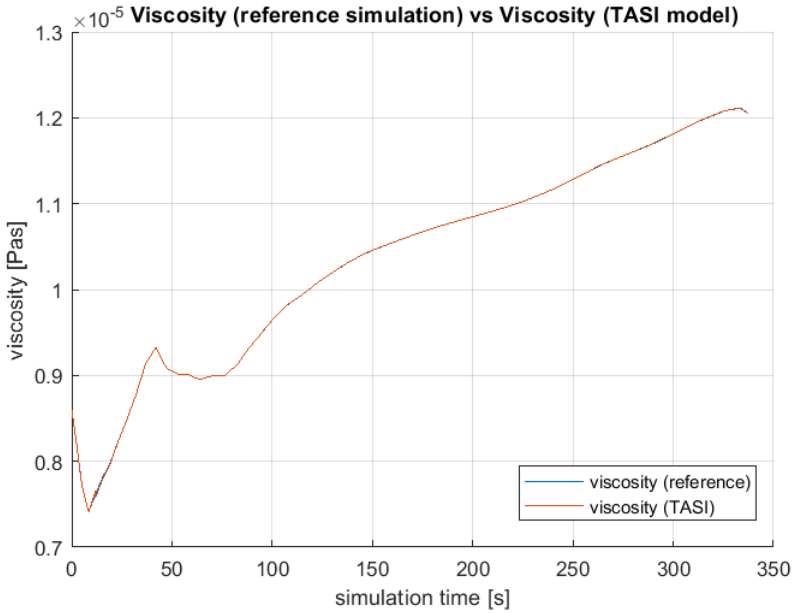


Figure 4.19: Comparison of viscosity values

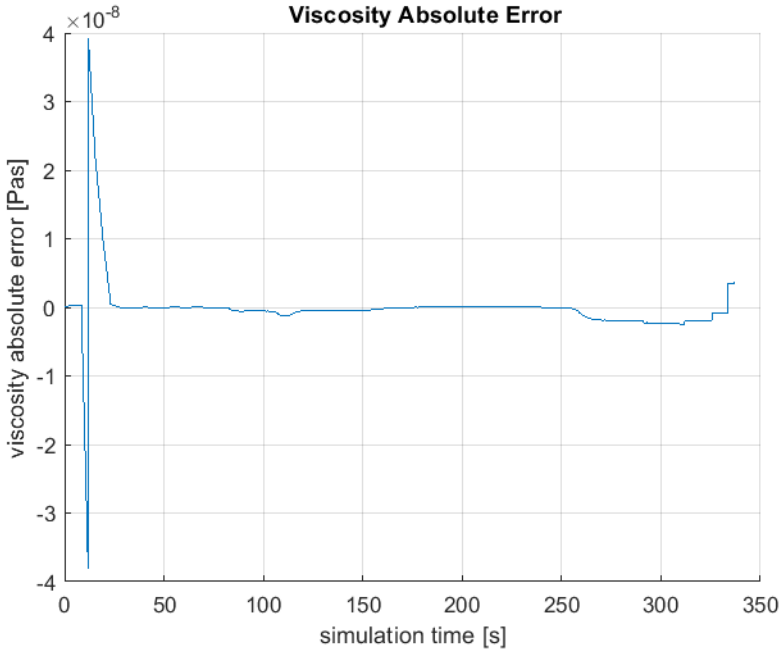


Figure 4.20: Absolute error of viscosity

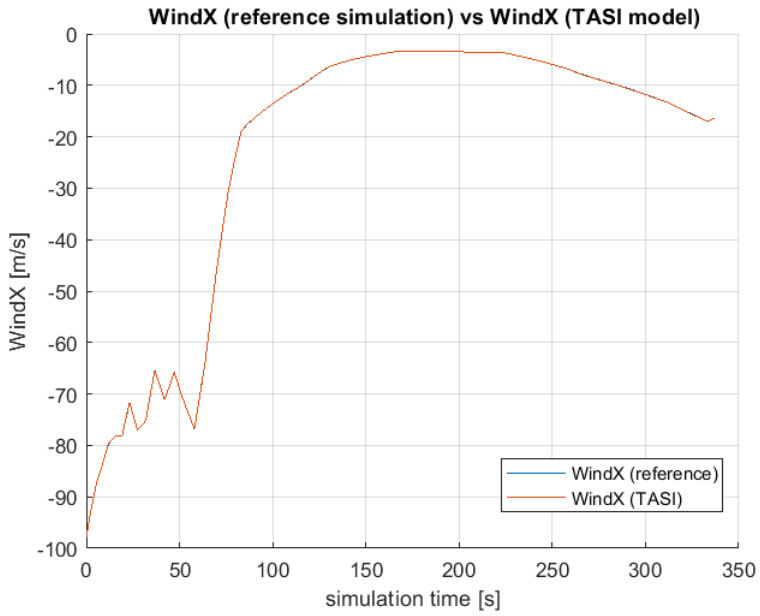


Figure 4.21: Comparison of wind(X) values

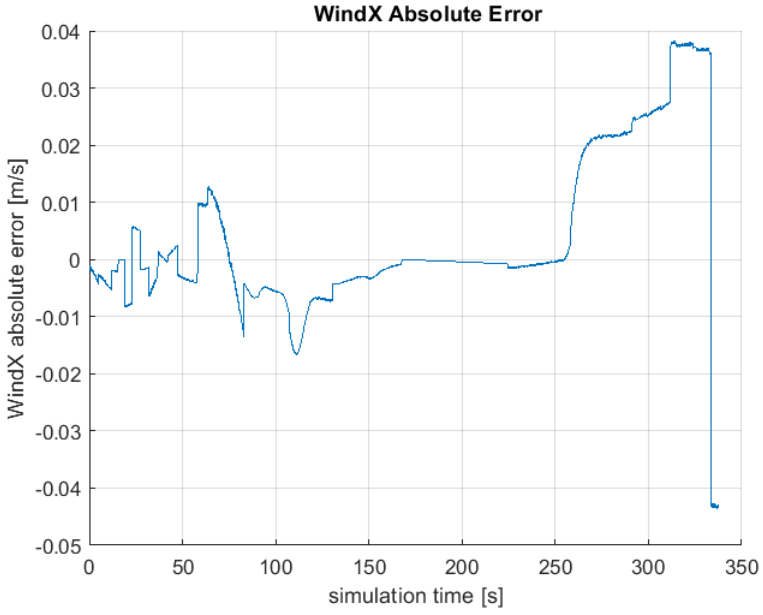


Figure 4.22: Absolute error of wind(X)

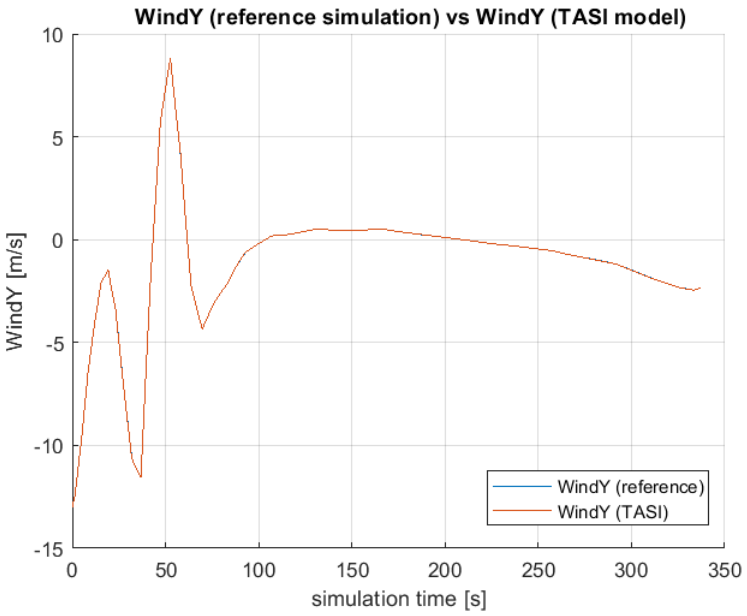


Figure 4.23: Comparison of wind(Y) values

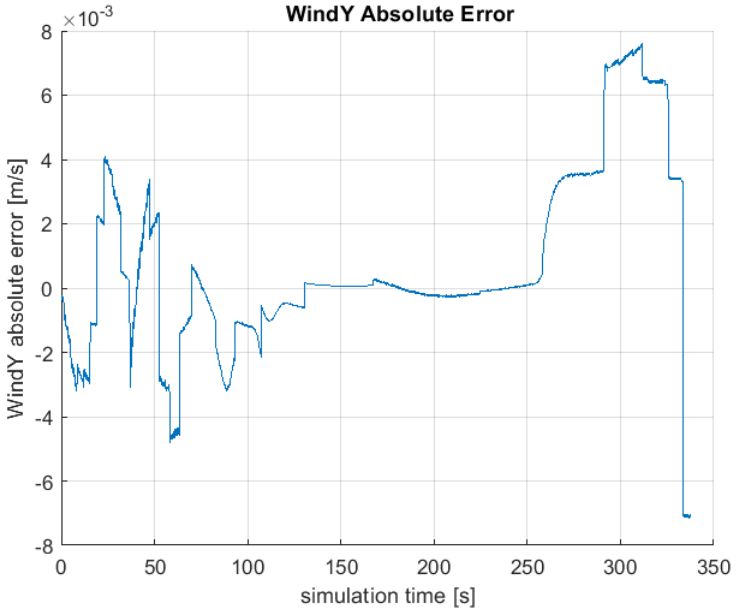


Figure 4.24: Absolute error of wind(Y)

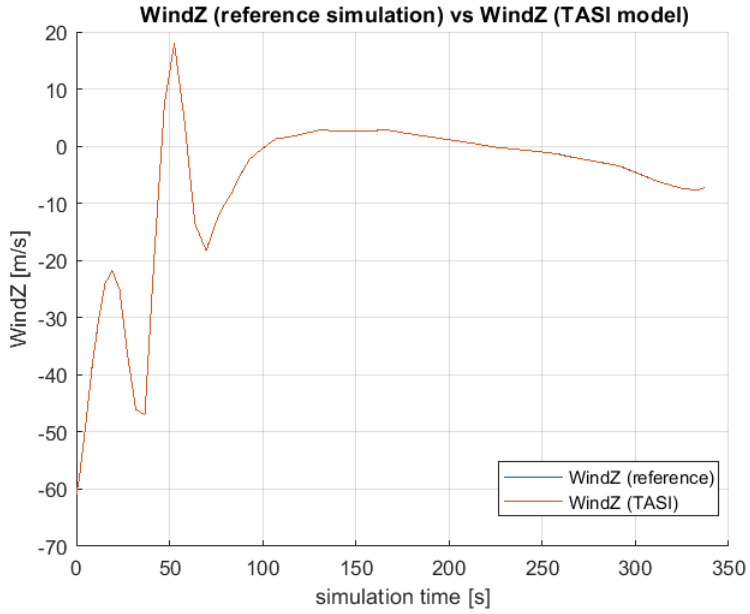


Figure 4.25: Comparison of wind(Z) values

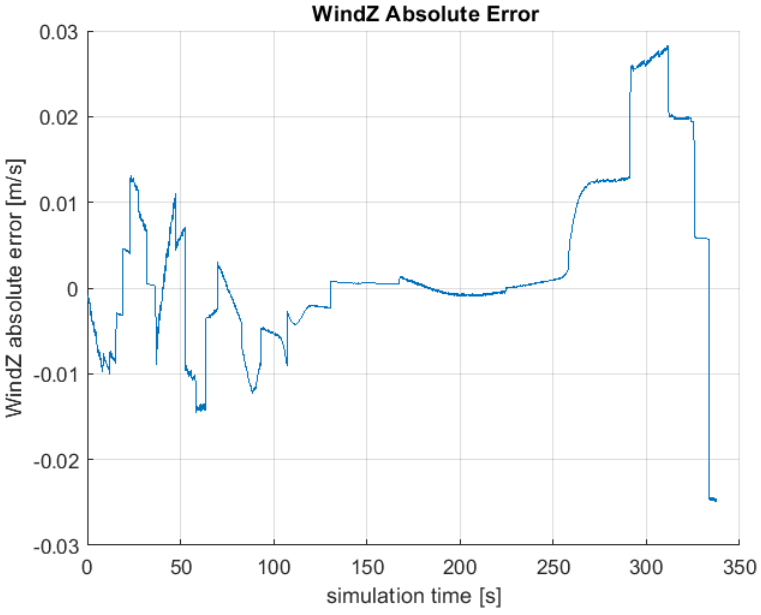


Figure 4.26: Absolute error of wind(Z)

Chapter 5

Altimeter Model

This chapter describes the development process of laser-based Altimeter sensor model. It consists of three sections, representing the main phases of the project: the first section describes the mesh interpolation method used to compute the altitude range, the second section illustrates the model design in Simulink environment and the last section discusses its validation activity.

5.1 Description of Mathematical Model

The Altimeter model must be able to determine the altitude of S/C above the Mars surface for a given position and attitude. While the earlier Altitude model, included into Atmosphere one, computes simply the lander local vertical without considering its orientation, the Altimeter model takes into account the attitude, i.e. the quaternion between Body Axis System (FB) and Planet Fixed Frame (FR), to measure the height value. Therefore, the range measurement evaluated by Altimeter model does not represent the "real" altitude but corresponds to the distance between altimeter sensor and beam-ground intersection.

In order to simulate accurately the behaviour of altimeter sensor, a higher resolution DEM of Martian surface, called PANGU DEM (developed by University of Dundee [3]), is used: its elevation data is expressed in a Local Terrain Frame (LTF), whose position and orientation in FR are available. Because the intersection algorithm requires a triangular mesh, each DEM square have been split in 2 triangles, as shown in figure 5.1. To improve the performance of the model in terms of execution time, only a 10 km^2 portion of planet surface, centred on the nominal landing side, is employed.

The triangular mesh intersection method is based on Ray/Triangle intersection algorithm developed by Tomas Möller and Ben Trumbore [9]. To compute the altitude range, the algorithm must identify in which triangle the beam-ground

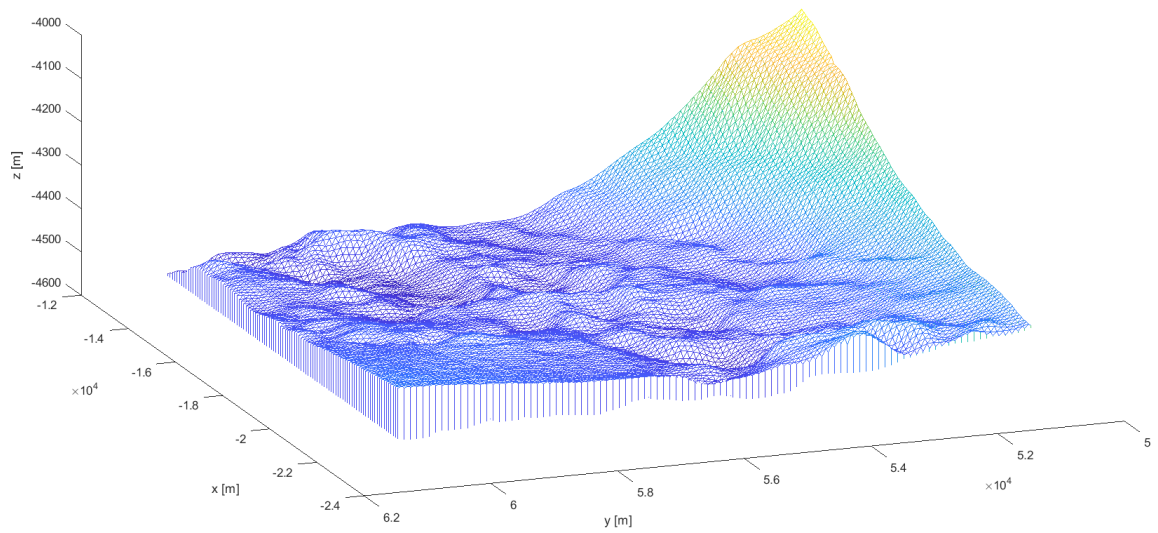


Figure 5.1: PANGU DEM in Local Terrain Frame

intersection lies.

Firstly, the altimeter states are converted from Planet Fixed Frame (FR) to Local Terrain Frame (LTF):

$$\vec{p}_{ALT/LTF_{FR}} = \vec{p}_{ALT_{FR}} - \vec{p}_{LTF_{FR}} \tag{5.1}$$

$$\vec{p}_{ALT_{LTF}} = A_{FR/LTF} \cdot \vec{p}_{ALT/LTF_{FR}} \tag{5.2}$$

$$\vec{d}_{ALT_{LTF}} = A_{FR/LTF} \cdot \vec{d}_{ALT_{FR}} \tag{5.3}$$

where $\vec{p}_{ALT_{FR}}$ is the position of altimeter in FR, $\vec{p}_{LTF_{FR}}$ the position of LTF origin in FR, $\vec{p}_{ALT/LTF_{FR}}$ the relative position between $\vec{p}_{ALT_{FR}}$ and $\vec{p}_{LTF_{FR}}$ in FR, $\vec{p}_{ALT_{LTF}}$ the position of altimeter in LTF, $\vec{d}_{ALT_{FR}}$ the direction of altimeter boresight in FR and $\vec{d}_{ALT_{LTF}}$ the direction of altimeter boresight in LTF. The term $A_{FR/LTF}$ represents the direction cosine matrix, function of the quaternion between FR and LTF. Once the position of altimeter and the direction of boresight in LTF are obtained, the triangle hit by laser ray is then identified through a inside-out test. A point P(u,v) on a triangle can be given by:

$$P(u,v) = (1-u-v)V_1 + uV_2 + vV_3 \tag{5.4}$$

where V_1, V_2 and V_3 are the vertices of the triangle and (u,v) are the barycentric coordinates which fulfill the following conditions:

$$u \geq 0 \qquad v \geq 0 \qquad u+v \leq 0 \tag{5.5}$$

Denoting $\vec{E}_1 = V_2 - V_1$ and $\vec{E}_2 = V_3 - V_1$ the distances between vertices, $\vec{T} = \vec{p}_{ALT_{LTF}} - V_1$ the distance between ray origin and first vertex, the barycentric coordinates can be computed as:

$$u = \frac{\vec{P} \cdot \vec{T}}{\vec{P} \cdot \vec{E}_1} \tag{5.6}$$

$$v = \frac{\vec{Q} \cdot \vec{d}}{\vec{P} \cdot \vec{E}_1} \tag{5.7}$$

where $\vec{P} = \vec{d} \times \vec{E}_2$ and $\vec{Q} = \vec{T} \times \vec{E}_1$. If the barycentric coordinates do not respect the condition given by (5.5), the algorithm skips to the next triangle of the mesh until it finds the beam-ground intersection. In the last case the range is equal to:

$$R = \frac{\vec{E}_2 \cdot \vec{Q}}{\vec{P} \cdot \vec{E}_1} \tag{5.8}$$

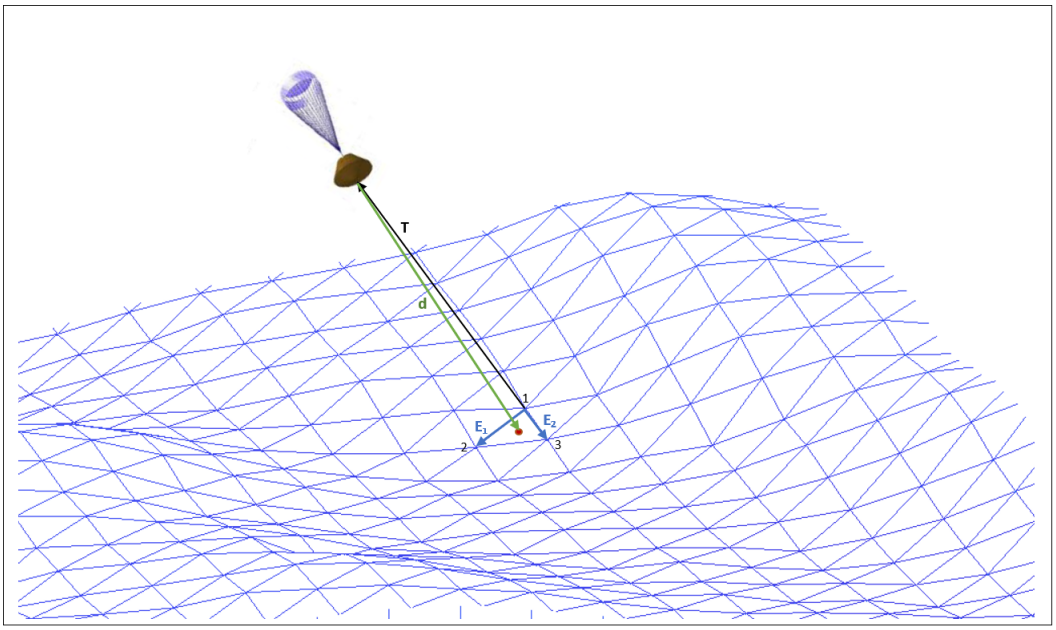


Figure 5.2: Illustration of Ray/Triangle intersection method

5.2 Modeling and Simulation in Simulink Environment

This paragraph provides a description of the development of Altimeter Sensor Model in Matlab/Simulink environment. The model, named *SEN_ALTdem*, takes as inputs 2 dynamic states and 7 parameters to compute the altitude range. Furthermore, an additional output parameter called *valflg_i* provides to simulator a validity flag[1] for the range measurement. The flag value is equal to 1 if the altimeter ray intersects the DEM whereas it is null if there is no beam-ground intersection and the measurement is not taken into account. The input/output time tags are not considered part of the model.

Tables 5.1, 5.2 and 5.3 summarize the input/output interface of the model.

Input	**Name**	**Type**	**Unit**
Position in Planet Fixed frame (FR)	Position_vec_FR	Vector	m
Quaternion between FR and FB	q_FR_FB	Vector	—

Table 5.1: Inputs of Altimeter model

Parameter	**Name**	**Type**	**Unit**
Boresight direction in FB	Boresight	Vector	—
Number of DEM triangles	DemVertexSize	Scalar	—
Position of Vertex 1 in LTF	LocalTerrainVertex1	Matrix	m
Position of Vertex 2 in LTF	LocalTerrainVertex2	Matrix	m
Position of Vertex 3 in LTF	LocalTerrainVertex3	Matrix	m
Position of LTF origin in FR	LocalTerrainPosition_vec_FR	Vector	m
Quaternion between LTF and FR	q_FR_LTF	Vector	—

Table 5.2: Parameters of Altimeter model

Output	**Name**	**Type**	**Unit**
Altitude range	range	Scalar	m
Validity flag	valflg_i	Scalar	—

Table 5.3: Outputs of Altimeter model

[1] In computer programming, a flag is a variable that can assume only two states: 0 or 1.

The coordinates of Local Terrain vertices, defined by *LocalTerrainVertex* parameters, are conveniently set to allocate the values of PANGU DEM to a longitude-latitude equispaced grid. Regarding the model axes convention, the boresight vector is rotated through the quaternion q_FR_FB from Body Axis System (FB) to Planet Fixed Frame (FR), in order to accurately perform the Ray/Triangle intersection algorithm.

Downstream of Altimeter model, there is a *RangeNoiseModel* block, which adds a random signal (Gaussian noise) to the altitude range value, and a block called *FDIR* which checks range measurement validity through $valflg\_i$ parameter.

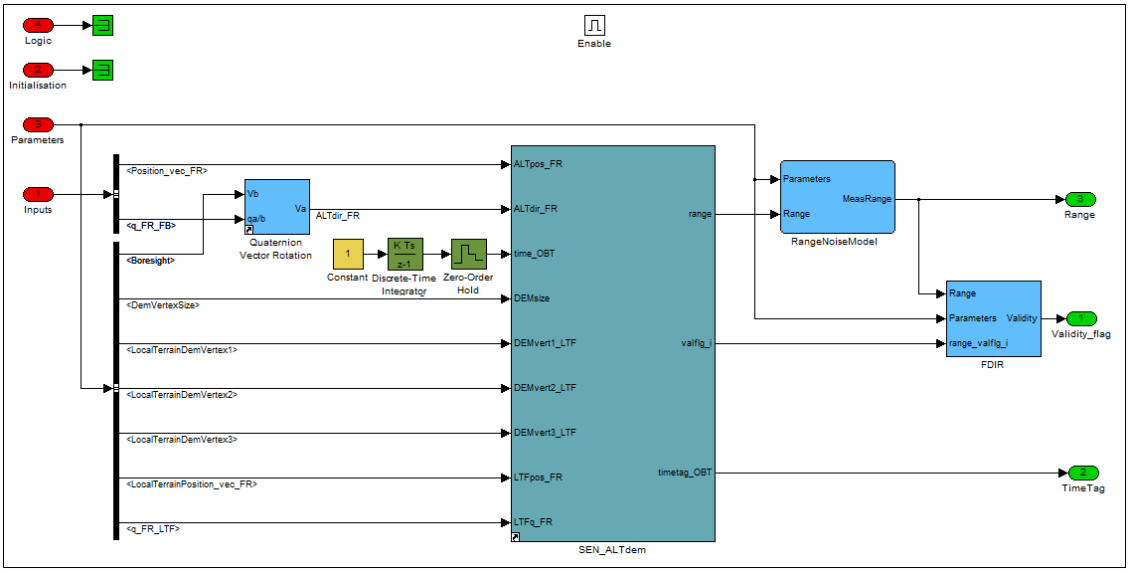


Figure 5.3: Altimeter model (SEN_ALTdem) in simulation environment

During the atmospheric entry phase, the Altimeter Sensor is not yet operational because of Front Shell deployment: it is later activated after 40 seconds from parachute deployment in agreement with the set mission strategy. In simulation environment, this condition is managed by MVM block through an *enable port*, as shown at the top of figure 5.3.

The altimeter model has been developed using Embedded Matlab Function block. It includes only a function able to perform the Ray/Triangle intersection algorithm: $SEN\_ALTdem\_mfun(DEMvert1\_LTF, DEMvert2\_LTF, DEMvert3\_LTF, ALTpos\_FR, ALTdir\_FR, LTFpos\_FR, LTFq\_FR, time\_OBT, DEMsize, DEMsize\_max)$, which takes as inputs the position and attitude of S/C, DEM height information and boresight direction to compute the altitude range.

As the Atmosphere model, the Altimeter model execution time has been examined through Matlab profiler tool in order to verify the real-time performance.

Function List

Name	Time		Calls	Time/call	Self time	
sim	750.47401070	100.0%	1	750.4740107000000	0.00000000	0.0%
ModelExecute	579.65291570	77.2%	1	579.6529157000000	16.28650440	2.2%
MajorOutputs	313.29680830	41.7%	3976	0.0787969839789	0.04680030	0.0%
EAGLE Simulator (Output)	313.25000800	41.7%	3976	0.0787852132797	40.48225950	5.4%
Integrate	248.41599240	33.1%	3975	0.0624945892830	0.10920070	0.0%
MinorOutputs	247.08998390	32.9%	19875	0.0124322004478	0.06240040	0.0%
EAGLE Simulator (MinorOutput)	247.02758350	32.9%	19875	0.0124290608050	89.41977320	11.9%
ModelInitialize	170.74309450	22.8%	1	170.7430945000000	170.72749440	22.7%
EAGLE Simulator/BodyTemplate/Sensors/Lidar/Enabled Lidar Model/SEN LDRdem/SEN LDRdem emf/ SFunction (Output)	115.08193770	15.3%	2	57.5409688500000	115.08193770	15.3%
EAGLE Simulator/BodyTemplate/Sensors/Lidar/Enabled Lidar Model/SEN LDRdem/SEN LDRdem emf (Output)	115.08193770	15.3%	1	115.0819377000000	0.00000000	0.0%
EAGLE Simulator/BodyTemplate/Sensors/Lidar/Enabled Lidar Model (Output)	115.08193770	15.3%	3976	0.0289441493209	0.00000000	0.0%
EAGLE Simulator/BodyTemplate/GNC/HDA/HDA/CAMERA-BASED HDA (Output)	30.17059340	4.0%	3976	0.0075881774145	0.00000000	0.0%
EAGLE Simulator/BodyTemplate/Sensors/Altimeter/DEM Based Altimeter/Sensor (Output)	27.03497330	3.6%	3976	0.0067995405684	0.21840140	0.0%
EAGLE Simulator/BodyTemplate/Sensors/Altimeter/DEM Based Altimeter/Sensor/SEN ALTdem/SEN ALTdem emf (Output)	26.09896730	3.5%	1303	0.0200299058327	0.03120020	0.0%
EAGLE Simulator/BodyTemplate/Sensors/Altimeter/DEM Based Altimeter/Sensor/SEN ALTdem/SEN ALTdem emf/ SFunction (Output)	26.06776710	3.5%	3909	0.0066686536454	26.06776710	3.5%
EAGLE Simulator/EnvironmentTemplate/AtmosphereModel/Atmosphere/LIB ATMcomplete/S-Function Builder (Output)	24.25815550	3.2%	47702	0.0005085353968	24.25815550	3.2%
EAGLE Simulator/BodyTemplate/Sensors/Inertial Measurement Unit/Inertial Measurement Unit 1/IMUmodel (MinorOutput)	22.26134270	3.0%	19875	0.0011200675572	0.57720370	0.1%
EAGLE Simulator/BodyTemplate/GNC/HDA/HDA/HDA HAZARDMAPS/HDA HAZARDMAPS sfcn (Output)	14.94489580	2.0%	3976	0.0037587766097	14.94489580	2.0%

Figure 5.4: Illustration of block profiling layout for Altimeter model simulation

5.3 Validation Activity

The outputs of Altimeter model have been compared with those of an available model in order to validate its accuracy and reliability. The flight simulation chosen to test the two models has consisted in a classical entry, descent and landing maneuver, whose duration has been 397.5 seconds. All the data have been acquired using 'To file' Simulink blocks so as to save the outputs of both models into .mat files. The values of range and absolute error have been then plotted as a function of simulation time. For convenience, the output range of the reference model is labelled as "reference simulation" and the output range of the new altimeter model as "TASI model". We can notice that the model has been enabled after 267.3 seconds from the beginning of simulation.

As shown in figure below, altitude range values of the new altimeter model overlap data of reference simulation and the absolute error remains under 1% threshold. According to these results, the validation test has been considered satisfactory.

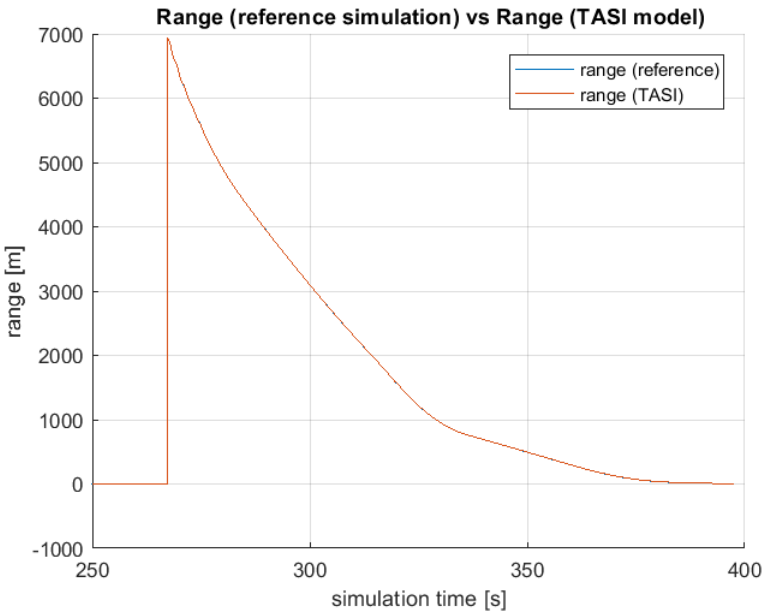


Figure 5.5: Comparison of range values

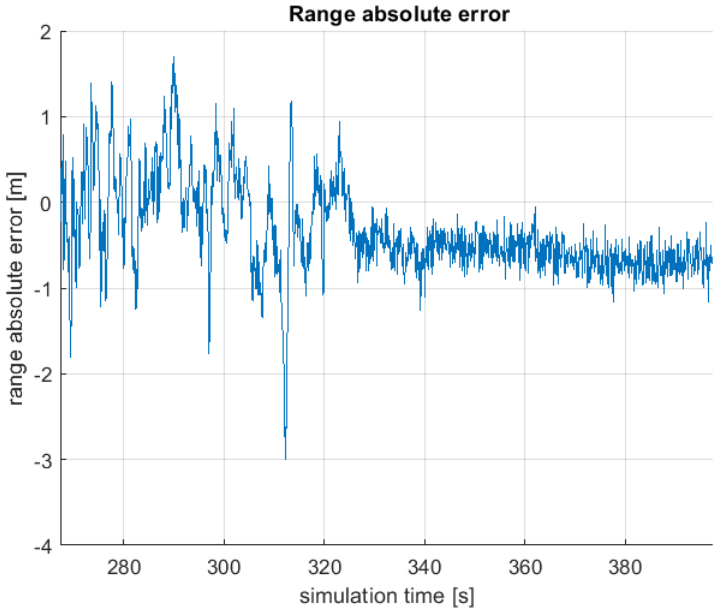


Figure 5.6: Absolute error of range

Chapter 6

Integration in dSPACE Real-Time Environment

6.1 dSPACE Platform Role in an Avionics Test Bench

This section illustrates the architecture of the avionics test bench used to host, test and validate hardware and software elements of flight avionics system.

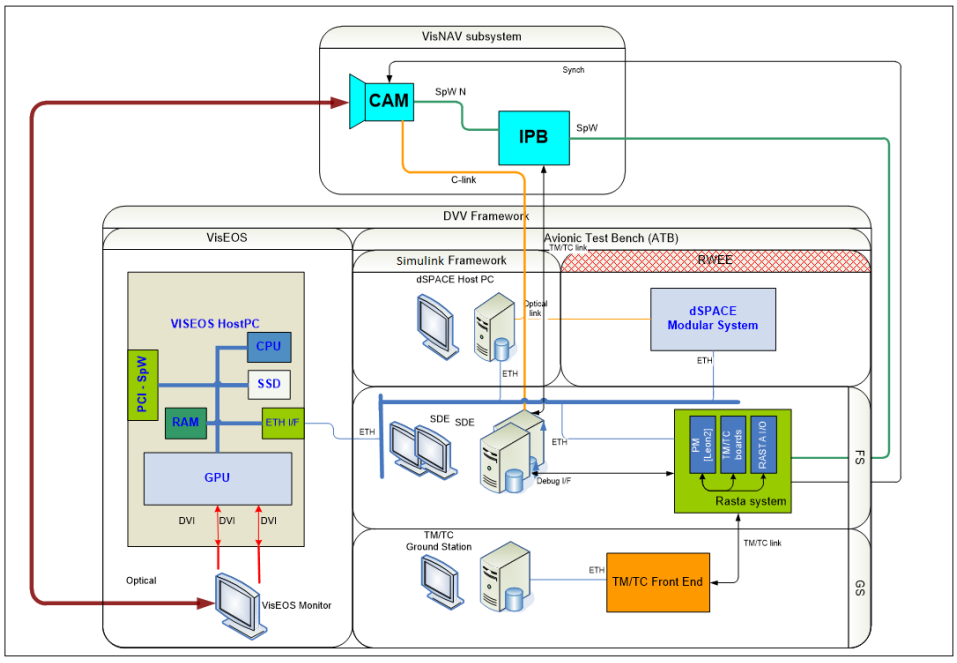


Figure 6.1: Illustration of the avionics test bench architecture (*Courtesy of TASI*)

Figure 6.1 shows the baseline architecture of the avionics test bench, consisting of two main components: the VisNAV subsystem, which includes all the HW and SW components useful to perform and test the visual-based navigation, and the Development, Verification and Validation (DVV) framework, which contains the spacecraft on-board systems, the external environment and the Ground Segment. Models of actuators, dynamics, sensors, environment and deployables, including the two models developed in this thesis, are integrated into a dSpace platform, which acts as a Real World Environment Emulator (RWEE), providing an external environment interface to the spacecraft on-board software and hardware. At a high level, the avionics test bench setup includes the following processing units and components:

- **dSPACE**: Computer capable of running Simulink models for which C code has been automatically generated.
- **Flight Segment (FS)**: Flight-like processor representing the On Board Computer responsible for executing the GNC functions and the other On Board Software tasks.
- **Ground Segment (GS)**: Workstation PC hosting the RAMSES mission control system SW capable to send and receive TM/TC signals.
- **VisEOS**: Optical stimulator of the Camera Functional Model generating images of Mars surface model displayed on a screen.
- **IPB**: Image Processing Board divided into a SW section (PowerPC), responsible for executing the HDA functions, and a HW section (FPGA), responsible for executing the FEIC image processing functions.

The Ground Station is connected via Ethernet link to the Telemetry and Telecommand Electrical Ground Support Equipment box (EGSE), able to communicate with the Flight Segment. The RASTA system furnishes the interface to LEON processor, representing the flight on-board computer, with other components of the bench, i.e. dSPACE, Ground Segment, VisEOS and VisNAV subsystem. A Windows XP, 32 bit Operating system PC is used to upload the SW into the LEON processor RAM. The VisEOS configuration consists in an optical stimulation system used to test the VisNAV subsystem by displaying images of Mars ground scenario on its Monitor. dSPACE platform provides Position and Attitude of the Camera via Ethernet link to VisEOS SW.

After being captured by the Camera, the image is sent via SPW to the IPB HW Section where FEIC and PowerPC are ready to process it. The IPB SW Section PowerPC processor is responsible of detecting whether an image is successfully

decoded, and running the Hazard Detection and Avoidance algorithm. The outputs of HDA are then sent to the OBC via SPW and processed by the GNC SW. Finally, the GNC system sends a force and torque command via Ethernet to the dSPACE platform, able to compute new S/C position and attitude.

The dSPACE environment allows a very accurate scheduling of simulator tasks and a simplified time management of the multicore processor, optimizing the execution time and giving deterministic traffic on the Ethernet link. The Processor Module board is the DS1006 mounting a quad-core processor and the communication board is the DS4504 Ethernet Board. The dSPACE extension box is linked via an optical cable to a Windows 7 (64bit) Workstation, which contains the Simulink environment used to automatically generate C code and cross compile it for the dSpace platform.

6.2 Integration Activity

In order to perform simulations in dSpace environment, a real-time application has been generated from altimeter and atmosphere Simulink models. This process has been the same for both models.

The first step has been the creation of a standalone model by replacing input/output buses respectively with Source Blocks from Workspace, containing input values saved from reference simulation, and Scopes.

In order to save memory on dSpace hardware, further adaptations have been done:

- Spatial reduction of LUTs size in longitude, latitude and altitude in line with trajectory pattern;
- Only 2 Martian seasons out of 12 have been included in the model in agreement with maneuver limited duration (algorithm interpolates variables between 2 neighboring seasons);
- Only one dust/solar scenario has been considered.

In this way, the size of database has been reduced from 5.46 GB to 1.59 MB.

The model has been then converted into an Atomic Subsystem block with the addition of Data Store Memory and Data Store Read blocks, available in Simulink libraries, in order to generate a global variable for execution time measurement of the real-time C code.

The real-time C code is divided into 7 different standard functions (e.g. output function, initialize function, etc...) which can be modified before the model building using Simulink blocks from Custom Code library.

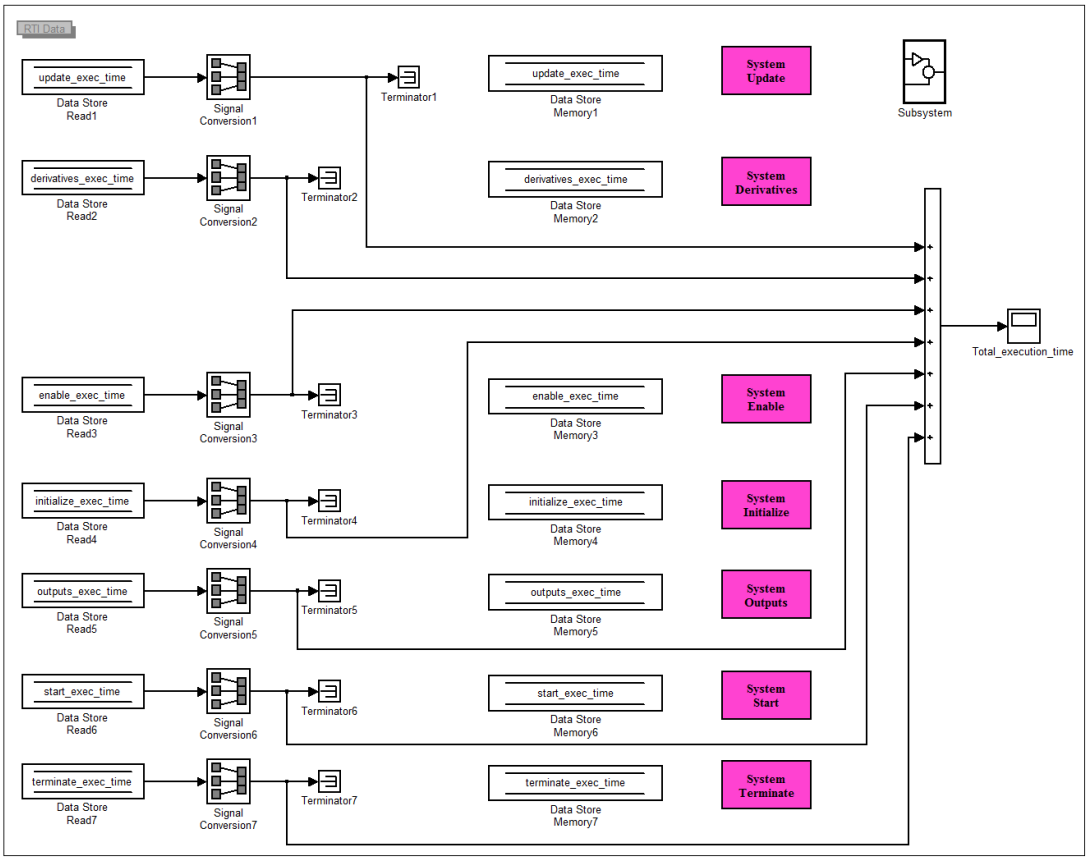


Figure 6.2: Simulink model for real-time simulation

In each of these blocks, shown in figure 6.2 in magenta background, functions for time profiling have been included. The computation time for each function is written in *subsystem_ exec_ time* variable: the sum of all these variables represents thus the total execution time of the model.

Finally, the model created with Simulink has been built using the Real-Time Workshop, able to automatically generate C code from the model. The generated C code has been cross-compiled into an executable application for the dSpace real-time processor. The real-time application has been then loaded to program memory via dSPACE ControlDesk 3.7.1. A system description file (SDF) has been used to upload the application with all related files.

Once the executable files have been loaded, the real-time processor automatically starts the application. To observe all data during the simulation, a layout window equipped with virtual instruments and displays has been added to the experiment. The simulation has been started in animation mode and all the selected variables have been captured during the run, including the total execution time.

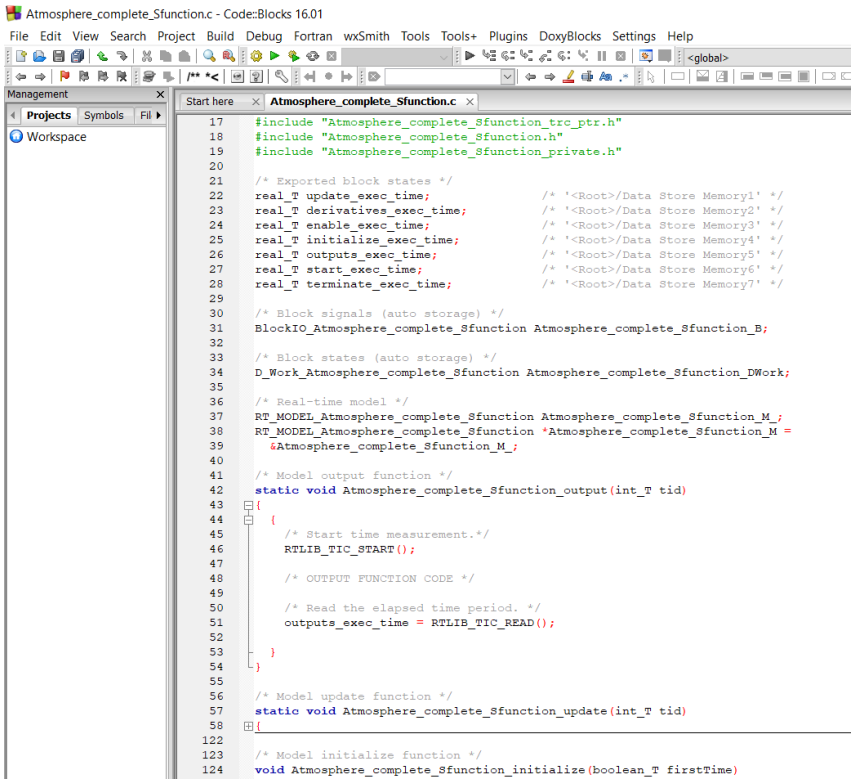


Figure 6.3: Example of a real-time code template

At the end of the simulation, all data acquired have been saved as .mat files and compared with data captured from the correspondent Simulink model.

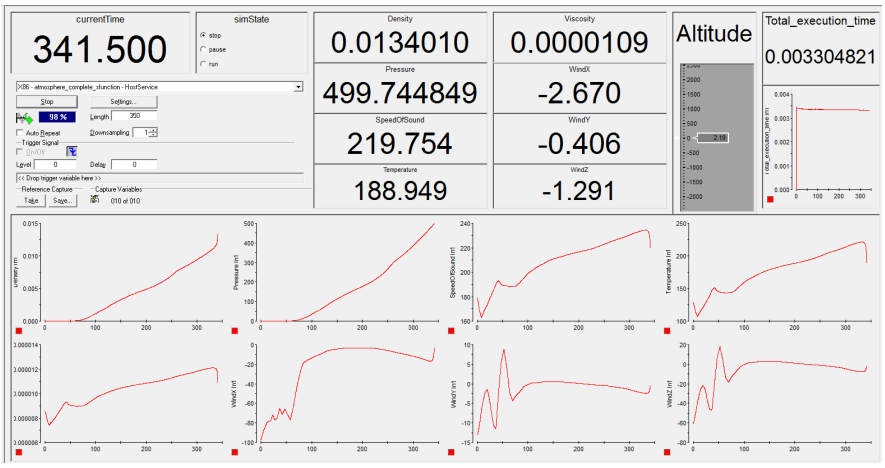


Figure 6.4: Control desk layout for atmosphere model

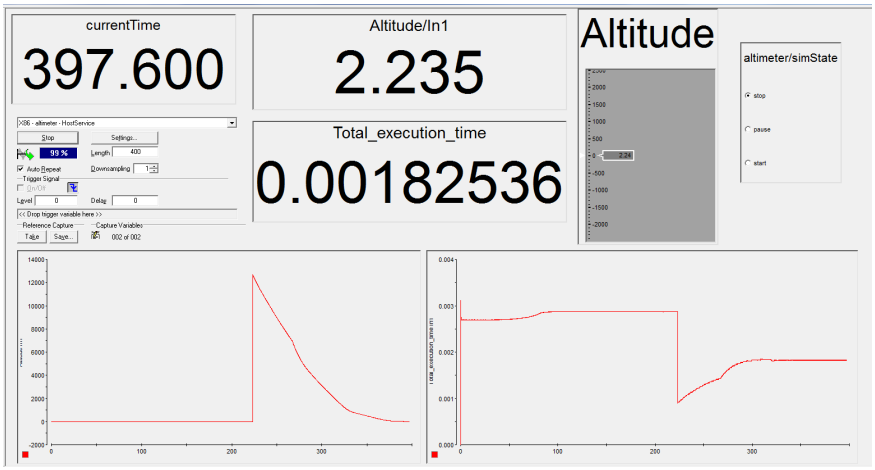


Figure 6.5: Control desk layout for altimeter model

Test campaign results have demonstrated that errors between dSpace environment and Simulink are negligible. Nevertheless, the main objective of this test has been the measurement of model total execution time. The values of this time have been collected for each simulation cycle in order to appreciate model worst case performance and behavior.

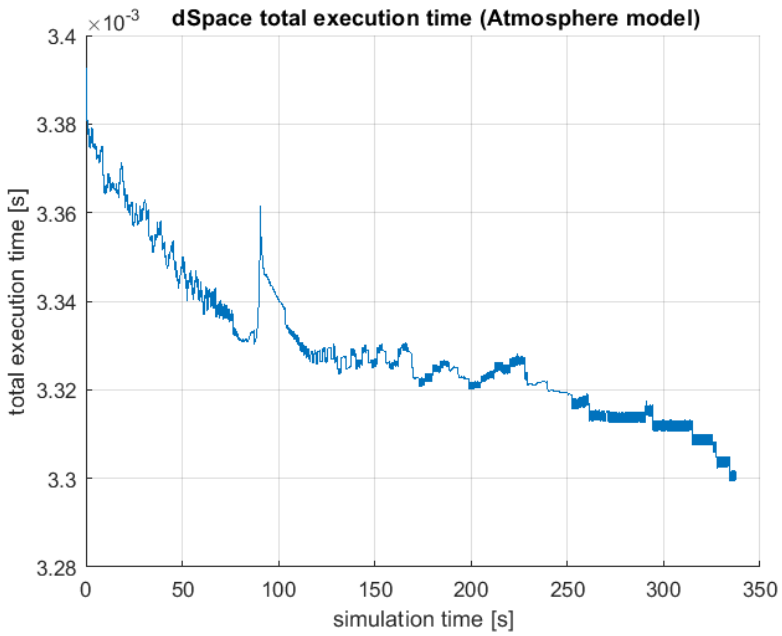


Figure 6.6: Total execution time of Atmosphere model

For the Atmosphere model, worst case is located at the first time step, where the model takes 3.40 ms to compute outputs, while the average execution time is 3.33 ms. It can be noted from figure 6.6 a decreasing trend of execution time throughout the simulation. Indeed, algorithm gradually builds a data column from local surface to spacecraft altitude to compute meteorological variables: as a consequence, the curve steadily decreases during the descent phase as less data are being evaluated. The execution time reaches a peak at 90.3 seconds of simulation time due to a higher time consuming during interpolation of encompassing grid values. However, all the peak values are negligible compared to the worst case.

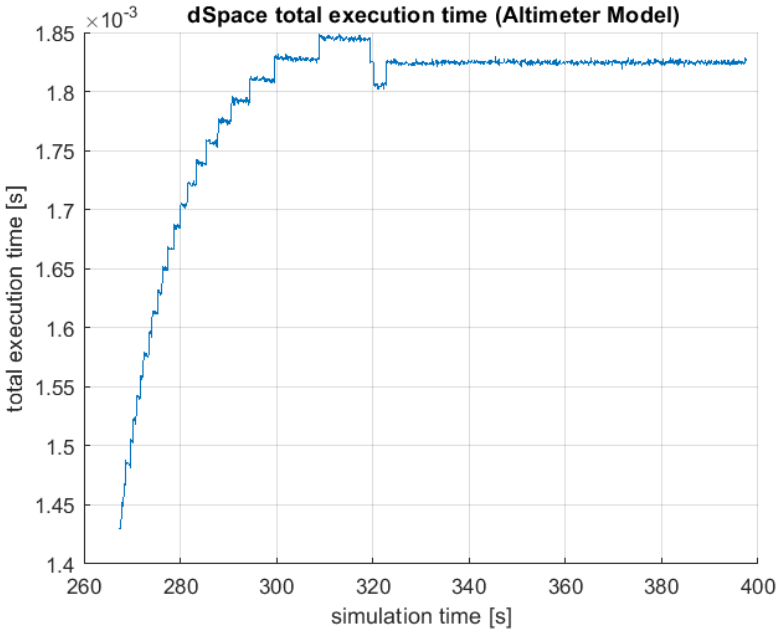


Figure 6.7: Total execution time of Altimeter model

The execution time of Altimeter model depends on the position of S/C in Local Terrain Frame. If the beam-ground intersection lies in first triangles of mesh, the execution time can be very low whereas it grows significantly as the algorithm tests the intersection for more triangles. In this case, the worst condition is equal to 1.85 ms, and the model takes 1.79 ms on average to compute the range.

Chapter 7

Conclusion and Future Work

The scope of this thesis has been the development of a Mars Atmosphere model and an Altimeter Sensor model to be integrated in a flight simulator designed to support the users in studying and simulating a Mars Entry, Descent & Landing mission scenario.

To reach this scope, the following steps have been accomplished:

- investigate new mathematical methods and representations for Mars Atmosphere and Altimeter Sensor models;
- develop both models in Simulink environment, using Matlab tools and C language;
- validate models' outputs and measure accuracy and execution time performance;
- upload the models into a dSpace platform in order to set-up real-time simulations with an avionic test bench including OBC and HW sensors (HIL).

This project has been developed in all its phases: design, coding, testing, bug fixing, integration in real-time hardware platform, optimization to reduce both memory size and execution time.

The introductory chapter of this thesis provides an overview of the successful past landing mission on Mars and a description of the mission scenarios and Mars' atmosphere and climate. The core of thesis is composed by three chapters (3rd, 4th, 5th).

The third chapter describes the model-based simulator architecture in which the new Atmosphere and Altimeter models have been successfully integrated.

The fourth chapter illustrates the mathematical methods used for Atmosphere model, its development and validation. The obtained results have been compared

versus previously developed reference models validated against flight data. The validation test has been considered satisfactory as the absolute error between the developed model and the reference one remained under 1% threshold.

The fifth chapter presents the activity carried out for the Altimeter model, which obtained results have been validated following the same procedure of the Atmosphere one.

The sixth chapter is dedicated to discuss how the integration in the DVV framework has been carried out (dSpace environment) to include the models in a real-time closed loop simulation. Cross compilation on dSpace high performance platform has implied optimization of the generated codes and of the databases with the final objective to remain within an acceptable execution time. The Mars Atmosphere model and Altimeter Sensor model have reached an average execution time equal to respectively 3.33 ms and 1.79 ms.

Future works could aim to improve the reliability of the two models. Regarding the Atmosphere model, a better estimation of climatic statistics could provide an accurate representation of critical Mars meteorological phenomena such as dust storms and dust devils. The Altimeter model could be improved using a denser surface mesh in order to better identify hazard points like stones and cliffs.

Future additional research activities could be undertaken to reduce models' computation time and constraints in order to increase the software potentialities and flexibility. The main goal could be to find a good compromise between the models' quality (resolution) and the computational effort suitable for closed loop simulations with HIL and at high frequency (10 Hz, 40 Hz). Indeed modeling and simulations are fundamental for the realization of a space exploration mission. They conduct a critical role for the design of the flight segment (HW+SW) and for the mission performance assessment.

Bibliography

[1] E. Bell. *Mars 3 Lander*. Retrieved from https://nssdc.gsfc.nasa.gov/nmc/spacecraftDisplay.do?id=1971-049F [Accessed: 18-June-2018]. 2017 (cit. on p. 16).

[2] Brian Dunbar. *Landing Accuracy on Mars: A Historical Perspective*. Retrieved from https://www.nasa.gov/mission_pages/msl/multimedia/pia16039.html [Accessed: 15-June-2018]. 1997.

[3] University of Dundee. *PANGU - Planet and Asteroid Natural scene Generation Utility*. Retrieved from https://www.star-dundee.com/products/pangu-planet-and-asteroid-natural-scene-generation-utility [Accessed: 29-May-2018] (cit. on p. 65).

[4] F. Forget E. Millour and S.R. Lewis. *Mars Climate Database v4.3 - Detailed Design Document*. May 2008 (cit. on pp. 24, 35, 44).

[5] D. E. Smith F. J. Lerch R. S. Nerem M. T. Zuber G. B. Patel S. K. Fricke and F. G. Lemoine. *An Improved Gravity Model for Mars: Goddard Mars Model-1 (GMM-1)*. May 1993 (cit. on p. 42).

[6] *Historical Log*. Retrieved from https://mars.nasa.gov/programmissions/missions/log/ [Accessed: 06-July-2018]. 2018 (cit. on p. 15).

[7] *Mars Pathfinder*. Retrieved from https://mars.nasa.gov/programmissions/missions/past/pathfinder/ [Accessed: 18-June-2018] (cit. on p. 18).

[8] *Mars Science Laboratory/Curiosity*. Retrieved from https://www.jpl.nasa.gov/news/fact_sheets/mars-science-laboratory.pdf [Accessed: 19-June-2018] (cit. on p. 22).

[9] Tomas Möller and Ben Trumbore. *Fast, Minimum Storage Ray/Triangle Intersection*. 1997 (cit. on p. 65).

[10] *Phoenix Landing*. Retrieved from https://www.jpl.nasa.gov/news/press_kits/phoenix-landing.pdf [Accessed: 19-June-2018]. May 2008 (cit. on p. 21).

[11] *Spirit and Opportunity*. Retrieved from `https://mars.nasa.gov/programmissions/missions/present/2003/` [Accessed: 18-June-2018] (cit. on p. 19).

[12] *Viking 1 & 2*. Retrieved from `https://mars.nasa.gov/programmissions/missions/past/viking/` [Accessed: 18-June-2018] (cit. on p. 17).

[13] David R. Williams. *Mars Fact Sheet*. Retrieved from `https://nssdc.gsfc.nasa.gov/planetary/factsheet/marsfact.html` [Accessed: 16-June-2018]. 2016 (cit. on p. 23).