POLITECNICO DI MILANO

Facoltà di Ingegneria Industriale

Corso di Laurea in
Ingegneria Spaziale

A VIRTUAL SIMULATOR FOR PLANETARY ENTRY DESCENT AND LANDING VEHICLES DESIGN

Relatore:
Prof.ssa Michèle LAVAGNA

Correlatori:
Ing. Roberto ARMELLIN
Ing. Riccardo LOMBARDI

Tesi di Laurea di:

Davide SOLIGNO Matr. 720685

Anno Accademico 2009 - 2010

RINGRAZIAMENTI

Arrivato a questo punto, ripenso ai miei anni trascorsi al Politecnico, alle fatiche e alle soddisfazioni dopo ogni esame, ai progetti svolti, alle numerose persone che ho conosciuto tra studenti e professori. È difficile e impossibile riassumere tutto in poche righe; vorrei avere molto più spazio per descrivere ogni cosa e ringraziare personalmente ogni persona. Tuttavia, ci sono alcune persone che mi sono state vicine e che meritano un ringraziamento particolare.

Ringrazio la professoressa Michèle Lavagna per avermi dato l'opportunità di lavorare a questa tesi, per il corso di progetto di sistemi spaziali, ma soprattutto per avermi sempre incoraggiato.

Ringrazio l'ing. Roberto Armellin, per i suoi consigli durante il lavoro di tesi, dagli inizi fino alla stesura finale.

Ringrazio l'ing. Riccardo Lombardi che, durante il suo dottorato, ha trovato il tempo per consigliarmi e fornire un grande aiuto al lavoro di integrazione del programma che ho realizzato.

Ringrazio i miei genitori, per avermi sempre sostenuto e incoraggiato, soprattutto nei momenti di maggiore sconforto. Mi sono sempre stati vicini e il loro supporto è stato fondamentale.

Ringrazio mia sorella, non solo come fratello, ma soprattutto per il suo supporto logistico, avendomi prestato il suo computer quando il mio mi ha abbandonato nei momenti di bisogno (per ben due volte durante questa tesi).

Ringrazio tutto il gruppo dei miei amici, per tutti i momenti di svago e relax che ho passato con loro, dalle semplici serate alle vacanze insieme.

Ringrazio tutti gli amici della sala calcolo, in particolare la Ele, Dario, Mattia e Giovanni, per avermi fatto compagnia durante il mio periodo di tesi.

Ringrazio tutti gli amici dell'atletica, perché è un ambiente fantastico dove potersi rilassare e stare in compagnia.

Ringrazio la Robi, per la sua compagnia durante i viaggi in treno verso Milano.

INDEX

INDEX OF FIGURE

INDEX OF TABLE

ABSTRACT

This Master thesis deals with the development of a graphical simulator for Entry, Descent and Landing missions on a planet surface. The high costs of space missions prohibit large scale experimental tests. Thus, it is necessary to use a computer code to simulate different operative conditions. Graphical visualization helps mission designers to gain more physical insight to the problem. The tool developed in this work deals about the dynamics of atmospheric re-entry, subdividing the re-entry trajectory in three successive phases. First the vehicle, due to the presence of the heatshield, is modeled as a capsule and descents towards the planet braked by aerodynamic drag. Then the parachute is deployed to further brake the capsule. The last phase considers the landing of the vehicle by retro rockets and starts after the release of the parachute and the backshell. Dynamics implemented in the tool considers both inertial and angular motion of the vehicle; forces are computed by modeling environment and aerodynamics, using for the latter fitted data coming from previous missions. The model verification is based on data taken from literature and the trajectory is displayed on the screen using a graphic engine. The software has been developed by using a modular approach: each element can be modified singularly. This allows a future upgrade of the software, to better describe each element that characterizes the re-entry. Thanks to the modular approach, the tool developed is integrated in a bigger project, already started at Politecnico, which aims to realize a graphical simulator to space exploration.

Keywords: atmospheric re-entry, graphic simulation, dynamics

SOMMARIO

Questa Tesi di laurea Magistrale documenta lo sviluppo di un simulatore grafico per missioni di rientro, discesa e atterraggio di un veicolo spaziale sulla superficie di un pianeta. L’elevato costo delle missioni spaziali non permette di provare sul campo tutte le possibili situazioni cui può essere soggetto un veicolo di rientro. Di qui la necessità di utilizzare un altro approccio, basato sulla simulazione al calcolatore delle diverse condizioni operative. La visualizzazione grafica fornisce un valido aiuto ai progettisti di missione, che, oltre ad un riscontro dai dati numerici, hanno la possibilità di un approccio più fisico e realistico al problema. Il simulatore sviluppato in questo lavoro di tesi riguarda la dinamica del rientro atmosferico, suddividendo la traiettoria in tre fasi successive. All’inizio il veicolo, a causa della presenza dello scudo termico, si presenta come una capsula che scende verso il pianeta frenato solo dalla resistenza aerodinamica. Successivamente è dispiegato il paracadute che frena la capsula. L’ultima fase inizia dopo aver sganciato il paracadute e lo scudo termico e prevede l’atterraggio del veicolo tramite l’uso dei retrorazzi. La dinamica considerata nel simulatore riguarda sia il moto del centro di massa che l’assetto del veicolo; le forze agenti sono calcolate mediante una modellazione dell’ambiente e dell’aerodinamica, ottenuta quest’ultima sulla base di dati riguardanti precedenti missioni di rientro. La verifica del modello si basa su risultati noti reperibili in letteratura e la traiettoria è visualizzata a video tramite l’utilizzo di un motore grafico. Il software è stato realizzato seguendo un approccio modulare: ogni elemento che lo compone può essere modificato singolarmente. Questo permette un futuro ampliamento del software, in modo da poter descrivere nel dettaglio ciascun elemento che caratterizza la fase di rientro. Grazie a quest'approccio il modulo realizzato è integrato facilmente in un progetto più ampio, già avviato all’interno del Politecnico, che prevede la realizzazione di un simulatore grafico per l’esplorazione spaziale.

Parole chiave: rientro atmosferico, simulatore grafico, dinamica

Chapter 1

Introduction

Exploring planets of our solar system is a great challenge for space researchers. Since the first years of space activities, the wish of exploring new worlds guided the research in space technology and engineering. Civil and military objectives bring man to explore planets of our solar system with manned and unmanned mission. Moon, for example, represented the desire to conquer and explore; the past and coming years relate to exploration of Mars.
If we want to explore new worlds outside our planet, we need to develop technology and engineering knowledge to reach them, explore, and come back. In this contest, it is necessary a development of an Entry, Descent and Landing (EDL) system to safely reach a planet, a satellite or an asteroid. As mission costs are very high, it is necessary to simulate all possible situations in order to develop a robust architecture for mission; in particular, a key feature is to develop a graphical simulator that allows mission designers to observe and analyze the real operative situations of the spacecraft, lander or rover.
In this thesis a virtual simulator for entry, descent and landing is presented. In particular, the simulator is focused on the dynamics of atmospheric re-entry. This is a part of the entire EDL tool that in future will be developed and so in this chapter a general description of entry, descent and landing system is given. First it is necessary to distingue between non-atmospheric (e.g. Moon) and atmospheric (e.g. Venus or Mars) environment that can characterize a landing mission. Then, graphical simulators developed by industries are presented, underlining differences with the EDL tool developed in this work. The last section of this chapter describes the structure of this thesis.

1.1 Entry, Descent and Landing

Before landing, a spacecraft can be in a circular or in an elliptical orbit around the planet; another option is approaching the planet directly from an interplanetary trajectory. The module that lands on the planet is released by the main spacecraft that can be a simple propulsive and power module used only to reach the planet or a complex spacecraft, which also provides scientific analyses in orbit around the planet. Once the EDL sequence starts, the spacecraft begins its cruise till the surface of the planet. The re-entry problem consists in three major phases:

- De-orbiting phase.

- Re-entry phase.
- Descent and landing phase.

The de-orbiting phase is the arc that allows the spacecraft to leave its orbit and to start the descent to the planet's surface. The re-entry phase is mission dependent and it can be characterized by a non-atmospheric or an atmospheric environment. Also for the final descent and landing phase there are different options as:

- Uncontrolled descent with hard landing.
- Uncontrolled descent with soft or semi-hard landing (by using parachute and airbags).
- Controlled descent with soft landing (with a landing propulsion system, such as in the Lunar Exploration Module).

In the next subsections, typical re-entry problems are analyzed, with a general description of a non-atmospheric and of an atmospheric landing.

1.1.1 Non-Atmospheric

With this definition, we typically refer to a Moon environment and analyze all the problems connected with it. A typical mission profile during landing on the Moon is characterized by three phases (Figure 1.1). The first one is the braking phase, in which a powered descent is performed by using retrorockets to slow the landing module. The propulsive subsystem is the most important in this phase, also with the Attitude Determination and Control System (ADCS) subsystem. While the retrorockets are still firing, a pitch-up maneuver is performed to orient the spacecraft in a vertical position needed for landing. Then, the approach phase starts, characterized by hazards detection and a relative navigation respect to the terrain. Passive sensor such as camera and active sensor as LIDAR (Laser Imaging Detection and Ranging) are used to calculate the relative motion of the lander and to identify obstacles. On Board Data Handling (OBDH) subsystem is very important in this phase because a large amount of data is collected from sensors and a robust guidance algorithm has to assure a soft landing. Dust effect on landing surface cannot be ignored during a descent in the Moon environment; also, mechanical loads have to be considered when the Lander arrives on the surface.

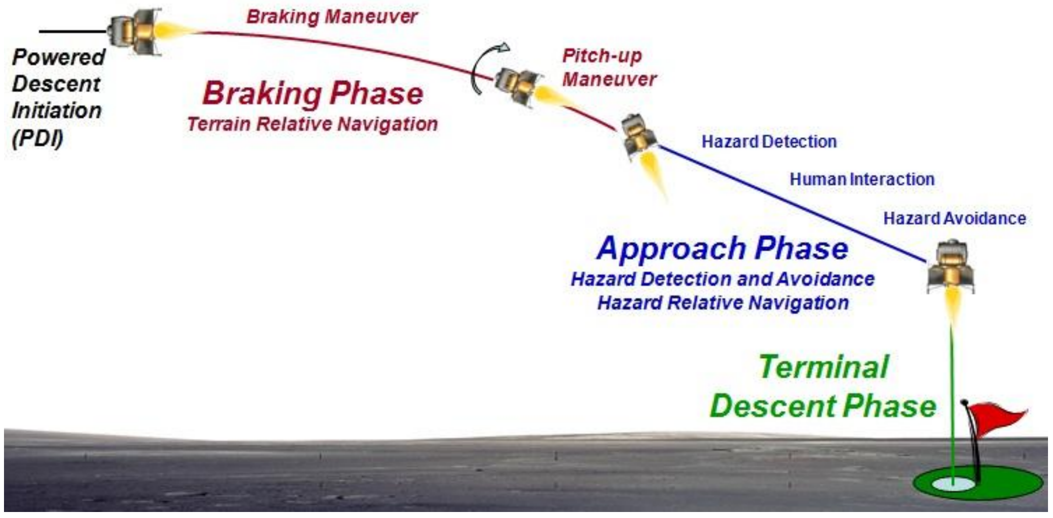


Figure 1.1. Re-entry phases the Moon environment

1.1.2 Atmospheric

A more complex EDL mission is characterized by the atmospheric re-entry. In fact, the presence of the atmosphere requires the spacecraft to pass through different regimes of motion, as described by Gnoffo [**1**]. At the highest altitudes, the interaction of the vehicle with the atmosphere is characterized by free molecular flow approximation; hypersonic Mach numbers are encountered and the energy developed during the motion causes significant chemical reaction (dissociation, ionization). These conditions cause high thermal loads, and thus a heatshield is fundamental to protect the payload. As the vehicle descends a little deeper into the atmosphere, the mean free path between atmospheric molecules decreases and incoming molecules can no longer be ignored; this flow condition is called transitional regime. As the re-entry vehicle (RV) continues its descent, it finally encounters the continuum regime, which is governed by the Navier-Stokes equation. In this regime, the motion is governed by aerodynamic loads, and so it is necessary to compute the correct aerodynamic coefficients.

1.2 Graphical simulators for Entry Descent and Landing

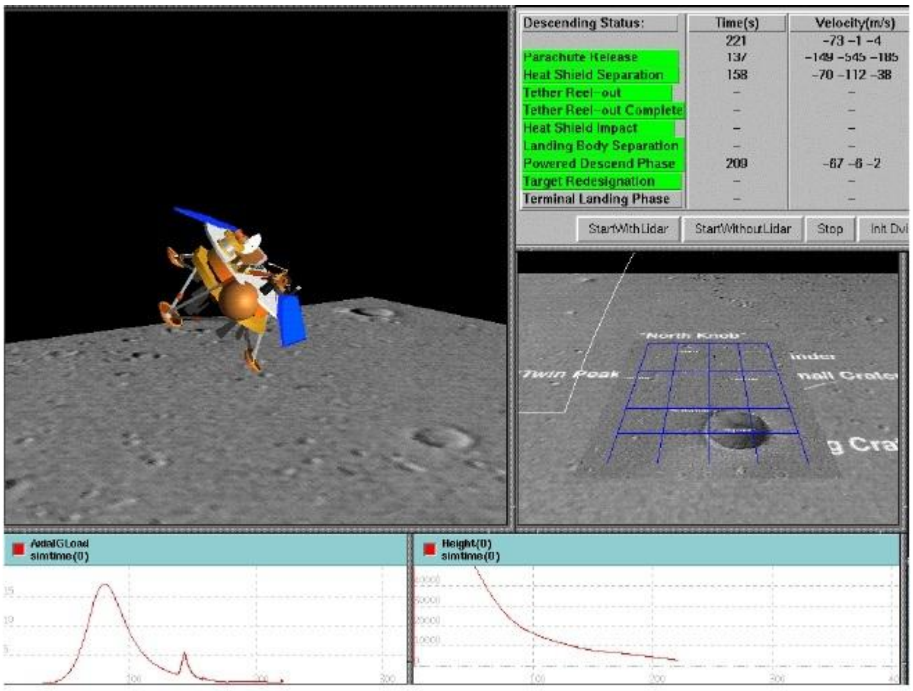
As reported at the beginning of this chapter, the main task of this work is to build a graphical simulator to support the design of entry, descent and landing vehicles. In the followings some of the simulators developed by industries are described.

DSENDS

The Dynamic Simulator for Entry, Descent and Surface landing (DSENDS) is capable of modeling spacecraft dynamics, device and subsystem and it is in use by interplanetary and science missions such as Cassini, Galileo, Starlight ([**2**] [**3**]). DSENDS is developed by the Jet Propulsion Laboratory (JPL) and it is composed by different tools. These ones involve a dynamic engine and models for aerodynamics, atmosphere, parachute, terrain and instruments. The simulator is constructed using a modular architecture, whose blocks are built using Matlab®, Simulink® and C++. DSENDS run under Solaris, Iris and Linux operating system.

Its main features are:

- A real time simulation, with the capability to allow extensive check out of the flight software.
- A high fidelity simulation, with the capability to realistically capture the relevant physics and device interactions.
- A high fidelity aerodynamic library for a detailed determination of aero-coefficients across various spacecraft attitude configurations.
- Different terrain models represented as Digital Elevation Maps (DEM).
- A rapid switching and fetching of terrain maps as the simulation evolves.
- Instrument simulations. For example, a library of routines provides to perform a LIDAR simulation and its interaction with the DEM.



Descending Status:	Time(s)	Velocity(m/s)
	221	-73 -1 -4
Parachute Release	137	-149 -545 -185
Heat Shield Separation	158	-70 -112 -38
Tether Reel-out	–	–
Tether Reel-out Complete	–	–
Heat Shield Impact	–	–
Landing Body Separation	–	–
Powered Descend Phase	209	-67 -6 -2
Target Redesignation	–	–
Terminal Landing Phase	–	–

Figure 1.2. A landing simulation from DSENDS

Dspace
The Dspace simulator is an interactive 3D visualization system that performs different visualizations referred to a large variety of space missions, including the entry, descent and landing [**4**]. Dspace is developed by JPL, and it interfaces also with DSENDS. It is built with an object oriented architecture developed in C++ and Phyton and uses Coin3D library and OpenGL as video devices.
Dspace main features include:

- Synthetic camera modeling, with the possibility to modify camera viewpoint information by mouse interaction.
- High performance terrain visualization.
- Real time shadows.
- Accurate representation of spacecraft vehicle kinematics, multiple scene viewpoints, image-based horizons detection and line of sight modeling.

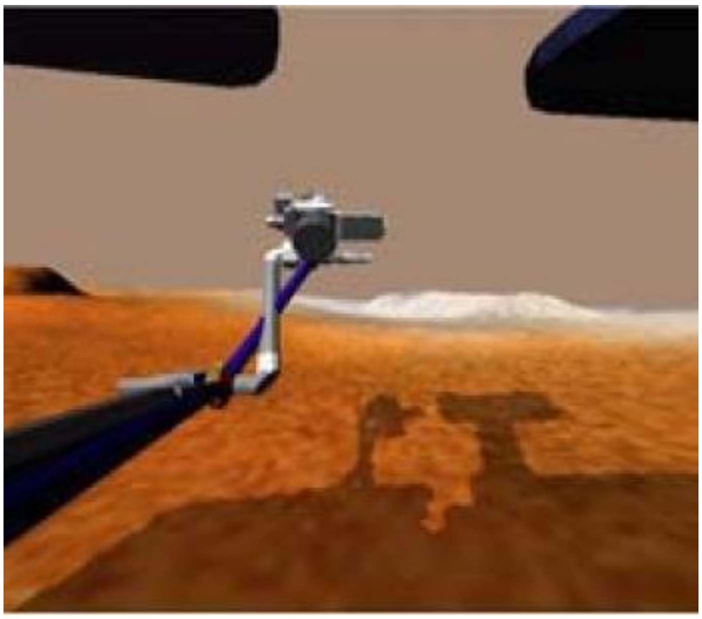


Figure 1.3. A camera view and shadows from Dspace

EAGLE
The Entry and Guided Landing Environment (EAGLE) is an environment for system engineers, which allows to simulate EDL missions [**5**]. It is developed by the Scisys under contract of the European Space Agency (ESA). The simulation kernel is implemented in Matlab® and Simulink®, but the simulator can interface with other commercial and open source software. The EAGLE architecture is based on six sub libraries: dynamics, environment, mathematics, actuators,

control, and sensors. These libraries build the whole simulator. Its main features are:

- A simulator that can be used equally in each of phase of the design of the system engineering lifecycle.
- Different level of simulation fidelity, that increases as the mission design matures.
- Available EDL simulations for Venus, Earth, Mars and Titan.
- Both variable and fixed step propagation of the continuum time states.
- An optimized multi-body dynamic.

Figure 1.4. A lunar approach from EAGLE

HyperPASS & HyperCMST

Global Aerospace Company (GAC) has participated in mission design and analysis including high-mass Mars transit and entry systems [**6**]. Several software tools have been developed by the GAC to aid in these analyses. The Hypersonic Planetary Aeroassist Simulation System (HyperPASS) is a software package that use Matlab$^{®}$ language. It enables users to perform simulations of EDL at various planets. Its major features include:

- Mission studies of aerocapture systems at planets with atmosphere.

- Trade studies to investigate performance with alternate aeroshell types, varying flight path angle and entry velocity, different g-loads limits, angle of attack and bank variation.
- Educational purpose because of its ease of use.

The Hypersonic Control Modeling & Simulation Tool (HyperCMST) provides trajectory, control and optimization in an each single tool developed by GAC.

1.3 Motivations

The software previous described are used by industries and research centers. Graphical visualization helps engineer to design different kind of missions; in particular, software used to simulate EDL sequence are considered. The EDL tool developed in this work differs from those previously mentioned in some important aspects.
First of all, the software previous described are not free of charge. They are property of industries that have developed them for commercial purposes; also, lots of software is built by using third-party programs, such as Matlab®. Thus, to use them, a license is required. The EDL tool developed in this work is open source. Its source code can be used and modified by everybody. Everyone can upgrade it, by adding tools or modifying the existent ones. This task is accomplished using the C++ language with an object oriented programming and adopting a modular architecture.
Another reason to develop the EDL tool is that Politecnico di Milano is missing such a tool. This lack forces Politecnico to use third-party software to analyze the re-entry problems. These analyses can be requested by industries or by student’s project. Having its own software, Politecnico can save money and be independent of software industries.
Furthermore, the EDL tool can be used for educational purposes. Students can have a vision of what they are studying and also use the software to apply what they have learned.
Finally, the EDL tool is part of a bigger project, which involves the realization of a planetary simulator to support different kind of mission design, ranging from the orbital mechanics to rovers moving on the planetary surfaces. A peculiarity of the tool is that it can be easily integrated with the other parts of the simulator.

1.4 Thesis structure

This thesis presents the development of the EDL tool, from its general definition to its validation and graphical representation of an EDL sequence. It is organized in various chapters:

- Chapter 2. Presents the EDL problem referred to an atmospheric environment, focusing on previous missions on Mars and defines the EDL sequence considered in this work.
- Chapter 3. Define the mathematical models used to simulate the dynamics together with the hypotheses and assumptions made.
- Chapter 4. Presents the software architecture and explains the C++ implementation of its models.
- Chapter 5. Validates the EDL tool by comparing its results with the ones coming from other sources.
- Chapter 6. Describes how the EDL tool is integrated in the whole simulator and its graphical view.
- Chapter 7. Contains final remarks and possibly future extension of this work.

Chapter 2

Atmospheric re-entry

As previously reported, the task of this thesis is to develop a graphical simulator for an EDL system, considering an atmospheric environment. First of all, it is necessary to study past EDL missions and analyze how the landing problem was solved. In particular, past missions on Mars are considered. Mars is chosen as a reference planet: past and future manned and unmanned missions focused on this planet and thus lot of reference literature can be used to validate the tool. Once the tool has been validated, the modular structure created with the object oriented programming in C++ allows using the EDL tool to simulate a re-entry on other planets, simply by changing the mission scenario.
In this chapter, first a description of the main missions that safely land on the surface of Mars is given focus on their mission architecture ([**7**], [**8**]). Then, the mission architecture considered in developing the tool is presented.

2.1 Previous mission on Mars

In this section, a brief overview on the past missions that safely land on Mars is presented [**9**], focusing on those that represented milestones in discovering the red planet.

Viking 1 & 2
NASA's Viking Mission to Mars was composed of two spacecraft, Viking 1 and Viking 2, each one consisting of an orbiter and a lander. The two Viking safely landed on Mars in 1976 and transmitted images of the surface, took samples and analyzed them for composition and signs of life, studied atmospheric composition and meteorology, and deployed seismometers [**10**]. The technology adopted for entry, descent and landing became the backbone for all EDL missions. The Viking landers started their re-entry cruise with a capsule shape (Figure 2.1), defined by a 70 deg sphere cone aeroshell made on ablative material. The back of the capsule was protected by a backshell. This shape assures a high hypersonic drag coefficient to slow the vehicle. At 6 km of altitude at about 250 m/s, the 16 m diameter parachutes were deployed. Seven seconds later the aeroshell was jettisoned, and 8 seconds after that, the three lander legs were extended. At 1.5 km altitude, retro-rockets were fired until the vehicle landed on the surface.

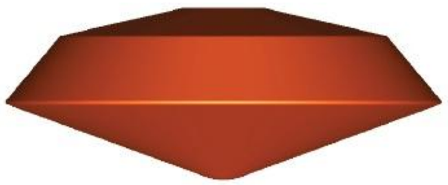
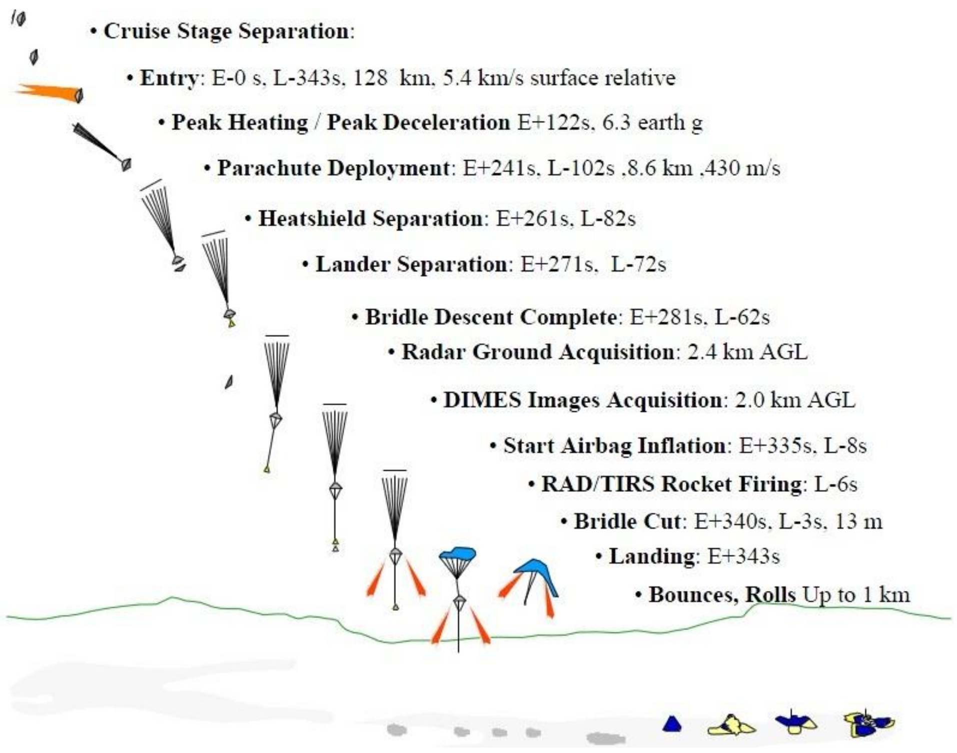


Figure 2.1. Viking capsule shape

Mars Pathfinder

The primary objectives of Mars Pathfinder mission (MPF) was to demonstrate an innovative, low cost and reliable method for placing a science payload on the surface of Mars [**11**]. The Pathfinder spacecraft entered the Martian atmosphere in the 1997 directly from the Earth-to-Mars interplanetary entry trajectory. The lander shape was derived from the Viking one. The parachute needed to slow the vehicle was deployed at a velocity of 340 m/s. After 20 second, the aeroshell was jettisoned and a bridle with the lander was deployed under the backshell. The system composed by the parachute, the backshell, and the lander flight till an altitude of about 13 meters. Then, airbags were inflated and the bridle was cut [**12**]. The lander fell on the surface, while the rest of the system is carried away by using rockets mounted on the backshell to avoid a backshell/parachute recontact with the lander. This sequence of events is presented in Figure 2.2.


• Cruise Stage Separation:
• Entry: E-0 s, L-343s, 128 km, 5.4 km/s surface relative
• Peak Heating / Peak Deceleration E+122s, 6.3 earth g
• Parachute Deployment: E+241s, L-102s ,8.6 km ,430 m/s
• Heatshield Separation: E+261s, L-82s
• Lander Separation: E+271s, L-72s
• Bridle Descent Complete: E+281s, L-62s
• Radar Ground Acquisition: 2.4 km AGL
• DIMES Images Acquisition: 2.0 km AGL
• Start Airbag Inflation: E+335s, L-8s
• RAD/TIRS Rocket Firing: L-6s
• Bridle Cut: E+340s, L-3s, 13 m
• Landing: E+343s
• Bounces, Rolls Up to 1 km
Figure 2.2. MPF sequence of events

MER A & B

The MPF mission was the outpost to the next ones, the Mars Exploration Rovers (MER) A and B, that safe landed Spirit and Opportunity rovers in 2004 [**13**]. The sequence of events to safely land on the Mars surface was the same of MPF. The MER missions have been very important milestones, in order to explore the planet by moving on its surface and collect scientific data. Figure 2.3 visualize the configuration of MER before and after airbag inflation.

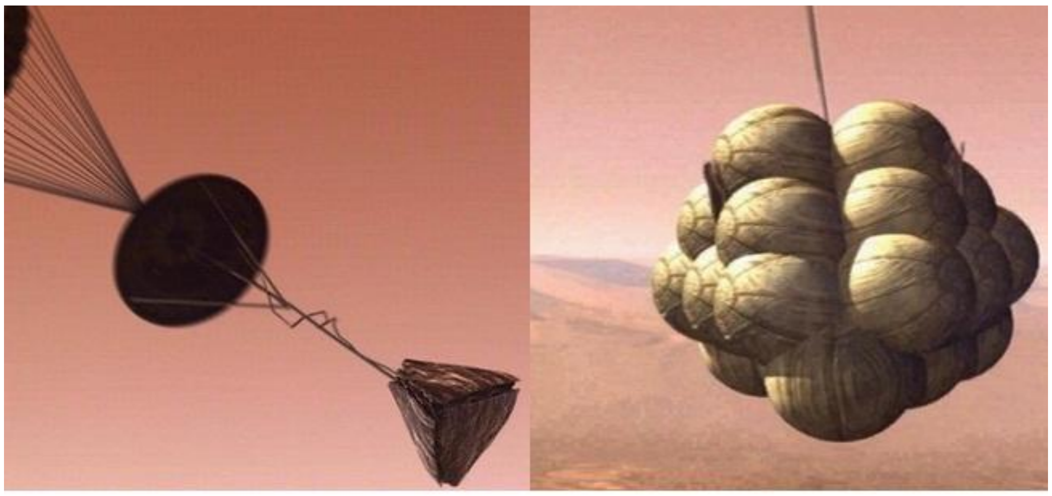


Figure 2.3. MER before and after airbag inflation

Phoenix

The Phoenix Mars Lander is designed to study the surface and near-surface environment of a landing site in the high northern area of Mars. It was launched on 4 August 2007 and arrived on the Mars surface on 25 May 2008. Before starting the atmospheric re-entry, the cruise stage that brought the lander was jettisoned. The spacecraft entered the atmosphere and the heatshield initially slowed the craft. After about 3 minutes the parachute was deployed, followed by the ejection of the heatshield 15 seconds later, the deployment of landing legs and the radar activation. At 1 km altitude the parachute was released and a powered descent with soft-landing was performed by using a pulsed propulsion system with 8 thrusters. Arrived on the surface, solar panels were deployed and the scientific activities started. Figure 2.4 visualize the Phoenix lander during powered descent.

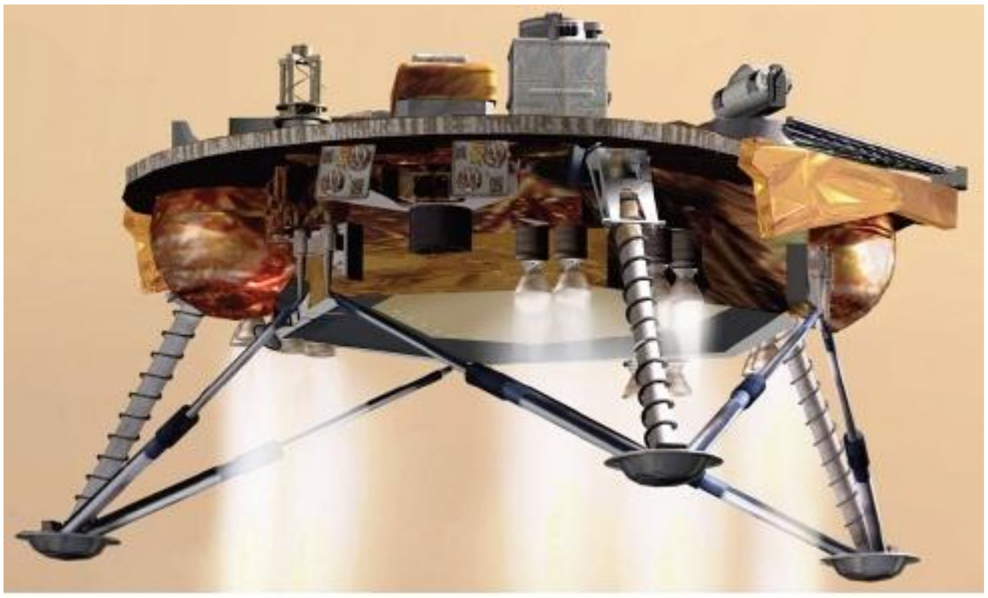


Figure 2.4. Phoenix powered descent

2.2 Entry, Descent and Landing sequence

This section presents configuration and the sequence of events that are considered in developing the EDL tool.
The EDL sequence of events starts with assigned conditions coming from the orbit. So the RV enters the atmosphere at velocity of several km/s. The vehicle has a conical fore-body and a truncated bi-conical aft body [**14**], following Viking's heritage. This configuration allows to increase the drag coefficient and to contain the heat that reaches its peak during the hypersonic flight, due to the high kinetic energy to dissipate. The parameter that characterizes the hypersonic motion is the ballistic coefficient β:

$$\beta = \frac{m}{C_d A} \tag{2.1}$$

where:
m: is the mass of the capsule
C_d: is the drag coefficient
A: is the reference surface

A low ballistic coefficient will achieve lower peak heat rate and peak deceleration values by decelerating at higher altitude in the atmosphere. To reduce ballistic coefficient, the largest aeroshell diameters are required, limited by the launcher capability. The only capsule is not enough to safely land, and so it is necessary a parachute to reduce velocity. Parachute is designed to operate in a supersonic condition and its shape is that of a disk-gap-band; the parachute is deployed at a fixed Mach number, which typical values are around Mach 2 (depends on the mission) and slows the capsule till subsonic velocity. Then, at a fixed altitude from the surface, typical 1 Km, the heatshield, the backshell and the parachute are released. The vehicle uses a powered descent to safely land on the planet surface. Figure 2.5 presents the different configurations in the EDL sequence considered in developing the tool.

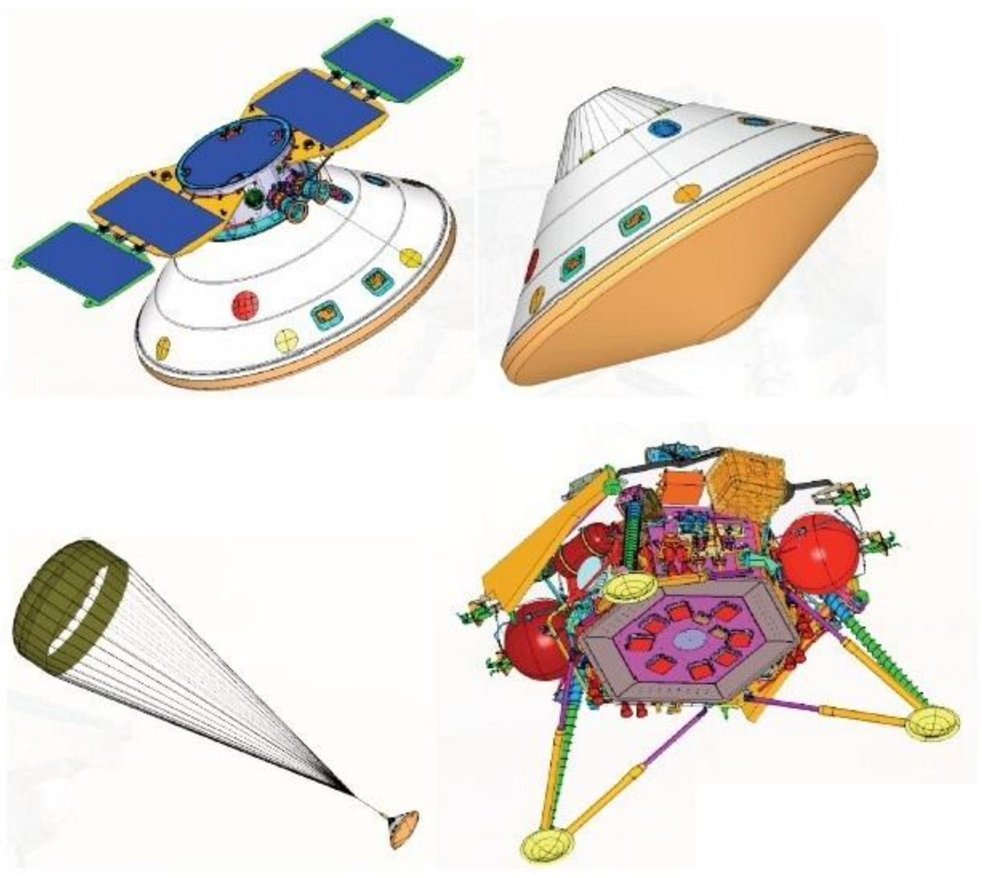


Figure 2.5. Different configuration in the EDL sequence
From the top-left: pre-entry, entry capsule, parachute deployment, powered terminal descent

Chapter 3

Mathematical models

A mathematical model is an abstract model that uses mathematical language to describe the behavior of a system. Mathematical models are used particularly in the natural sciences and engineering disciplines, but also in the social sciences. Using a model it is possible to have a representation of what happens in the real environment and analyze all variables and parameters that characterize a system. So, a thermal model of a spacecraft, for example, allows to identify the variation of temperature and its gradient in different parts and to verify the correct life temperature of the components and the need of a thermal control system. Focusing on this thesis, the main task is to identify variables and parameters that characterize the dynamic of an EDL system, not only the velocity and the altitude from the surface of a planet, but also the angles that defines the attitude of the vehicle. Describing the re-entry motion it is not a simple task: many variables characterize its dynamic and modeling all of them is very onerous and in most cases not necessary. So, simplification hypotheses were made in order to simplify the model, guided by the aim of the project.
In this chapter the mathematical model of the EDL system is presented, describing all assumptions and hypotheses made in order to correctly describe the real behavior of the system. First, reference systems are presented; later, the motion of the vehicle is described through a system of Ordinary Differential Equation (ODE). Force and moments that act on the RV are defined by creating models that describe the environment and the aerodynamics of the vehicle.

3.1 Reference frames

The motion of a re-entry vehicle can be easily described if particular reference systems are chosen to represent variables. For example, it is easier to describe the aerodynamic forces relative to the velocity direction, whereas the gravity force is better described if referred to the planet radius direction. Once the reference systems are defined, a set of rotation matrices occurs to perform a coordinate transformation of a vector, to correctly define the set of ODE. The next subsections describe the reference systems used and define the directional cosine matrices for axes transformation. The reference systems have been defined by using a rectangular coordinates system which follows the right hand rule.

3.1.1 Inertial Reference Frame

This reference system is fixed with the planet with the origin on the center of the planet and the X-Y plane on the equatorial plane. The **Z**-axis is directed towards the North Pole. The particular direction of the **X**-axis is different for each planet. For example, for the Earth, the **X**-axis points to the Greenwich meridian at time t=0. For Mars, the International Astronomical Union (IAU) fixes the reference direction along a vector defined by the intersection between the Mars Mean Equator and the Earth Mean Equator of Epoch J2000.
This system is identified by the tag "**I-frame**" and a vector in this frame is denoted through its components as $[X^I Y^I Z^I]$.

3.1.2 Local Reference Frame

This frame refers to the local horizontal considering a plane tangent to the planet surface. The normal to the plane points toward the center of the planet and define the **Z** axis direction, while the **X** axis is directed toward the local north.
This frame can be obtained from the I-frame by three rotations:

- A first rotation of the longitude λ around the 3 axis.
- A second rotation of $\left(\frac{3}{2}\pi - \varphi\right)$ around the 2 axis. This rotation is defined by the latitude φ and by a constant that allows to define the third axis (**Z**) pointing down towards the centre of the planet. In this way it is defined a NED (North East Down) system, characterized by the **X**-axis with the local north direction, the **Y**-axis follows the local East direction and the **Z**-axis follows the right hand rule.
- A third rotation is performed by rotating of $\left(\frac{\pi}{2} - \psi\right)$ around the 3 axis. Ψ is defined as the angle between the East and the direction of the velocity vector. With this last rotation, it is possible to describe a motion on the velocity plane (X-Z plane).

This system is represented in Figure 3.1. It is identified by the tag "**ℓ-frame"** and a vector in this frame is denoted through its components as $[x^l y^l z^l]$.

3.1.3 Aerodynamic (Wind) Reference Frame

To better describe the aerodynamic forces acting on the vehicle, it is set an aerodynamic frame (Figure 3.2), fixed on the mass centre of the vehicle and with the **X-**axis collinear with the airspeed vector and the **Y**-axis parallel to that one of the l-frame.
This frame is obtained by rotating the Local Frame of the FPA (Flight Path Angle) γ around the **Y**-Axis. The FPA is defined as the angle between the **X**-axis in the aerodynamic frame and the **X**-axis in the local frame.

This system is identified by the tag “**v-frame”** and a vector in this frame is denoted through its components as $[x^v y^v z^v]$.

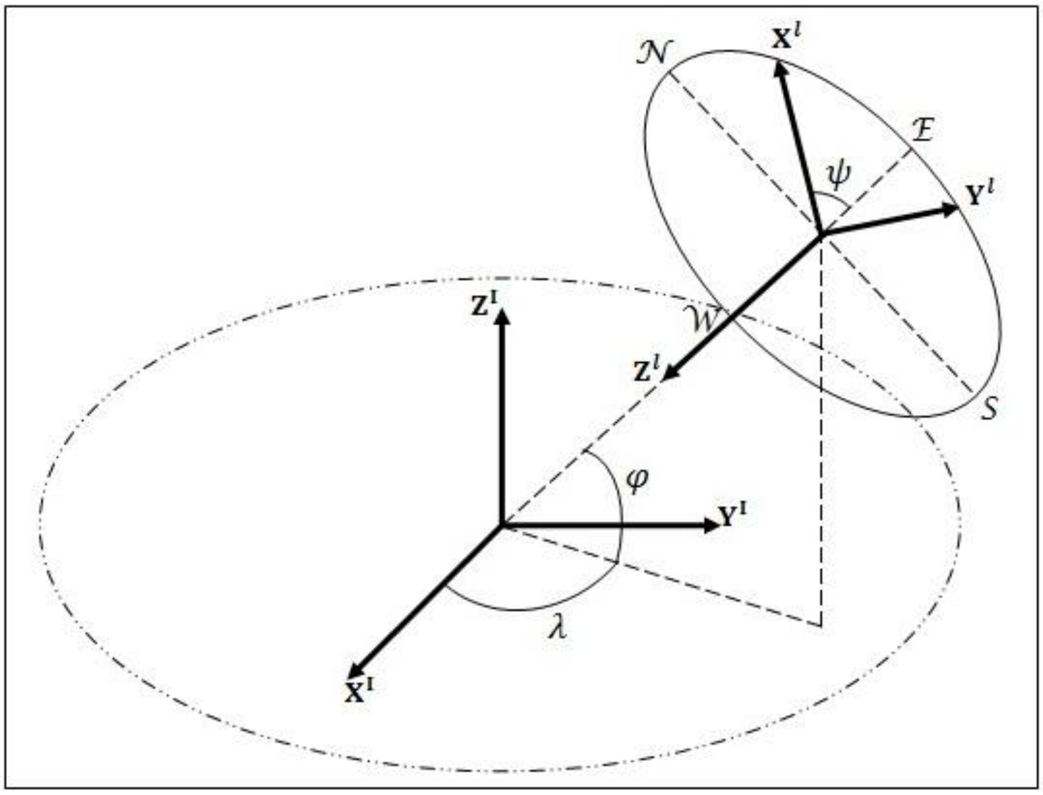


Figure 3.1. Local Reference Frame

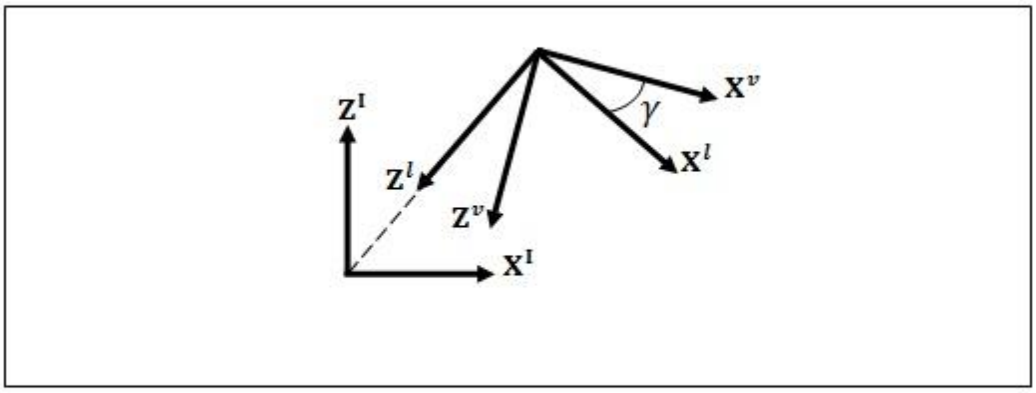


Figure 3.2. Aerodynamic Reference Frame

3.1.4 Body Reference Frame

This reference frame is necessary in order to define the attitude of the re-entry vehicle. This system is fixed on the vehicle and its axes are directed as the inertia principal axes. This reference system can be obtained in three ways, depending on the available information. All the methods presented allow to construct the same rotation matrix to perform rotation from the I-frame to the body one.

- Using attack and sideslip angle (if given).
  The attack angle α is defined in the velocity plane as the angle from the **X**-axis in the v-frame and the **X**-axis in the body frame. The sideslip angle β is defined in a similar way, considering the plane outside the velocity vector.
  Once defined the attack and sideslip angles, the Body Reference Frame is obtained from the v-frame by a first rotation of α around the 2nd axis and a second rotation of β around the 3rd axis.
- Using quaternions:
  The orientation of the body axes respect to the Inertial ones can be described by using quaternions. This kind of parameterization allows to avoid angles singularity problems [**15**]. As will be described in the next section, quaternions are considered as state variables in the motion equation. Thus their values are known at each time and the attitude matrix which performs rotation from the I-frame to b-frame can be constructed.
- Using an Euler's angle parameterization:
  This kind of parameterization allows to pass from the I-frame to the body one by making 3 successive rotations, following the scheme 123 (Figure 3.3):
  - 1st rotation of an angle χ around the 3 axis ($\mathbf{Z^I}$).
  - 2nd rotation of an angle ξ around the 2 axis (**y'**).
  - 3rd rotation of an angle η around the 1 axis (**x''**).

This system is identified by the tag "**b-frame"** and a vector in this frame is denoted through its components as $[x^b y^b z^b]$.

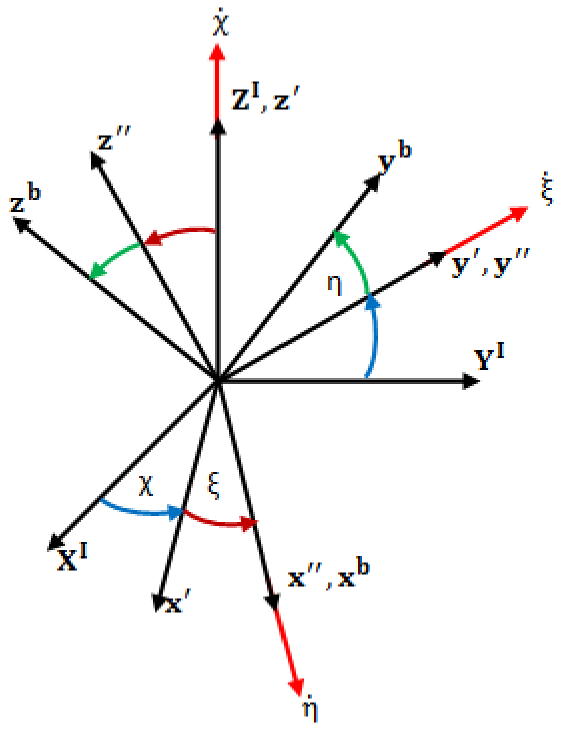


Figure 3.3. Euler angle 123 sequence of rotation

All reference systems are represented in Figure 3.4. The directional cosine matrices (DCM) to pass from a reference frame to another are reported in Appendix A, considering the notation C_a^b to describe the matrix to pass from the “a-frame” to the “b-frame”.

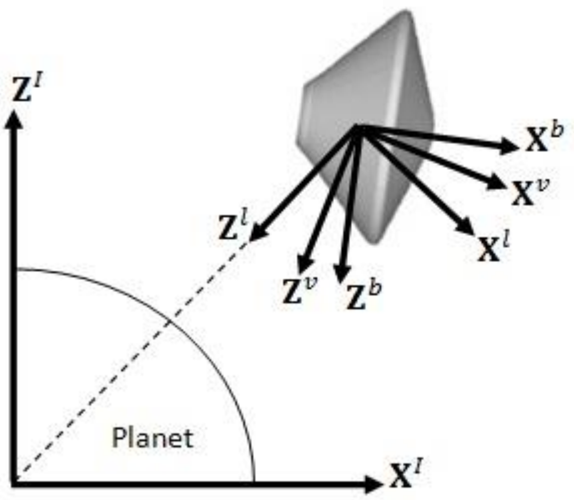


Figure 3.4. Reference systems

3.2 Dynamic models

As a first step to build the simulation tool, a planar motion on the velocity plane is considered to describe the re-entry vehicle trajectory [**16**].
Dynamics of re-entry is defined by the force equation, that defines the mass centre trajectory, and the moment equation, that deal with the attitude of the vehicle (3.1):

$$\begin{cases} m\dfrac{d\mathbf{V}^{\mathbf{I}}}{dt} = \mathbf{F}^{\mathbf{I}} \\ \dfrac{d\mathbf{H}^{\mathbf{I}}}{dt} = \mathbf{M}^{\mathbf{I}} \end{cases} \tag{3.1}$$

where:
m is the mass of the vehicle.
V is the velocity vector.
F is the vector of forces applied to the vehicle.
H is the angular momentum.
M is the vector of moment acting on the vehicle.
The superscript refers to the reference system used to represent vector, as presented in section 3.1.

Once analyzed a typical mission profile for an EDL system in an atmospheric environment, the landing trajectory has been divided into three phases. The first one analyzes the motion of a capsule from atmosphere entrance to the parachute opening. Then, the second phase starts and it is characterized by a multi-body dynamics of the capsule and parachute, linked by a riser. Once reached a defined altitude, the parachute, the backshell and the heatshield are released; the configuration changes, and the motion is described by the lander trajectory till the surface of the planet. Each of these phases is characterized by some peculiarity and hypotheses, described in the next subsections. Common to each phase are the following considerations:

- A rigid body model has been considered both for the capsule and the parachute.
- The rotation of the planet has been neglected in a first approximation to simplify the dynamics. This is a good hypothesis because of the times of flight during reentry are small (in the order of six minutes).
- Equations of mass center are solved in the v-frame, whereas the moment equations consider a body frame.

3.2.1 Phase 1

The motion refers to a capsule that enters the atmosphere. The capsule has an axial symmetric shape, and it is subjected by aerodynamic (**F**$_L$, **F**$_D$), gravitational (**F**$_g$) and control forces (**T**$_C$), and moments as visible in Figure 3.5.

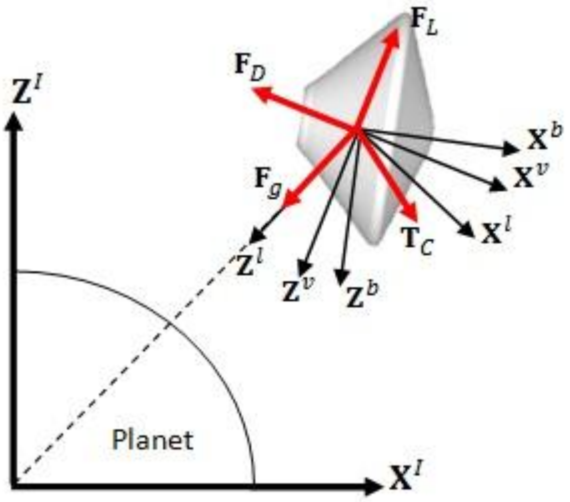


Figure 3.5. Forces acting on the capsule

Moments on the capsule are due to the aerodynamic and the control forces. Control forces and moments have been added in the equations of motion as parameters currently set to zero. This choice considers future developments of the tool, but in this thesis a control law for the vehicle attitude is not set. The EDL tool is built to easily be updated by adding new models. If in future a control law will be developed, equations are already written in the correct way.

Using the system (3.1), the trajectory of the vehicle during re-entry, can be computed by solving this system of equations:

$$\dot{h} = V \sin\gamma \tag{3.2}$$

$$\dot{\lambda} = \frac{V\cos\gamma}{(R_p + h)} \frac{\cos\psi}{\cos\varphi} \tag{3.3}$$

$$\dot{\varphi} = \frac{V\cos\gamma}{(R_p + h)} \sin\psi \tag{3.4}$$

$$\dot{V} = \frac{Fa_x}{m} - g\sin\gamma - \frac{Tc_x}{m} \tag{3.5}$$

$$\dot{\gamma} = \frac{V\cos\gamma}{(R_p + h)} - \frac{Fa_z}{mV} - \frac{g\cos\gamma}{V} - \frac{Tc_z}{mV} \tag{3.6}$$

$$\dot{\psi} = -\frac{V\cos\gamma}{(R_p + h)}\cos\psi\tan\varphi - \frac{Tc_y}{mV} \tag{3.7}$$

$$\dot{q}_0 = 0.5(-pq_1 - qq_2 - rq_3) \tag{3.8}$$
$$\dot{q}_1 = 0.5(+pq_0 + rq_2 - qq_3) \tag{3.9}$$
$$\dot{q}_2 = 0.5(+qq_0 - rq_1 + pq_3) \tag{3.10}$$
$$\dot{q}_3 = 0.5(+rq_0 + qq_1 - pq_2) \tag{3.11}$$

$$\dot{p} = \frac{1}{I_x}\left[\left(I_y - I_z\right)qr + Ma_x + Mc_x\right] \tag{3.12}$$
$$\dot{q} = \frac{1}{I_y}\left[(I_z - I_x)pr + Ma_y + Mc_y\right] \tag{3.13}$$
$$\dot{r} = \frac{1}{I_z}\left[\left(I_x - I_y\right)pq + Ma_z + Mc_z\right] \tag{3.14}$$

Groups of equations describe:
- Centre of mass kinematics. Eq. from (3.2) to (3.4).
- Centre of mass dynamics. Eq. from (3.5) to (3.7).
- Attitude kinematics. Eq. from (3.8) to (3.11).
- Attitude dynamics. Eq. from (3.12) to (3.14).

The state variables used in equations (3.2)-(3.14) to describe the motion have been reported in the previous section; for a clear description they are also listed below with their measure units used in the computation:
- h [m]: is the altitude of the vehicle, computed from the mean equator radius of the planet.
- V [m/s]: is the velocity of the vehicle.
- γ [rad]: is the flight path angle, defined as the angle between the $\mathbf{X}^{v}$ and the $\mathbf{X}^{l}$ axis.
- λ [rad]: is the longitude of the vehicle.
- φ [rad]: is the latitude of the vehicle.
- ψ [rad]: is the angle between east and the vector **V**.
- p, q, r [rad/s]: are the components of angular rate vector in the b-frame.
- q_0, q_1, q_2, q_3 [-]: are the quaternions, used to reconstruct the attitude matrix.

Other parameters in the previous equations are:
- R_p [m]: is the mean radius of the planet, fixed to $3.389 \cdot 10^6$ m for Mars.
- m [Kg]: is the mass of the vehicle.
- g [m/s^2]: is the gravity acceleration.
- I_x, I_y, I_z [Kg·m^2]: are the moments of inertia.

Forces and moments applied on the vehicle are listed by their components (subscripts x, y, and z):

- Fa [N]: are the aerodynamic forces.
- Ma [N·m]: are the aerodynamic moments.
- Tc [N]: are the control forces.
- Mc [N·m]: are the control moment.

3.2.2 Phase 2

This phase is characterized by a multi body dynamics, deal with the presence of the capsule and the parachute. The equations of motion consider capsule and parachute as single objects, and a constraint force has been added to take into account the connection between the two bodies. So, the system of equations is the same of the phase 1, but characterized by a double number of states, considering both the state variables for the capsule and the parachute. In addition, components of the vector representing the relative distance between capsule and parachute have been added as state variables. Compared to the phase 1, the equations of motion are also characterized by:

- The absence of the control forces and moments both on the capsule and the parachute.
- The presence of a constraint force acting on the capsule and the parachute.

Parachute modeling

Parachute has been modeled as a rigid body, and it is supposed to be already deployed when 2nd phase starts. In [**17**] it is presented a mathematical model to describe the physical, aerodynamic, and mass properties of a generic parachute.
The parachute model uses a body coordinate system, where the $\mathbf{X^b}$ axis is aligned with the bridle; the $\mathbf{Y^b}$ and $\mathbf{Z^b}$ axis of the parachute are in the symmetry plane of the canopy (Figure 3.6).

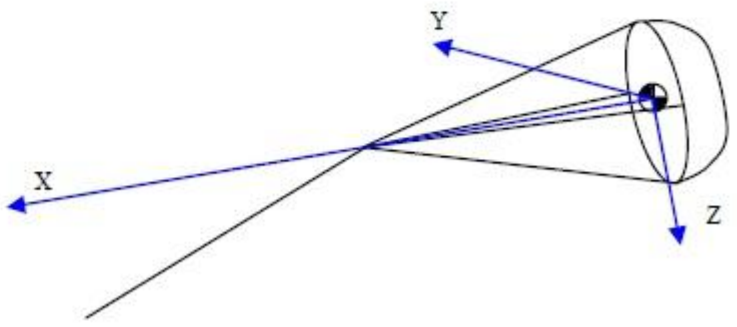


Figure 3.6. Parachute body coordinate system

Considering aerodynamics, the drag coefficient (C_d) is modeled by a first degree approximation (3.15), whereas the lift coefficient is modeled as a perturbation, considering small values as a function of incidence. Moment pitch coefficient is described as a function of attack angle by using a linear model.

$$C_d = C_{d0} + C_{d1}\alpha^2 \tag{3.15}$$

Where α is the angle of attack.
The mass properties of the parachute may either specified directly or estimated by using the radii of gyration method, that calculate them basing on the overall height, span and bridle.

Constraint force
To correctly describe the motion of the capsule and the parachute, a constraint force has to be added to the equations of motion to consider the interaction between the two bodies.
The riser attachment points on the vehicle and the parachute are assumed to rotate freely and therefore no torques are transmitted.
With this hypothesis, two models have been used to represent the constraint force; their equivalence has been successively verified (see Chapter 5).
A first method considers the constraint force by modeling the riser as a **mass-less spring damper** (Figure 3.7).

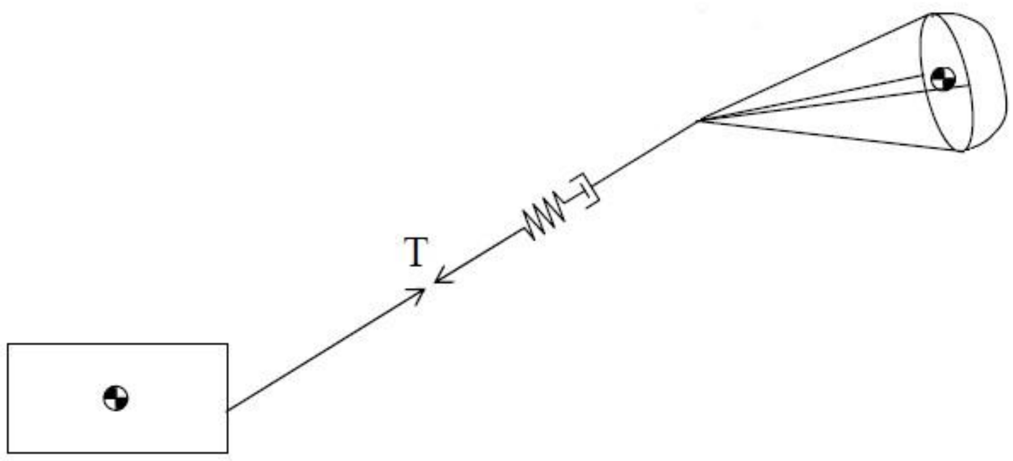


Figure 3.7. Riser line tension

The force modulus T on the riser line is computed based on the spring constants and damping ratio [**18**].

$$T = k_1\Delta L + k_2\Delta L^2 + c\frac{d\Delta L}{dt} \tag{3.16}$$

Where:

- k_1: is the linear spring constant
- k_2: is the second order spring constant
- c : is the damping ratio
- ΔL : is the change in the riser length

The second method used to calculate the modulus of the constraint force is by using directly the **constraint equation**. Let's call **d** the vector that links the capsule and the parachute mass centers in the I-frame (Figure 3.8):

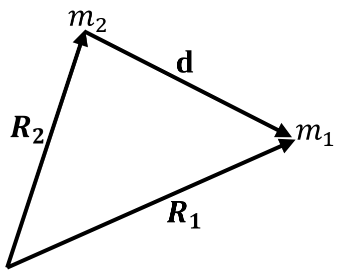


Figure 3.8. Distance vector between capsule (1) and parachute (2)

Then, the constraint equation can be written as:

$$|\mathbf{d}| = l_p \tag{3.17}$$

Where l_p is the length of the riser that connects the parachute to the capsule. Deriving the (3.17) twice respect to the time, the constraint equation becomes:

$$\ddot{\mathbf{d}} \cdot \mathbf{d} + \dot{\mathbf{d}} \cdot \dot{\mathbf{d}} = 0 \tag{3.18}$$

Vector **d** and its derivates are expressed as:

$$\mathbf{d} = \mathbf{R_1} - \mathbf{R_2} \tag{3.19}$$

$$\dot{\mathbf{d}} = \dot{\mathbf{R}}_\mathbf{1} - \dot{\mathbf{R}}_\mathbf{2} \tag{3.20}$$

$$\ddot{\mathbf{d}} = \ddot{\mathbf{R}}_\mathbf{1} - \ddot{\mathbf{R}}_\mathbf{2} \tag{3.21}$$

Vectors $\mathbf{R_1}, \dot{\mathbf{R}}_\mathbf{1}, \mathbf{R_2}, \dot{\mathbf{R}}_\mathbf{2}$ are known at each time because they are state variables. The second derivatives with respect to the time are obtained by the motion force equations for the capsule and the parachute.

$$m_1\ddot{\mathbf{R}}_1 = \mathbf{F}_1 - T\hat{\mathbf{d}} \tag{3.22}$$

$$m_2\ddot{\mathbf{R}}_2 = \mathbf{F}_2 + T\hat{\mathbf{d}} \tag{3.23}$$

Where $\mathbf{F_1}$ and $\mathbf{F_2}$ are the resultant vectors of the forces acting on the capsule and the parachute respectively. T is the constraint force modulus and $\hat{\mathbf{d}}$ is the versor representing the direction of the riser line in the I-frame.
Using equations from (3.18) to (3.23), the modulus of the constraint force can be obtained as:

$$T = \frac{\left(\frac{\mathbf{F}_1}{m_1} - \frac{\mathbf{F}_2}{m_2}\right) \cdot \mathbf{d} + \dot{\mathbf{d}} \cdot \dot{\mathbf{d}}}{\left(\frac{1}{m_1} + \frac{1}{m_2}\right) l_p} \tag{3.24}$$

Both using a spring damper method or a constraint equations, the constraint force vector is calculated in the I-frame as:

$$\mathbf{F^I} = T\hat{\mathbf{d}} \tag{3.25}$$

To be used in the equations of motion, the constraint force vector has to be described in the v-frame of capsule and parachute respectively; thus, it has to be rotated by using the directional cosine matrices referred to the two different bodies.

$$\mathbf{F_{12}} = \mathbf{C}_{\mathbf{I1}}^{v}\mathbf{F^I} \tag{3.26}$$

$$\mathbf{F_{21}} = \mathbf{C}_{\mathbf{I2}}^{v}\mathbf{F^I} \tag{3.27}$$

Where the notation $\mathbf{F}_{AB}$ means the force acting on the *A* body due to the *B* body.

3.2.3 Phase 3

The phase 3 starts after the release of the parachute, the backshell and the heatshield, when the vehicle is at about 1 km from the surface (this depends on the mission). The motion in this phase is solved in the same way already described for Phase 1.

3.3 Environment model

To develop a correct simulation of a re-entry vehicle motion, it is needed a description of aerodynamic and gravitational loads, which are defined by the vehicle’s configuration and by the environment characteristics. The last ones are presented and analyzed in this section.
An environment model must be able to represent atmosphere, considering the vertical variation of pressure, density and temperature; in addition to these, there are many additional derived quantities such as speed of sound, molecular mean free path, dynamic viscosity. At first glance it is useful to consider the gravitational field constant, thus eliminating each variation in gravitational loads due to planet shape. This is a good hypothesis because of the little duration of a landing and the less magnitude of the gravitational loads respect to the aerodynamic ones.

3.3.1 Gravitational force

Gravitational force is directed as the **Z**-axis in the local frame; thus, it can be expressed by the vector:

$$\boldsymbol{F_g^l} = \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} \tag{3.28}$$

Where m is the re-entry vehicle mass and g the gravity acceleration, 3.72 m/s^2 for Mars.

3.3.2 Density

There are several way to calculate density for an atmospheric environment, both numerical than experimental methods. The first ones differ by their complexity, from the simpler methods based on a single law, to the methods that, once divided the atmosphere in several layers, consider a particular law for each layer. Experimental methods, instead, base on the data surveys from previous missions; these data are collected in a database and density model are developed by fitting them [**19**]. A simulator must be capable to work with different level of approximation, based on particular requirements defined by the user. For these reasons, both a simple mathematical model and data fitting have been used in developing the EDL tool.
Data loaded in the simulator comes from a general circulation model of Mars [**20**]; this kind of model uses data from previous missions and combines them with a physical representation of the atmosphere phenomena, for example radiative heating, cloud microphysics, interaction and coupling with solid surface. Atmospheric data considered in this work are obtained from the European Mars Climate Database [**21**].

The mathematical model, instead, calculates the density by using an exponential model, whose parameters are defined for each planet. So, the density is:

$$\rho = \rho_0 e^{-\frac{h}{H}} \tag{3.29}$$

where:

- ρ_0: is the sea level density
- h: is the altitude
- H: is the scale height factor defined for each planets

Parameter for Venus, Earth and Mars are listed in Table 3.1:

Table 3.1. Planet parameters for density calculation

Planet	**Sea level density ρ_0 [Kg/m^3]**	**Scale height H [Km]**
Venus	6.48e-1	14.9
Earth	1.225	8.5
Mars	1.55e-2	10.6

3.3.3 Temperature

Temperature has been calculated by using a linear model (3.30), based on the surface temperature of the planet and by a thermal lapse rate, defined in Table 3.2**.** For the Earth the “1976 Standard Atmosphere Model” [**22**] has been used.

$$T = T_0 + L_h h \tag{3.30}$$

Table 3.2. Planet parameters for temperature calculation

Planet	**Average surface temperature T_0 [K]**	**Thermal lapse rate L_h [K/Km]**
Venus	700	-2.85
Earth	1976 Standard atmosphere Model	
Mars	210	-0.45

3.3.4 Pressure

Having density and temperature, the pressure P at each altitude has been obtained using Gas Perfect Law:

$$P = \frac{R_0}{M_o}\rho T \tag{3.31}$$

where:
- R_0: is the universal gas constant $8314\ \frac{J}{Kg\ Mol\ K}$.
- M_0: is the molecular weight (Venus: 43.45 [g/mol], Earth: 28.96 [g/mol], Mars: 43.34 [g/mol]).

3.3.5 Derived quantities

Once defined the state variables (ρ, P, T) for the atmosphere, it is useful to calculate some derived quantities that are important to identify some fundamentals parameters used in the definition of motion regime, such as the Knudsen number (3.32) and the Mach number (3.33):

$$\mathrm{Kn} = \frac{\lambda}{\mathrm{L_c}} \tag{3.32}$$

$$\mathrm{M} = \frac{\mathrm{V}}{\mathrm{c}} \tag{3.33}$$

where:
- λ: is the mean free path between atmospheric molecules.
- L_C: is the reference length of the re-entry body.
- V: is the velocity of the re-entry body.
- c: is the speed of sound.

For a perfect gas, the speed of sound c is given by:

$$\mathrm{c} = \sqrt{\gamma R T} \tag{3.34}$$

Where:
- γ: is the adiabatic polytrophic constant (Venus: 1.286, Earth: 1.44, Mars: 1.33).
- R: R_o/M_o is the gas constant.

The mean free path length is the average distance traveled by a gas particle before collision with another particle and it is given by:

$$\lambda = \frac{RT}{\sqrt{2}\pi\sigma^2 N_A P} \tag{3.35}$$

where:
- σ: is the effective collision particle diameter.
- N_A : is the Avogadro’s number.

3.4 Aerodynamic model

The behavior of an EDL system in an atmospheric environment is strictly connected with the aerodynamic characteristics of the body; aerodynamic loads are bigger than other forces. For this reason, a correct estimation of the aerodynamic coefficients and their uncertainties is critical. There are several methods to estimate aerodynamic characteristics of a re-entry body:

- Ground tests in wind tunnel, considering corrections due to the different composition for the atmosphere of a particular planet (e.g. for Mars the large amount of CO_2).
- A modern Computational Fluid Dynamics (CFD) analysis. Several tools show the computation of aerodynamic coefficients for different regimes of motion [**23**]. In the free molecular regime, collisions between individual molecules are so infrequent that they don't need to be modeled; this regime is characterized by using a direct simulation Monte Carlo (DSMC) method. The same method is used in the transitional regime, but considering molecules interaction as done in DAC tool (DSMC Analysis Code). The continuum regime is governed by Navier-Stokes equations, solved in the Langley Aerothermodynamics Upwind Relaxation Algorithm (LAURA).
- Using available data from previous missions.

These methods are used together to estimate aerodynamic characteristics of a re-entry body, to study future entry vehicles configurations.
This approach is the best solution, but it requires dedicate studies, an extensive numerical and experimental simulation campaign, a large amount of time and resources. For these reasons, this work uses available aerodynamic coefficient from previous missions. A complete aerodynamic database for a Mars entry capsule is not freely available, so information coming from different missions have been analyzed ([**23**] [**24**] [**25**] [**26**] [**27**] [**28**] [**29**] [**30**]). The more complete aerodynamic database results that of Viking missions. Viking capsule has been selected as reference to validate the code, using the aerodynamic coefficients to compute loads acting on the re-entry vehicle.

3.4.1 Aerodynamic forces

Once the aerodynamic coefficients are available, it is possible to calculate lift (L) and drag (D) forces, with the classical formulas:

$$L = \frac{1}{2}\rho V^2 S C_L \qquad (3.36)$$

$$D = \frac{1}{2}\rho V^2 S C_D \tag{3.37}$$

Where S is the reference length of the vehicle and C$_L$ and C$_D$ are respectively the aerodynamic lift and drag coefficients, coming from the database previous described. By their definitions, we identify drag as the force acting opposite to the velocity vector, while lift acts perpendicular to it. So, the aerodynamic force $\boldsymbol{F_a^v}$ vector can be easily represented in the v-frame as:

$$\mathbf{F}_\mathbf{a}^\mathbf{v} = \begin{Bmatrix} -D \\ 0 \\ -L \end{Bmatrix} \tag{3.38}$$

The aerodynamic torque is represented referred to body axis, considering the relative aerodynamic coefficients (C$_m$) and reference length (d$_{ref}$). The motion analyzed is on the velocity plane; there are not forces acting out of the plane, and so the only component of the aerodynamic moment is that one along the **y**-axis.

$$\mathbf{M} = \frac{1}{2}\rho V^2 S \begin{Bmatrix} 0 \\ d_{\mathrm{ref}} C_m \\ 0 \end{Bmatrix} \tag{3.39}$$

During the motion, the capsule oscillates around the trim condition, if there are only aerodynamic and gravitational loads. A damping moment has been introduced, to simulate the air damping effect. The damping moment has been represented with a vector in the b-frame and added in the motion equations, in the same way of the aerodynamic moment.

Chapter 4

Software architecture

The software architecture of a system presents the elements that identify the system and the relationships among them. The main task of software architecture is to reduce problem complexity through abstraction and separation of concepts. Following this philosophy, it is possible to reduce a big problem into different smaller problems, to be solved more easily. The relationships among them are established by logical criteria of cause and effect. Once defined the connections, problems are solved step by step and linked, in order to solve the original one.
This method is not only used in computer programming to develop software, but it is a good way to proceed in all kind of problems. In computer programming, developing software architecture helps the designer to construct a robust tool and the user to easier understand it.
The software architecture developed for the EDL tool is presented in this chapter. EDL problem has been subdivided in many parts connected each other, that can be developed separately. In this way, a robust scheme is implemented, whose modules can be singularly updated.
The first section of this chapter presents the requirements to create a robust architecture, also referring to the integration with other existing tools of the whole simulator. Then, the architecture developed is presented, giving a general description of the single parts. After that, each part is analyzed, with a description of its functionality and peculiarity.

4.1 Requirements and constraints

EDL tool is a part of a bigger simulator, implemented to support engineers in mission design. So, the tool developed in this work must satisfy both its own requirements and those coming from the integration with the other tools. The simulator must be able to:

- Give a realistic description of what really happens.
- Describe different scenarios.
- Provide a virtual framework for the development of different levels of autonomy.
- Allow integration of modules and externally developed tools.
- Support design and configuration trade offs for different kind of missions.

- Validate the mission plans.

In order to achieve these goals, the philosophy followed in developing the tool is to construct it in a flexible and modular way. Each modulus is independent of the other ones. In this way, it is possible to update or change it without modify the whole simulator. With a modular scheme, it is also possible to accomplish and follow the evolution of the mission needs and system design. This was performed in computer programming by using the C++ language, an open source programming language characterized by an object-oriented programming. In this way, the software can be updated and modified without license problem and also the object oriented structure allows to easily define each modulus. The architecture scheme is then translated in a series of C++ classes described in the following sections.

4.2 Architecture for Entry Descent and Landing dynamic

The mathematical models that define the EDL system have been already described in the previous chapter. The goal of this chapter is to convert them in a computer program. In order to accomplish this task, each part of the EDL system has been considered as a single C++ class connected with the other by logical criteria. The highest level of software architecture is presented in Figure 4.1:

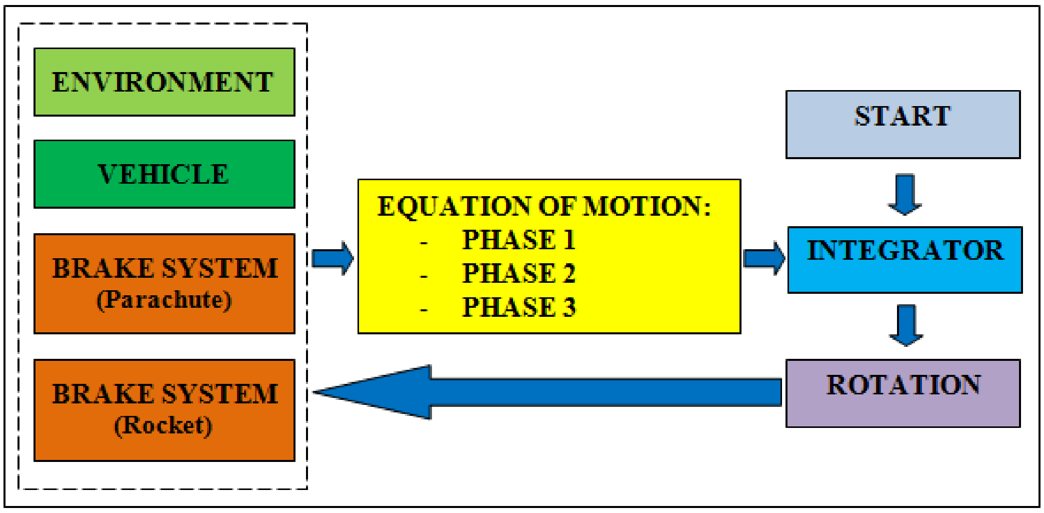


Figure 4.1. EDL tool architecture

Each block refers to a C++ class. Parameters of motion, such as geometric ones, forces and moments, are defined by classes that identify the elements involved in the re-entry dynamics. In particular, the class Environment defines the atmospheric properties at each altitude. The class Vehicle sets the geometric, aerodynamic and control characteristics of the re-entry vehicle. The Brake System is a class that sets the method used to reduce the RV velocity. The output of these classes enters the block “Equation of motion”, which defines the ODE system. The Integrator class performs the numerical integration of the dynamics, and it is initialized by the class Start, which sets the initial condition of the re-entry trajectory. Class Rotation sets the DCM matrices needed to pass from different reference frames. Each class is updated every step with the new conditions coming from integration.

4.3 Class definition

This section presents each single class. The order in which the classes are presented is the same used to build them in the software.

4.3.1 Start

This class sets the initial conditions needed to start the simulations. The initial conditions are passed to the simulator by using a text file. The user has to prepare this file that contains the value of the state variables relatives to the 1st phase (3.2.1); angle values in the file are expressed in degrees, because of a better physics vision. Quaternions are not listed in the file, but they are computed from the input data. This task is accomplished by the class Rotation Start, defined as a subclass of Start class, and which also sets the initial DCM matrices.

4.3.2 Environment

This class performs the computation of atmospheric properties at each altitude. The user has to set a number to identify the planet; planets are counted starting from the Sun (1-Mercury, 2-Venus, 3-Mars, …). In addition, the user can select the method to calculate atmospheric properties. As reported in section 3.3, it is possible to choose whether to use a mathematical model or a data fitting.
The Environment class is built by different functions, and it can be easily updated or modified. If in future a more accurate model for the atmosphere of a particular planet will be developed, it can be easily added to the class without changing the other functions. Changes in the atmospheric properties depend on the altitude of the RV. So, at the initial step, the class stores the parameters relative to the planet selected and in the successive steps it is simply updated by changing the altitude. This allows to reduce the amount of memory required, because data are overwritten on the previous ones. As output, the Environment class gives parameters for the system and the aerodynamic forces computation.

4.3.3 Vehicle

This is the main class of the EDL tool because it sets the geometrical and the inertial properties of the re-entry vehicle and computes the aerodynamic and control loads. All these functions are accomplished by different subclasses, visible in Figure 4.2:

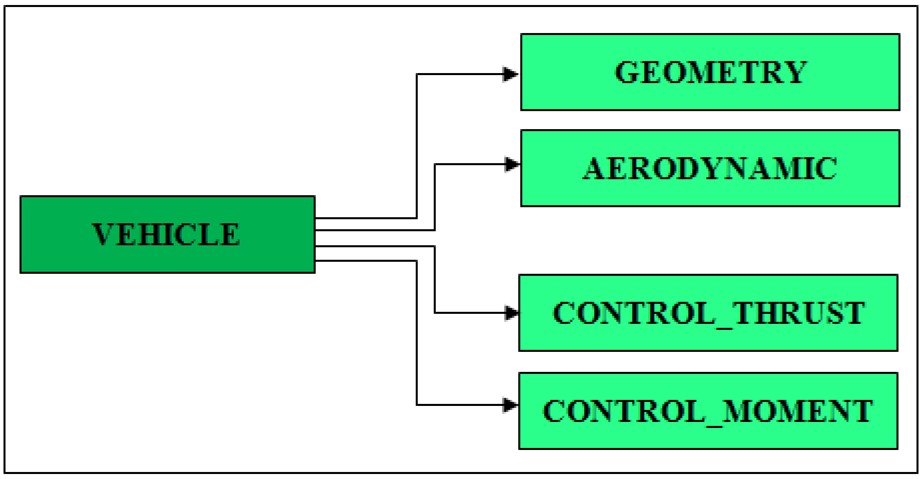


Figure 4.2. Class vehicle and its subclasses

Geometry class defines the configuration of the vehicle, whereas the aerodynamic class sets its properties relative to the interaction with the environment.
In section 3.2.1 it is explained that a control law has not been set in this work. However, control classes have been created for future upgrading of the tool. These classes will contain a mathematical model to compute a control moment (class Control_moment) or a control thrust (class Control_thrust), depending on the actuators that are used in the re-entry vehicle (defined by the user). For example, if retro-rockets are used, the control thrust class may contain a law to adjust the duration of rocket firing.

Geometry
With this class (Figure 4.3), the user is able to insert the re-entry vehicle characteristics, needed to set parameters for the dynamics. Data are inserted by using a text file, in the same way of the environment properties. Data required are the matrix of inertia, the mass, the reference surface and the reference length of the RV. This class is the same during the complete simulation, and changes in configuration during the re-entry trajectory are considered by updating the class. Output of this class sets some system parameters and gives input to the aerodynamic class.

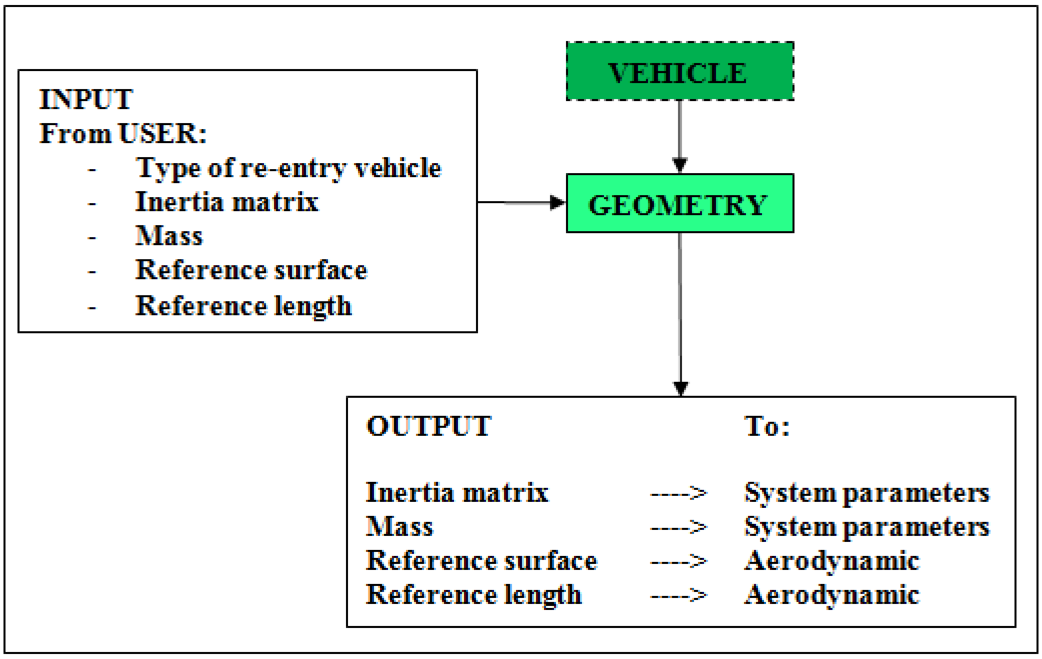

VEHICLE
INPUT
From USER:
- Type of re-entry vehicle
- Inertia matrix
- Mass
- Reference surface
- Reference length
GEOMETRY
OUTPUT
To:
Inertia matrix ----> System parameters
Mass ----> System parameters
Reference surface ----> Aerodynamic
Reference length ----> Aerodynamic

Figure 4.3. Geometry class

Aerodynamic

This class (Figure 4.4) is constructed by data coming from different sources and computes aerodynamic forces acting on the RV. In fact, the aerodynamic loads depend on the configuration, the environment, the motion characteristics and the orientation of the body axes with respect to the wind frame.

The core of this class is to compute the aerodynamic coefficients, coming from previous missions' data (section 3.4.1). These ones are available in scientific literature as function of motion characteristics defined by the dimensionless numbers of Knudsen, Mach and by aerodynamic angles (attack and sideslip). Aerodynamic coefficients are listed in a discrete way, thus it is necessary to interpolate them to have a complete aerodynamic description of the vehicle during the re-entry trajectory. The interpolation of aerodynamic coefficients is performed by using cubic splines. Once the aerodynamic coefficients are defined, the forces are computed by using formulas given in section 3.4.1. At each step the class is updated and aerodynamic forces and moments are given as output.

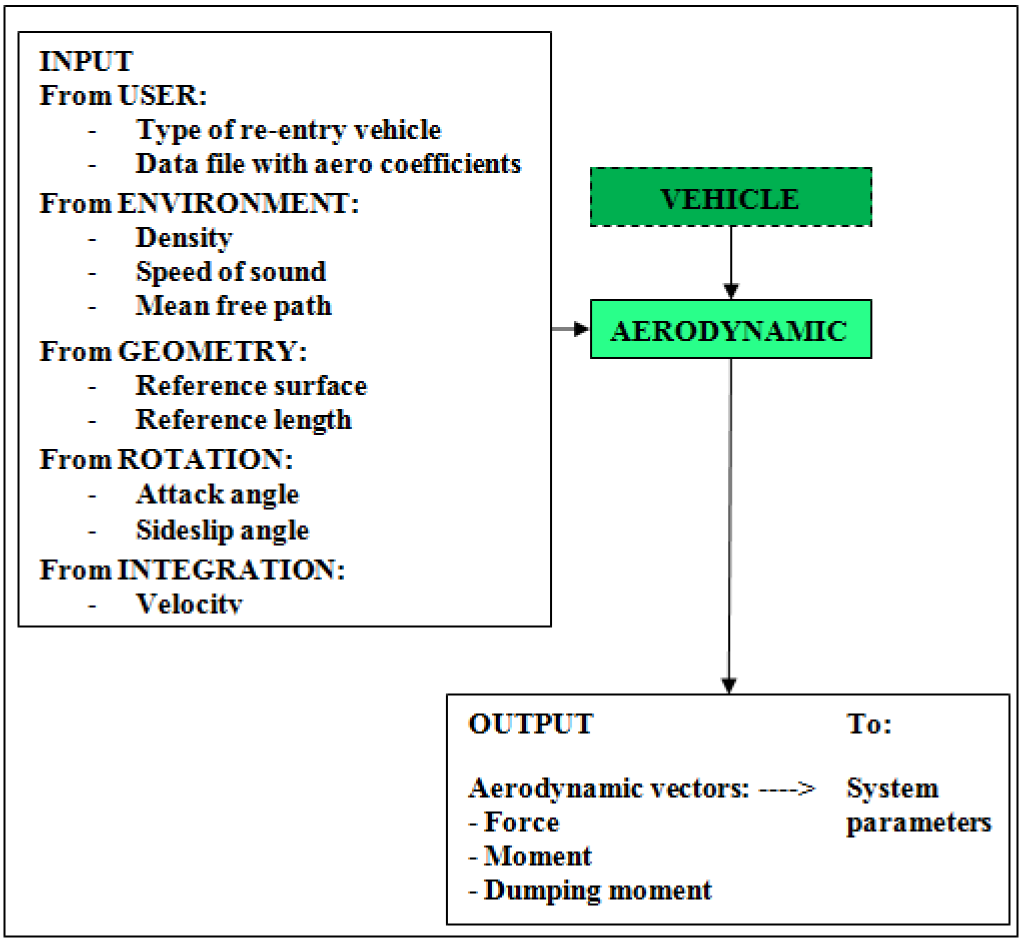


Figure 4.4. Aerodynamic class

4.3.4 Brake System

Several systems are used during EDL missions to accomplish this task ([**31**] [**32**]): parachute, retro-rockets, supersonic inflatable structures. The Brake Class defines which system is used to reduce the vehicle velocity during the re-entry trajectory (Figure 4.5). Considering the parachute, the scheme is the same used for the re-entry vehicle, constructing the geometry and the aerodynamic subclasses. Functions are the same used for the vehicle, except for the aerodynamic coefficients computation, in which a mathematical model has been used (section 3.2.2) instead of getting data from a file. The Brake Thrust Class, instead, sets the numbers of retro-rockets, their characteristics, and their orientation respect to body axes. The value of thrust to use in the system is computed by using these information and the ones that come from the control thrust class.

For a better expenditure of the memory, the brake class objects are created before computing the integration, even they are not necessary (e.g. the parachute is used in the 2nd phase). When they are used, classes are updated with correct values coming from the motion characteristic.

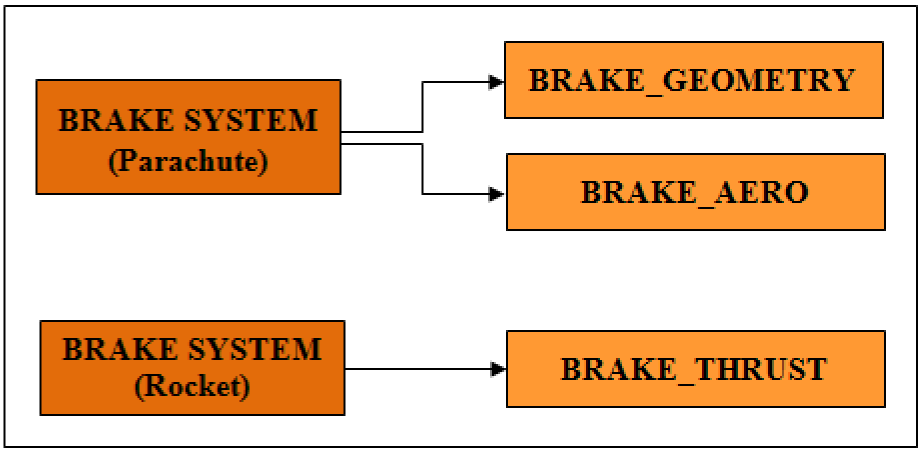


Figure 4.5. Brake classes and their subclasses

4.3.5 Rotation

The Rotation class takes the data coming from the motion integration as input and gives the DCM matrices at each step, computed in the same way described in section 3.1 and in Appendix A. In this way, the position of the re-entry vehicle and its attitude are easily described at each time.

4.3.6 Integrator

This class sets and applies the integration method. The user has to set the system dimensions and the type of integration scheme. The integration algorithm used comes from an external C++ library, the Gnu Scientific Library (GSL), and can perform the integration both with fixed or variable step. The GSL algorithm with a variable step sets the integration step to satisfy a fixed tolerance; this causes an extreme reduction of the step size that increases the time of the entire simulation. The GSL algorithm with a fixed step doesn’t have this problem, but the step size have to be chosen correctly. In the EDL tool a fixed step of 0.1 sec with a Runge Kutta 4th order method is used. This is the best choice to have very good results with acceptable integration times.

4.3.7 Equation of motion

This block contains the system ODE definitions, in the form developed in section 3.2.1; a block has been created for each phase. Parameters in the system are set by using the output coming from the classes previously described. For each phase (Figure 4.6, Figure 4.7), an array containing the state variables has been allocated and updated at each step of integration. State variables are the same for the 1st and the 3rd phase as previously reported (section 3.2). The 2nd phase block contains a double number of state variables, because of the presence of both the capsule and the parachute; in addition, three state variables are added considering the components of the vector of relative distance between the two bodies. 2nd phase block also contains a function that computes the constraint force between capsule and parachute: the user can select among the two methods considered for this computation (section 3.2.2).

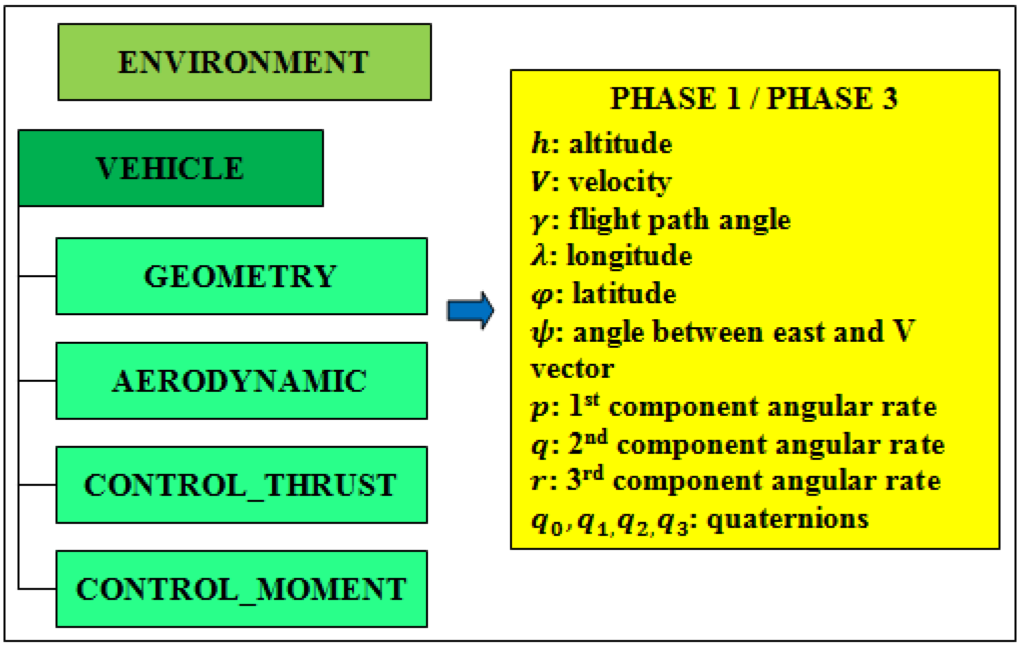

ENVIRONMENT
VEHICLE
GEOMETRY
AERODYNAMIC
CONTROL_THRUST
CONTROL_MOMENT
PHASE 1 / PHASE 3
h: altitude
V: velocity
γ: flight path angle
λ: longitude
φ: latitude
ψ: angle between east and V vector
p: 1st component angular rate
q: 2nd component angular rate
r: 3rd component angular rate
q_0, q_1, q_2, q_3: quaternions

Figure 4.6. Motion architecture for phase 1 and 3

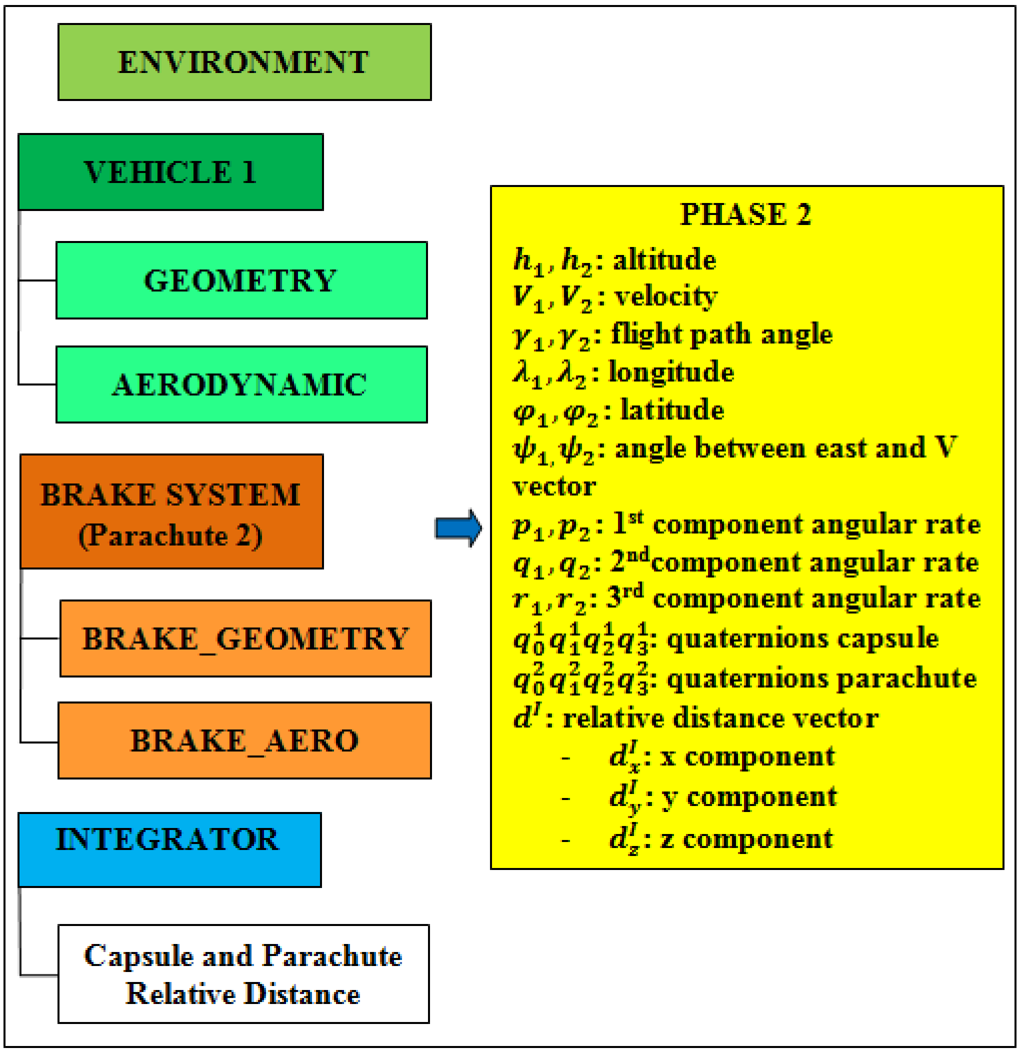

ENVIRONMENT
VEHICLE 1
GEOMETRY
AERODYNAMIC
BRAKE SYSTEM (Parachute 2)
BRAKE_GEOMETRY
BRAKE_AERO
INTEGRATOR
Capsule and Parachute Relative Distance
PHASE 2
h_1, h_2: altitude
V_1, V_2: velocity
γ_1, γ_2: flight path angle
λ_1, λ_2: longitude
φ_1, φ_2: latitude
ψ_1, ψ_2: angle between east and V vector
p_1, p_2: 1st component angular rate
q_1, q_2: 2nd component angular rate
r_1, r_2: 3rd component angular rate
$q_0^1 q_1^1 q_2^1 q_3^1$: quaternions capsule
$q_0^2 q_1^2 q_2^2 q_3^2$: quaternions parachute
d^I: relative distance vector
- d_x^I: x component
- d_y^I: y component
- d_z^I: z component

Figure 4.7. Motion architecture for phase 2

4.4 Main algorithm

A scheme of the main program that computes the re-entry trajectory, from the initial conditions to the landing on the planet's surface, is presented in Figure 4.8. The computation of the re-entry trajectory is inserted in a loop till the surface of the planet is reached (H_0). The selection of the mission phase is based on the Mach number (M) and the altitude (H) of the re-entry vehicle. In particular M_2 corresponds to the Mach number required to start 2nd phase; H_3 represents the altitude to release the parachute and to start the 3rd phase. The reference Mach number and altitude are set by the user.

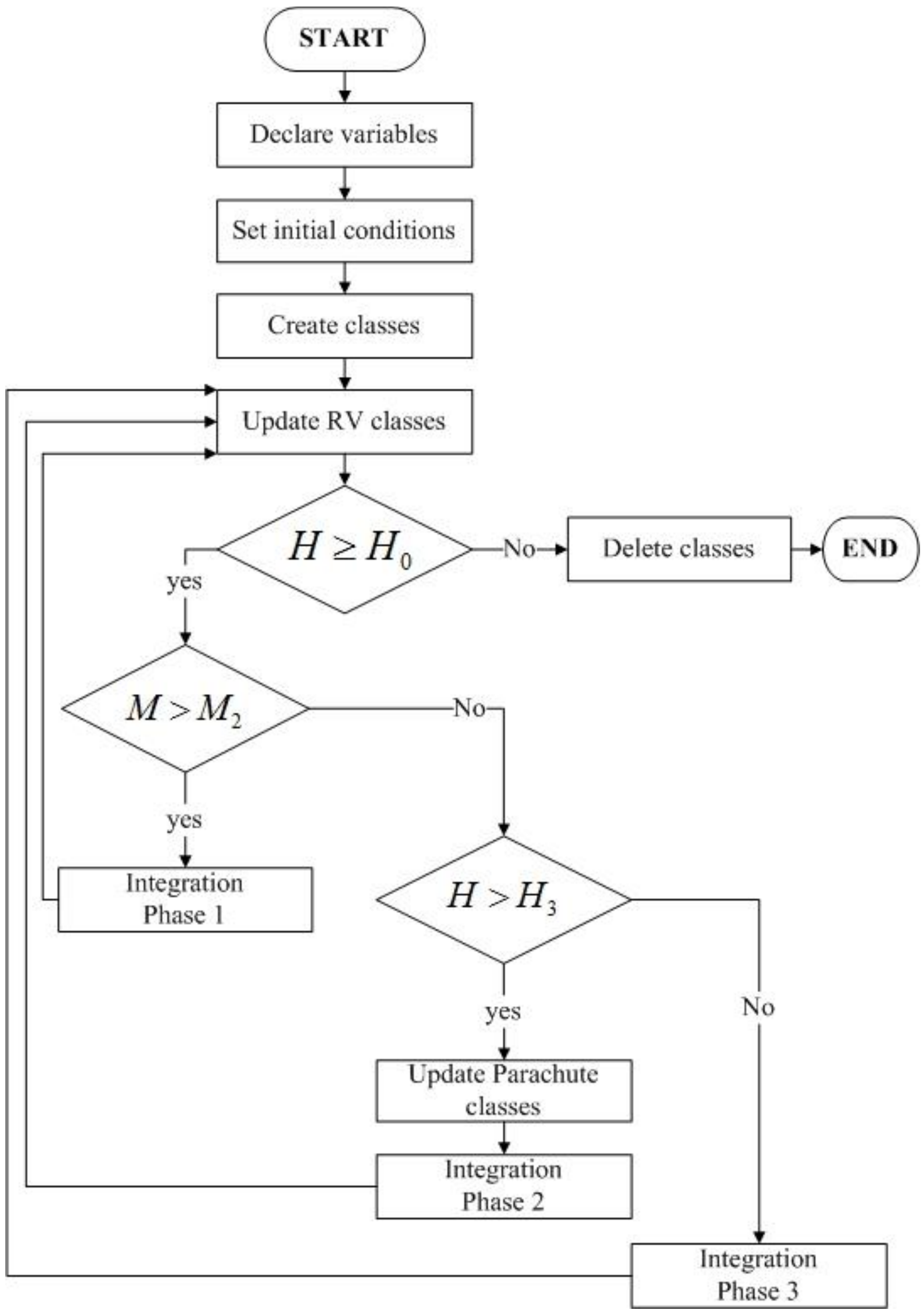


Figure 4.8. Main algorithm

Chapter 5

Software validation

Validation is the process that checks if the software meets the specifications and the purposes for that it was built for. During validation, the correct behavior and the results of software are checked. Considering the EDL tool developed in this work, the validation of the software is extremely important. A wrong description of the re-entry trajectory could result in mission and vehicle configuration design very different from the real one needed. This can cause a large waste of time and resources. Another element relates the tool itself that must work even changing its initial conditions. In fact, a simulator represents a real situation defined by the start conditions, set by the user.
Starting compiling a code, a test condition is chosen. Once the code has been developed, several tests are made by changing the initial conditions to verify the correct behavior of the program.
During the development of the EDL tool, each block has been verified and tested. The single blocks implemented are linked to build the whole program that simulates the re-entry trajectory. In this chapter, the validation of the re-entry trajectory is presented. To validate the EDL tool, results from scientific paper and Matlab® Simulink Aerospace Toolbox model are used. First, validation of each single phase is presented, and then results from a complete simulation are reported.

5.1 Phase 1

The first phase considers the motion of the vehicle up to the parachute deployment. The simulator is stopped at a Mach number lower to that of parachute deployment. In this way, phase 1 is validated in a wider range respect to its real application. Table from Table 5.1 to Table 5.3 report the input data used to validate the 1st phase:

Table 5.1. 1st Phase - Entry conditions

Parameter	**Value**
Altitude [m]	130000
Velocity [m/s]	7.470e3
Flight path angle [deg]	-18
Latitude [deg]	0
Longitude [deg]	0

Table 5.2. 1st Phase - Capsule geometric properties

Parameter	**Value**
I_{xx} [Kg·m^2]	2714.2
I_{yy} [Kg·m^2]	2714.2
I_{zz} [Kg·m^2]	1475.2
Mass [Kg]	500
Reference surface [m^2]	25
Reference length [m]	2.82
Ballistic coefficient [Kg/m^2]	13.5

Table 5.3. 1st Phase – Control conditions

Event	**Parameter**	**Value**
End 1st phase	Mach [-]	0.4

Results on the variation of the altitude and the modulus of velocity are reported in Figure 5.1:

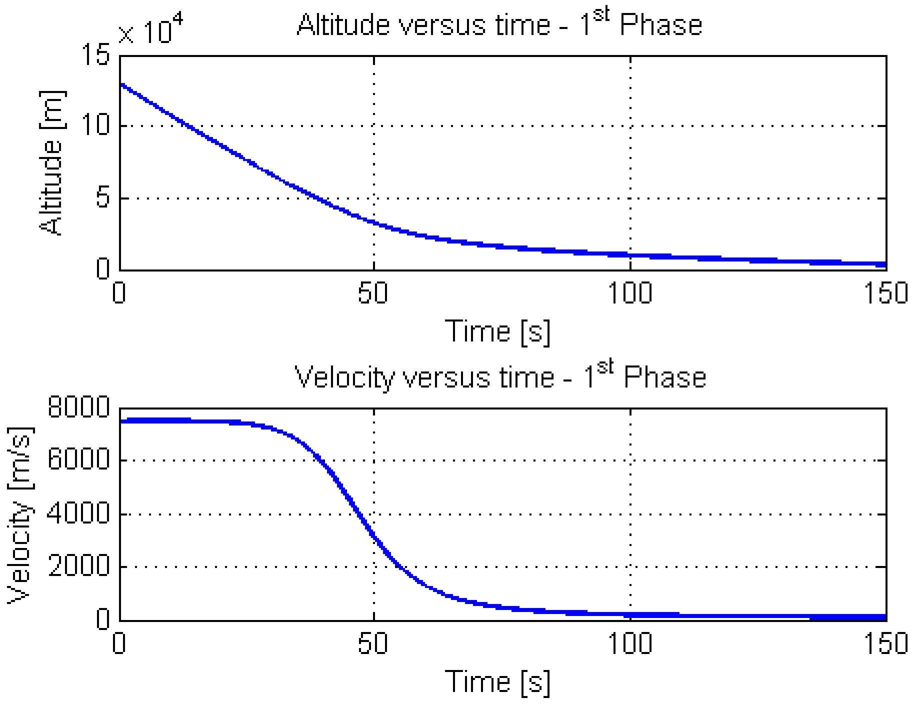


Figure 5.1. 1st Phase - Altitude and velocity

We can notice a rapid decrease of the altitude in the first phase of the re-entry. This is due to the low density of the atmosphere at high altitudes; this results in a low drag. This situation is clearly visible in the velocity representation: the speed profile is flat at the beginning, because the aerodynamic forces are lower than the gravitational ones. When aerodynamic forces (especially drag) become bigger, we can observe a rapid decrease of the speed. These results can be confirmed by comparing them to other simulations. A Simulink model has been created for the first phase and its results have been compared with the EDL tool written in C++ (Figure 5.2). A good agreement between them is achieved.

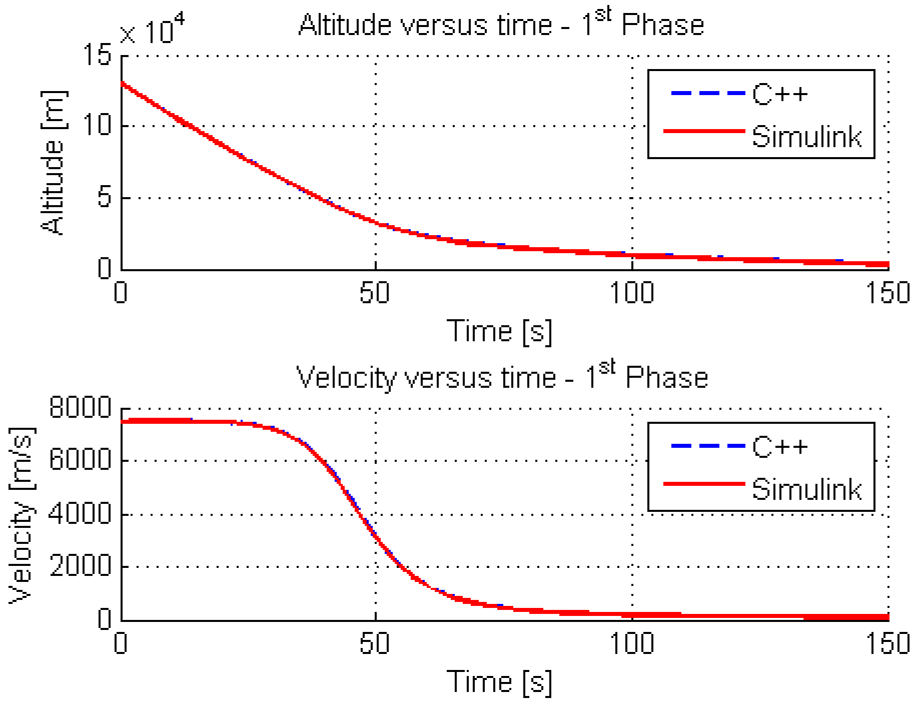


Figure 5.2. 1st Phase - C++ vs. Simulink

Another comparison has been made based on the results of a master thesis developed in “Politecnico di Milano” for an EDL control system [**33**]. Comparing them with those obtained in this work, we can notice similar altitude and velocity profiles (Figure 5.3).

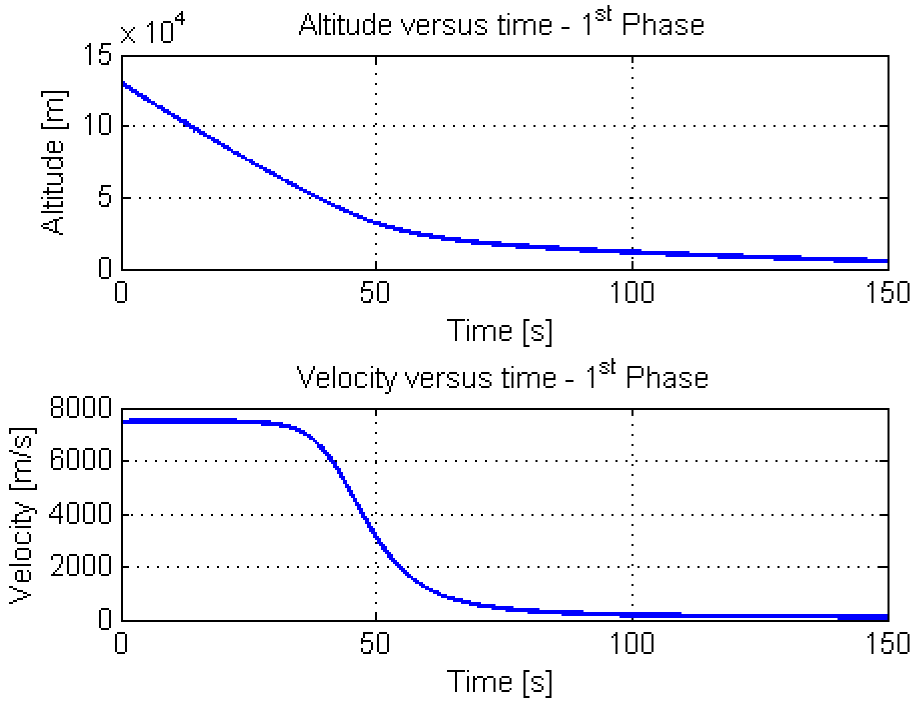


Figure 5.3. 1st Phase - Results from [**33**]

As already said in Section 3.4, a correct aerodynamic description is important to better compute the aerodynamic forces that influence the motion. Figure 5.4 gives a representation of the angle of attack and sideslip. It is known that a body subjected only to aerodynamic forces, tends to fly in a trim condition at a particular value of attack angle, called trim angle. This is verified in the figure, obtained considering the same capsule used by Ferraro [**33**], whose trim angle was about 11.5 degrees. The motion of the re-entry vehicle is considered planar (hypothesis made in section 3.2); so, as there are no forces acting outside the velocity plane, the sideslip angle keeps its initial value (that has been set to zero), as visible in Figure 5.4.

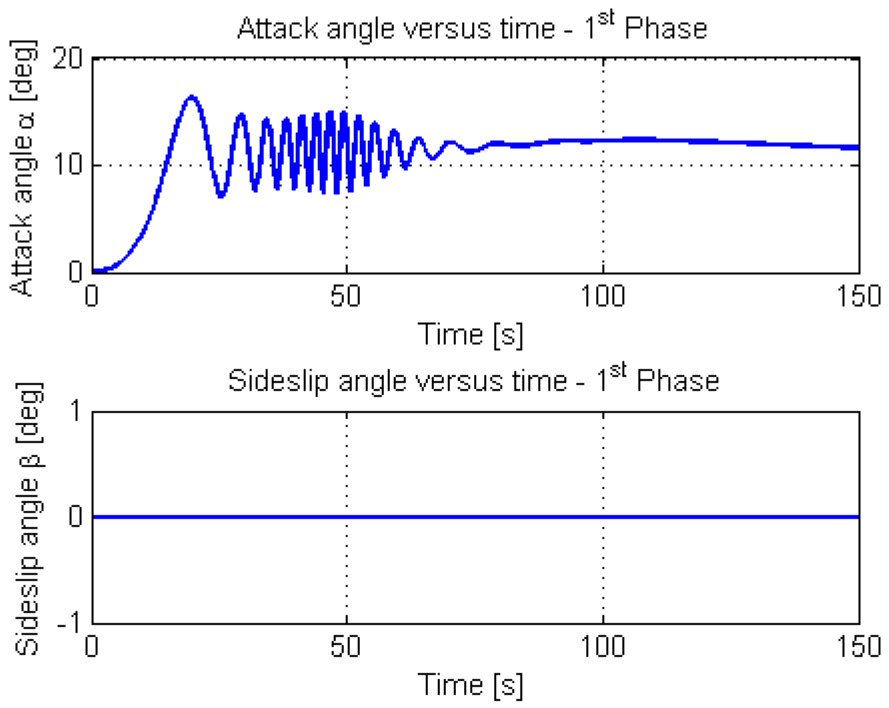


Figure 5.4. 1st Phase – Attack and sideslip angles

5.2 Phase 2

The second phase is characterized by the simultaneous presence of the capsule and the parachute. The equations of motion consider the capsule and the parachute as independent objects linked by a riser that transmits a constraint force, to be added in the system of ODE. Without this force, the equations of motion are the same of the 1st phase. For this reason a first approach to the phase 2 considers the equations of motion for the capsule and the parachute with the constraint force set to zero. In this way, the correct variations of state variables for each body (the capsule and the parachute) are verified. Once this is done, the second step introduces the constraint force, analyzing the multi-body motion.
Parameters used to validate the 2nd phase are the same of the previous one, adding the parachute properties (Table 5.4) and a condition to release it (Table 5.5).

Table 5.4. 2nd Phase - Parachute geometric properties

Parameter	**Value**
I_{xx} [Kg·m^2]	27142
I_{yy} [Kg·m^2]	27142
I_{zz} [Kg·m^2]	14752
Mass [Kg]	46.283
Reference surface [m^2]	204
Riser length [m]	21.7109

Table 5.5. 2nd Phase – Control conditions

Event	**Parameter**	**Value**
End 1st phase	Mach [-]	1.79
End 2nd phase	Altitude [m]	1500

The presence of the parachute adds a drag force to the re-entry vehicle, which brakes the system. This can be observed as a discontinuity and can be noticed a steeper decrease of the speed when the 1st phase end and the 2nd phase starts (red star in Figure 5.5).

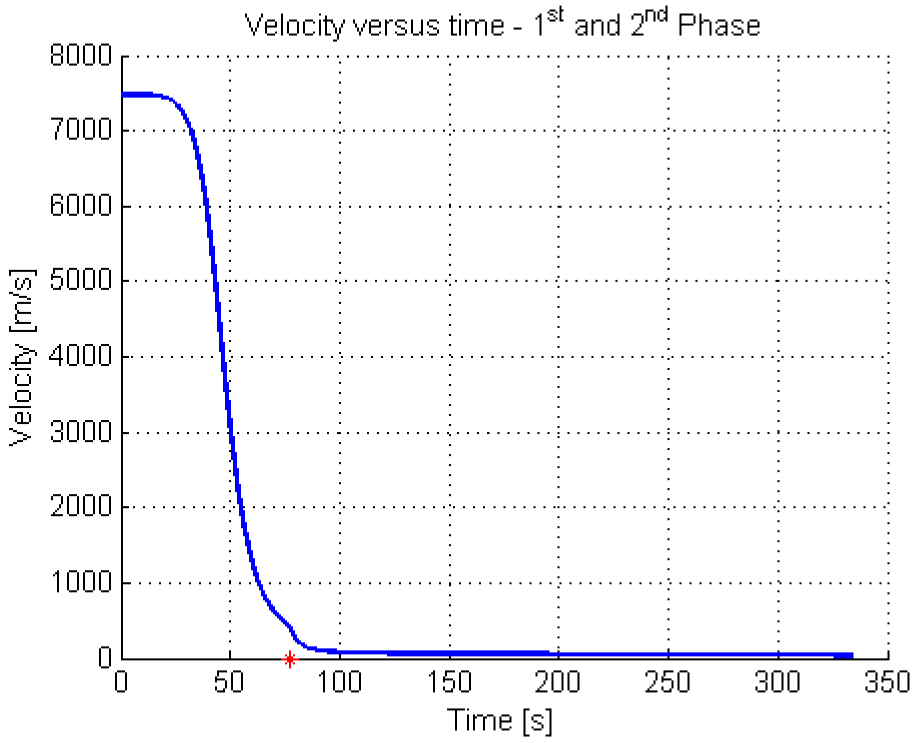


Figure 5.5. 1st and 2nd Phase – Velocity profile

The aerodynamic angles of the parachute are shown in Figure 5.6. We can notice that the parachute reaches a trim condition, with a zero angle of attack. This means that the parachute **X** body axis (along the riser) is aligned with the **X** wind axis, directed as velocity; the canopy maintains the riser on its symmetry plane.

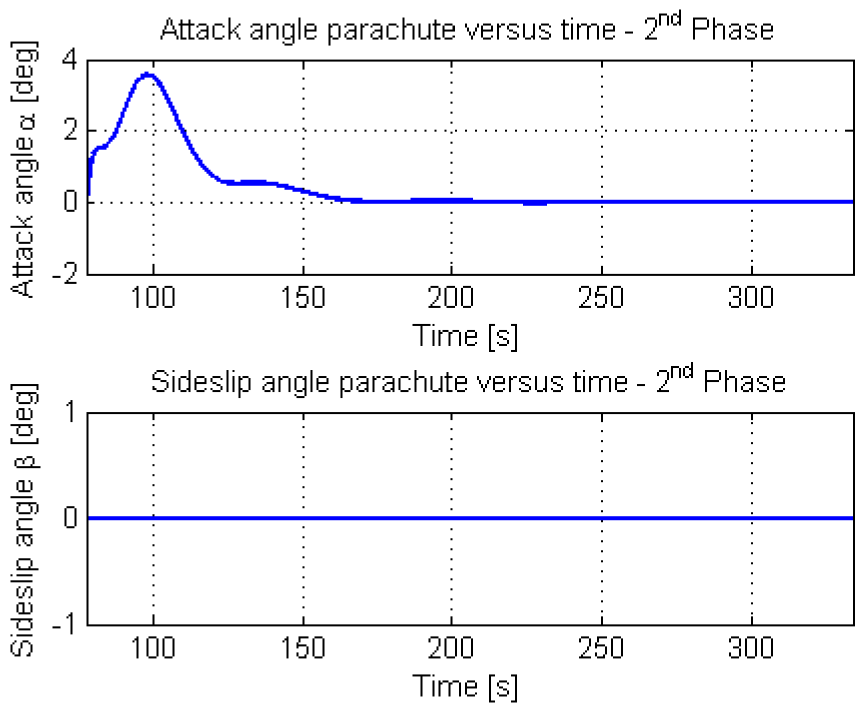


Figure 5.6. 2nd Phase – Attack and sideslip angle for the parachute

Considering a planar motion, the altitudes of the capsule and the parachute can be compared, as represented in Figure 5.7. We can observe that the two bodies maintain their distance, given by the riser length. This is also confirmed numerically by observing the modulus of the relative distance vector.

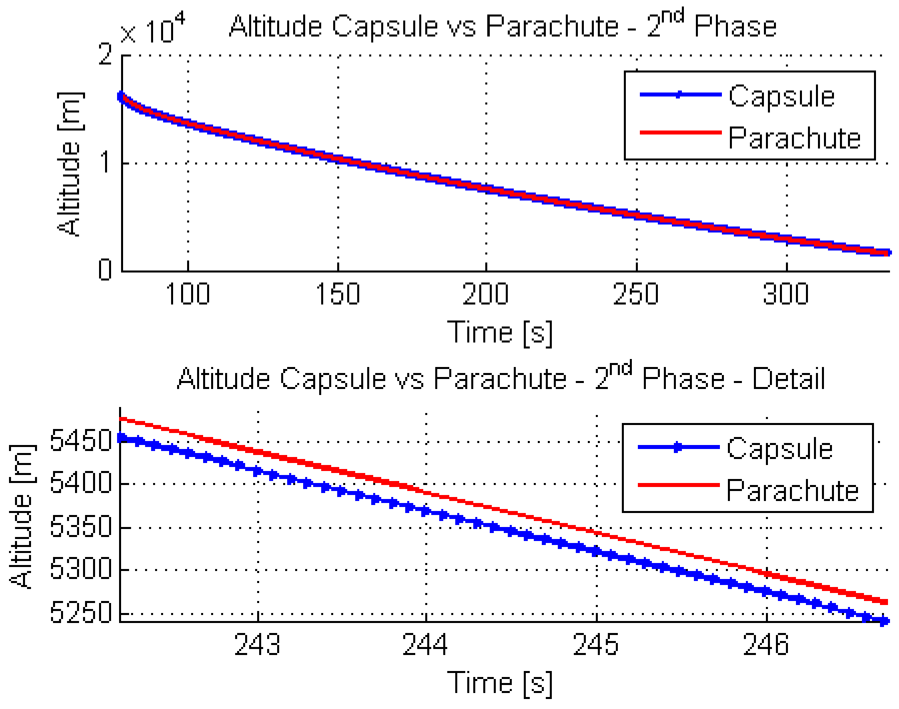


Figure 5.7. 2nd Phase – Capsule and Parachute altitude

Further consideration can be made on the constraint force. As explained in section 3.2.2, the riser force has been modeled in two different ways: using a constraint equation and a mass-less-spring-dumper model, which parameters are given in Table 5.6. The two models are compared in Figure 5.8 and we can observe that they give similar results.

Table 5.6. Mass-less-spring-dumper model parameters

Parameter	**Value**
Linear spring constant [N/m]	1000
Second order spring constant [N/m^2]	500
Damping ratio [N·s/m]	0.6

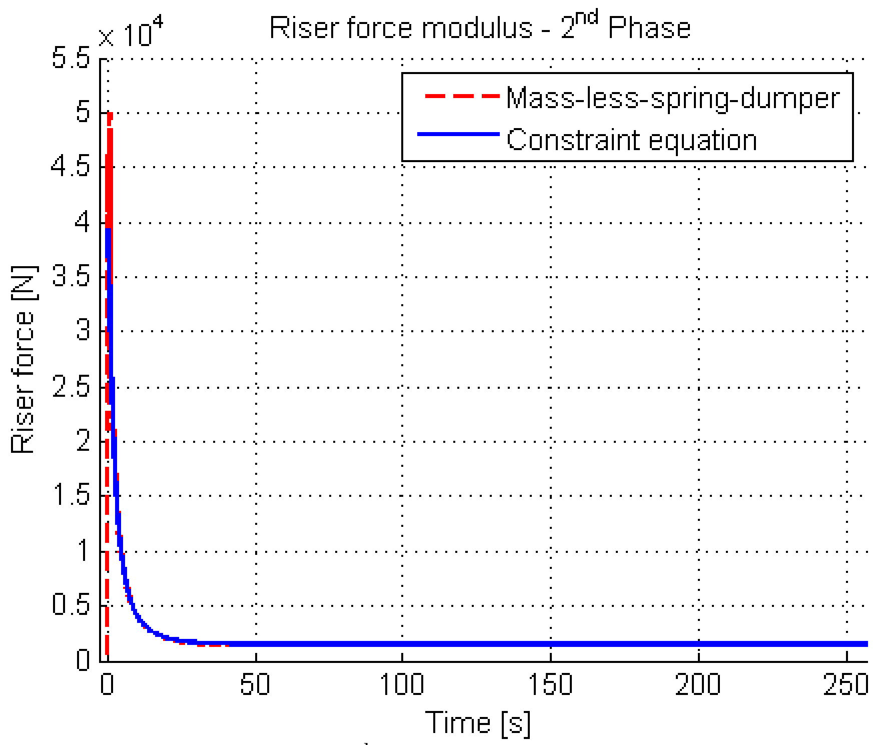


Figure 5.8. 2nd Phase – Riser force modulus

The mass-less-spring-dumper model presents some oscillations in the initial phases, missing in the constraint approach. Constraint force models are also verified by comparing force modulus with its typical values, reported in [**18**].

5.3 Complete simulation

Once the 1st phase and the 2nd phase models have been validated, the next step is to link them to obtain a complete simulation of the re-entry trajectory. To verify the correct behavior of the system, a research on past missions on Mars was done, identifying typical mission profiles. In Figure 5.9 is given a comparison between Mars entry trajectory for different missions [**26**].

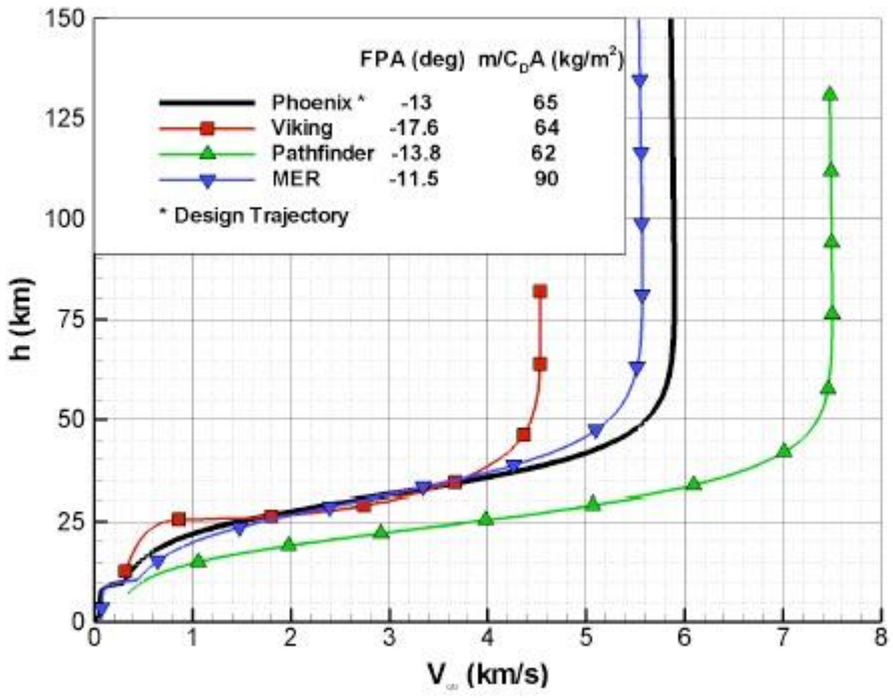


Figure 5.9. Comparison of entry trajectories in past Mars mission

Among the missions that landed safely on Mars surface, the Mars Pathfinder mission (MPF) is the one for which more data are available. The primary objective of this mission, landed on Mars in 1997, was to demonstrate an innovative low-cost and reliable method for placing a science payload on the surface of the planet. This has been the outpost for the next missions that placed rovers on the surface of the red planet. Results available from MPF ([**7**], [**8**], [**11**]) are illustrated in order to be compared with results coming from the EDL tool developed in this work. The entry condition and the geometric properties for the MPF mission used in the EDL tool are listed in Table 5.7 and Table 5.8.

Table 5.7. MPF entry conditions

Parameter	**Value**
Altitude [m]	133200
Velocity [m/s]	7264.2
Flight path angle [deg]	-14
Latitude [deg]	22.6303
Longitude [deg]	337.9976

Table 5.8. MPF entry vehicle geometric properties

Parameter	**Value**
I_{xx} [Kg·m^2]	339
I_{yy} [Kg·m^2]	248
I_{zz} [Kg·m^2]	254
Mass [Kg]	585.3
Reference surface [m^2]	5.51
Reference length [m]	2.65
Ballistic coefficient [Kg/m^2]	63

Figure 5.10 and Figure 5.11 presents the re-entry profile for the MPF mission, reconstructed by on board measurements. Parameters represented in the graphs refer to altitude, velocity and flight path angle.

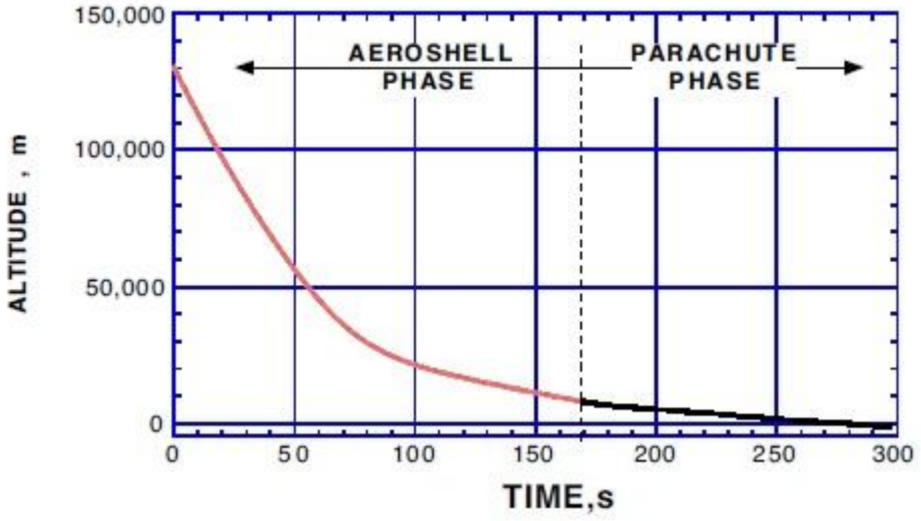


Figure 5.10. MPF altitude variation from [**11**]

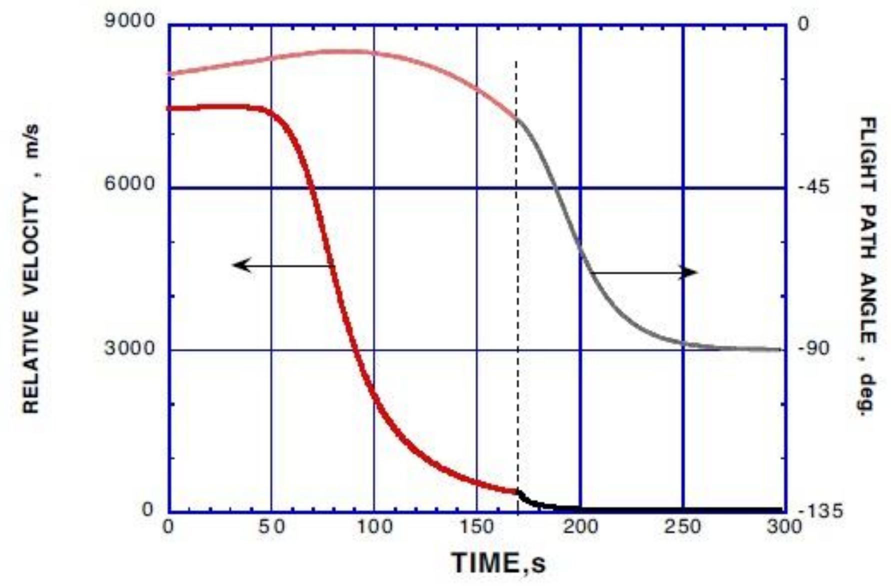


Figure 5.11. MPF Velocity and Flight path angle from [**11**]

Data of MPF mission have been used in the EDL and the results are presented. Figure 5.12 presents the altitude variation computed with the EDL tool.

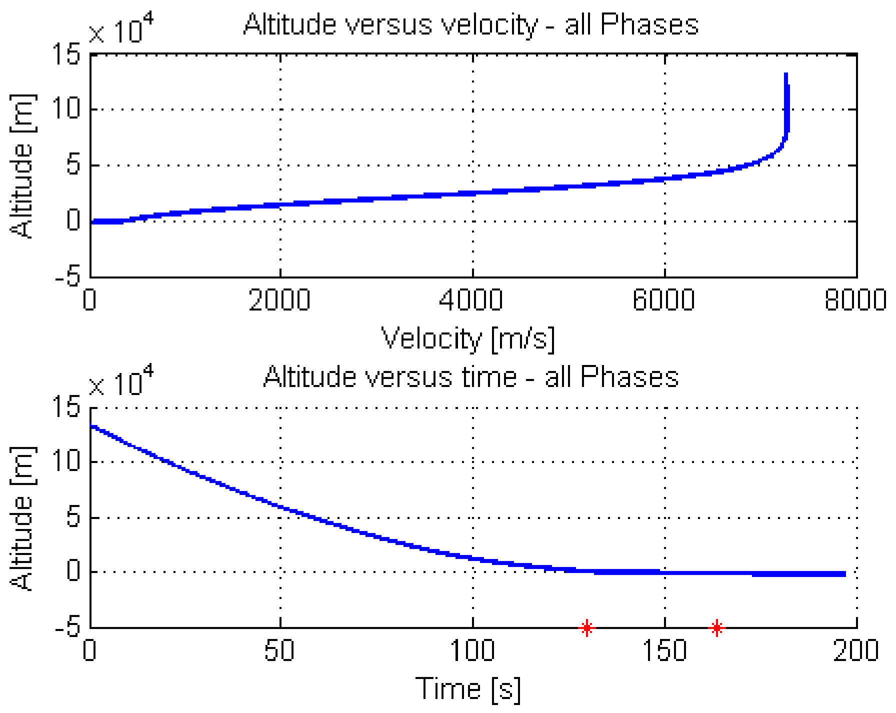


Figure 5.12. Altitude variation with MPF data

The variation of velocity and flight path angle is presented in Figure 5.13 and Figure 5.14.

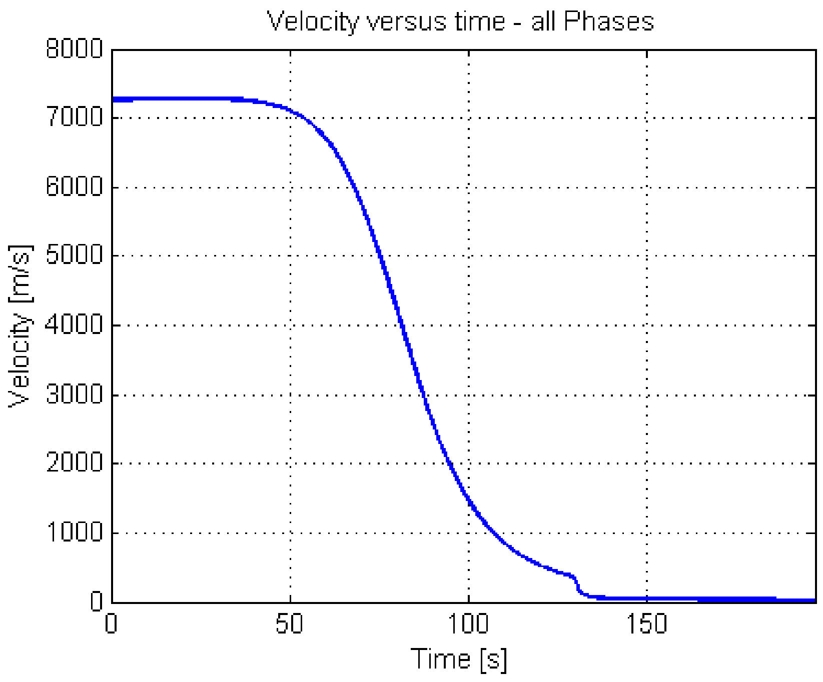


Figure 5.13. Velocity variation with MPF data

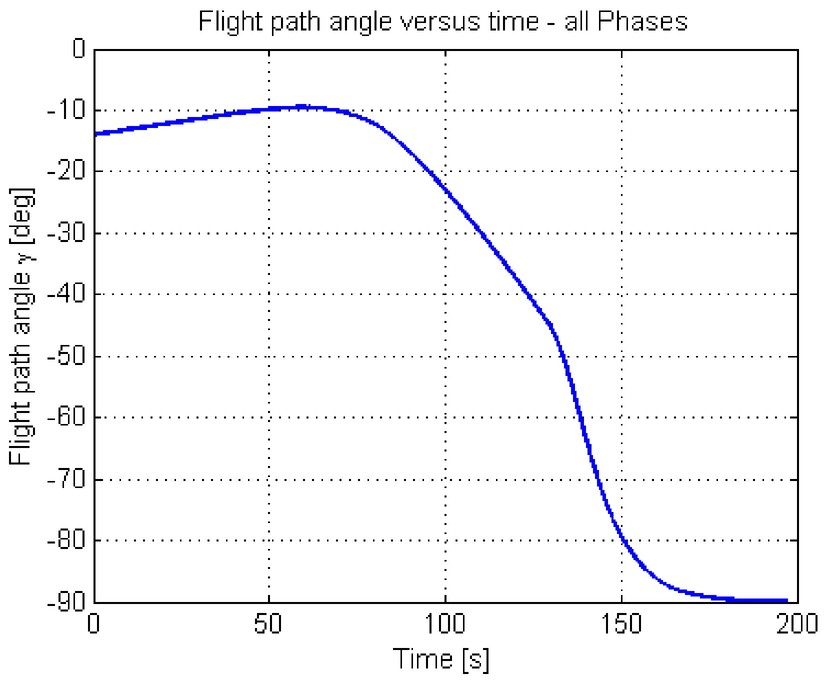


Figure 5.14. Flight path angle variation with MPF data

The above results (from Figure 5.12 to Figure 5.14) are compared to whose presented in Figure 5.10 and Figure 5.11. Differences among the figure come from the lack of data in the EDL simulator. MPF results from [**11**] comes from real data based on the whole re-entry vehicle, whereas results from the simulator are obtained through some simplification hypotheses and on the only available data. In spite of this, it can be observed a similarity among the figures, that confirm the validity of the model developed.
The altitude profile presents a change corresponding to the opening of the parachute. The flight path angle changes its value, arriving at -90 degrees at the end of the re-entry trajectory. This means that the velocity vector is perpendicular to the X local vector, and so that the re-entry vehicle land in a vertical position respect to the soil.
Focus on the velocity profile, the discontinuities at the beginning of each phase can be observed in Figure 5.15:

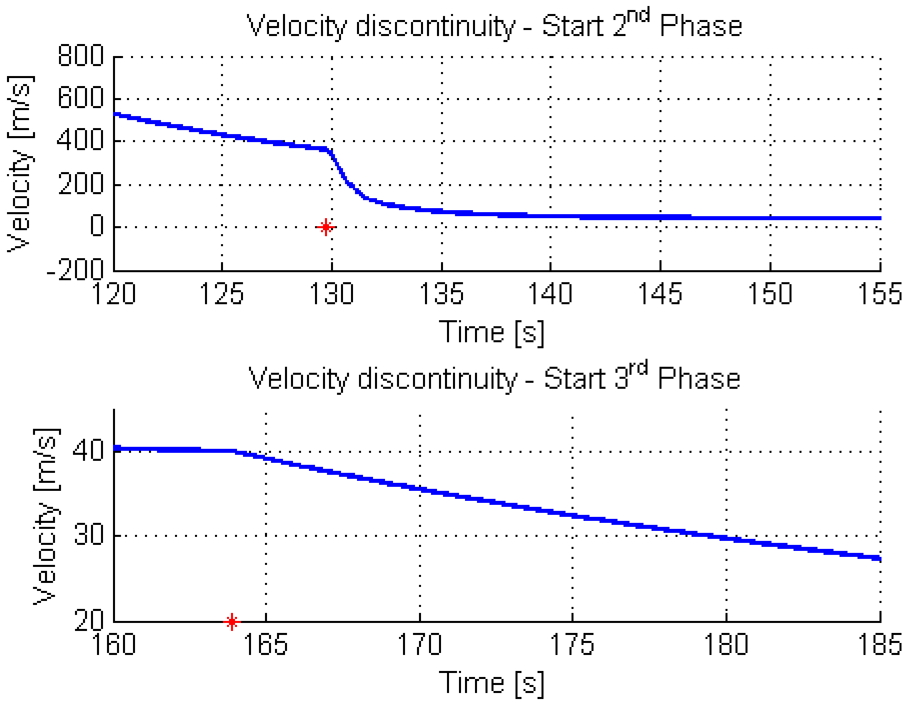


Figure 5.15. Velocity discontinuities among phases

Aerodynamic angles of the re-entry vehicle are reported in Figure 5.16. It is possible to observe that the capsule reaches a trim condition and the motion remains in the velocity plane.

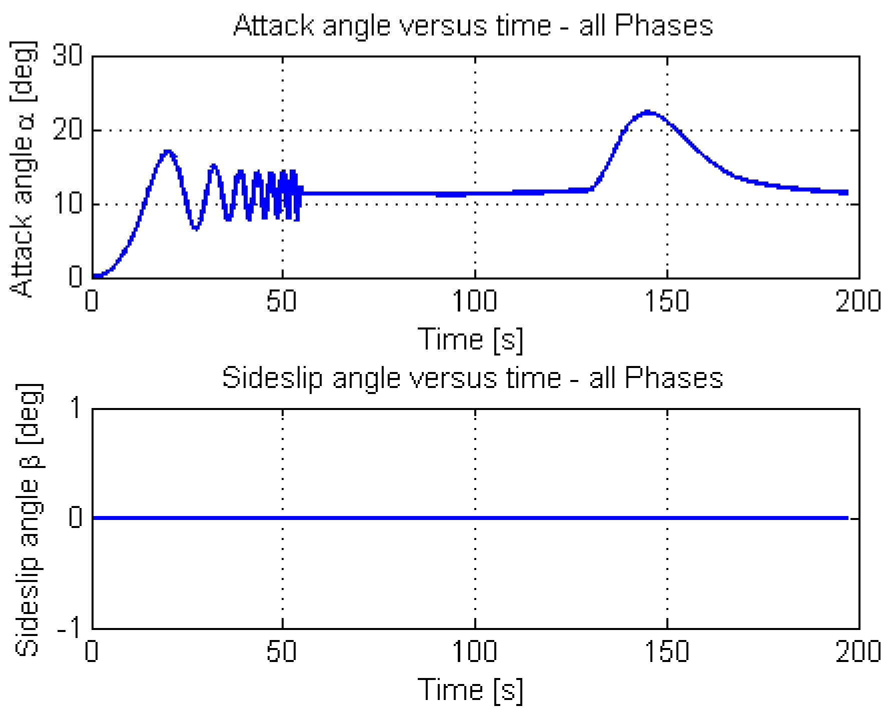


Figure 5.16. Attack and sideslip angle with MPF data

The variation with time of the RV position over Mars is shown in Figure 5.17 expressed in spherical coordinates. The longitude is defined in a range between 0 and 360 degrees, whereas the latitude is comprised in a range between -90 and 90 degrees.

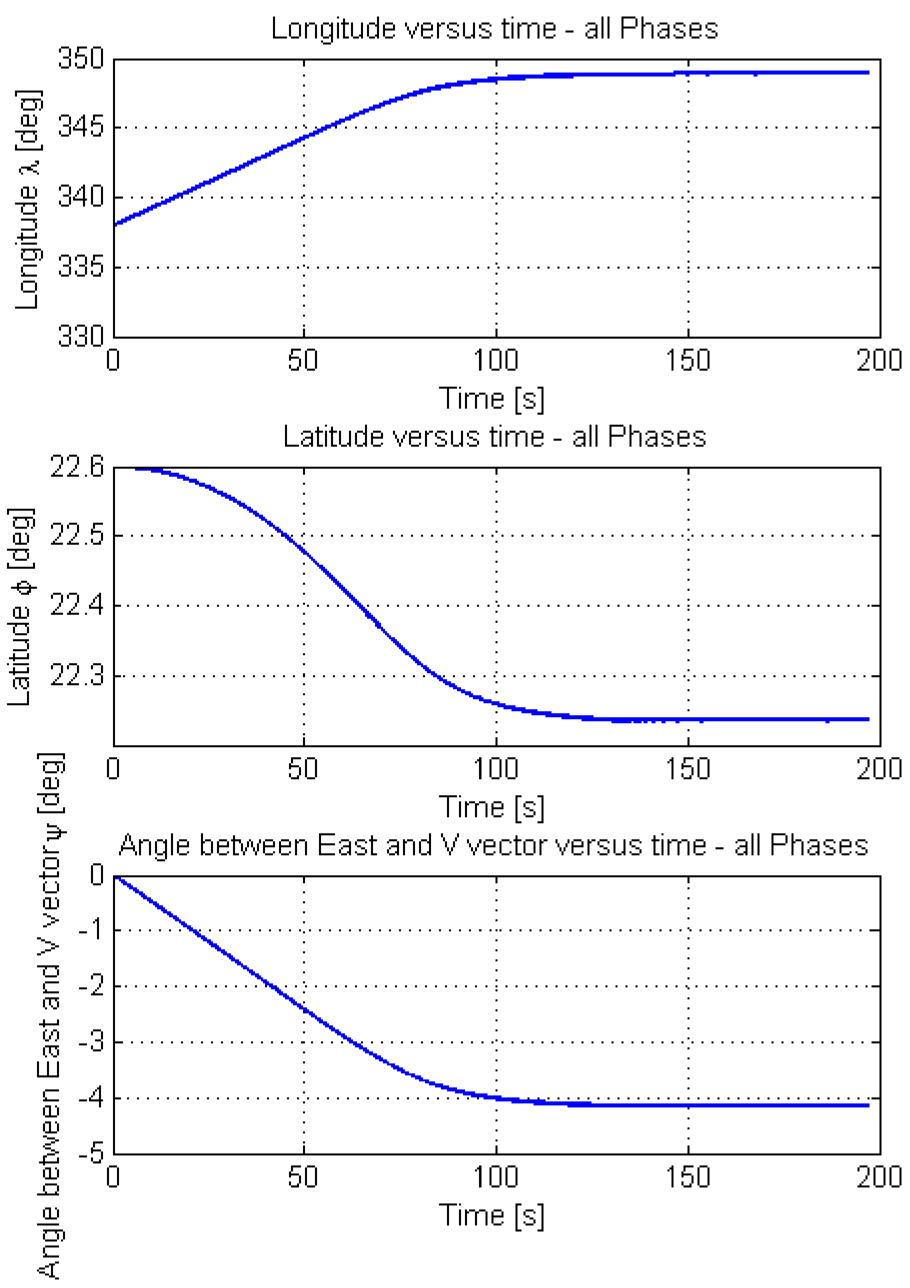


Figure 5.17. Position of the RV in spherical coordinates

Chapter 6

Integration into the simulator

In the previous chapters, the development of a tool for the analysis of Entry Descent and Landing trajectories has been presented and validated. However, the last and the most important step is still missing. The goal of this work is to develop a simulator to support planetary entry descent and landing vehicles design. This means that with the EDL tool we must be capable to visualize what really happens, in order to have both a physical and a visual representation of the problem. Once the environment and the re-entry vehicle characteristics are set, the trajectory is integrated and visualized by a graphics engine.
The EDL tool is a part of a bigger project, whose objective is to create a simulator that supports mission design for different scenarios and vehicles. The whole simulator could be capable to deal with orbital motion, descent and landing maneuvers and robotic explorations on a planet surface. To create the scenario for the virtual simulation, the Irrlicht graphic engine is used [**34**]. This is a cross platform (Windows, Linux, MacOS), high performance real-time 3D engine written in C++, open source, compatible with most video drivers (e.g. Direct 3D, OpenGL). This engine is used in lots of 3D games such as space simulators and first person shooters, and in scientific programs like Eve, a simulator for a modular robot developed at the EPFL (École Politechnique Fédérale de Lausanne) [**35**].
In this chapter, a brief description of the whole simulator is presented as well as the integration of the present tool in the simulator. Then, a description of graphic features needed to EDL is presented, identifying the terrain visualization and the algorithm to render it during motion.

6.1 Simulator overview

The current version of the simulator is capable to visualize Mars and Earth orbits and a planetary scenario with a rover moving on its surface. With the EDL tool development, a landing scenario is added to the simulator, visualizing the trajectory of the vehicle from the orbit to the planet's surface; the object oriented programming allows to easily do this.
The main program of the whole simulator is organized in different steps:
- Setting parameters. This part allows the user to define the parameters of the simulation. First of all the scenario is chosen, selecting the reference planet; then, the user defines the elements to be considered in the

simulation. For example, an orbit visualization, a re-entry one, or a rover moving on the planet surface can be selected. These simulations can be performed together, considering the sequence of events.

- Construction of classes. Base on the selected parameters, the simulator constructs C++ classes associated. For example, Astrodynamic class defines the time conversion and the orbit of the planet select. Considering the class vehicle, it is composed by the rover and the re-entry vehicle, which are constructed using parameters selected by the user. All the classes constructed allow to define the correct scenario, the elements to be displayed and the graphic properties to visualize them.
- Simulation. This part computes the simulation. Each object previous defined is updated following the sequence of events defined by the mission profile.

EDL tool is inserted in the simulator as an independent object and the re-entry vehicle is defined as a subclass of the Vehicle class, which contains also a rover and a spacecraft.

6.2 Terrain visualization

Terrain visualization is fundamental in the simulation of a re-entry trajectory. During re-entry, we must be able to observe the landing site, identifying mountains, plains, craters. So a map of the surface of the planet that contains the information needed to its correct description is necessary. This task is accomplished by using data from the Mars Global Surveyor spacecraft. This spacecraft carries an experiment called Mars Orbiter Laser Altimeter (MOLA), which produces a topographic map of the entire surface of Mars. The MOLA map is available on the internet through a binary file that contains the altitude of the surface point at each longitude and latitude. Different file are available, depending on the resolution. As first step the lower resolution (corresponding to 4 pixel/degree) has been chosen, to be able to test the scenario with the limited amount of memory.

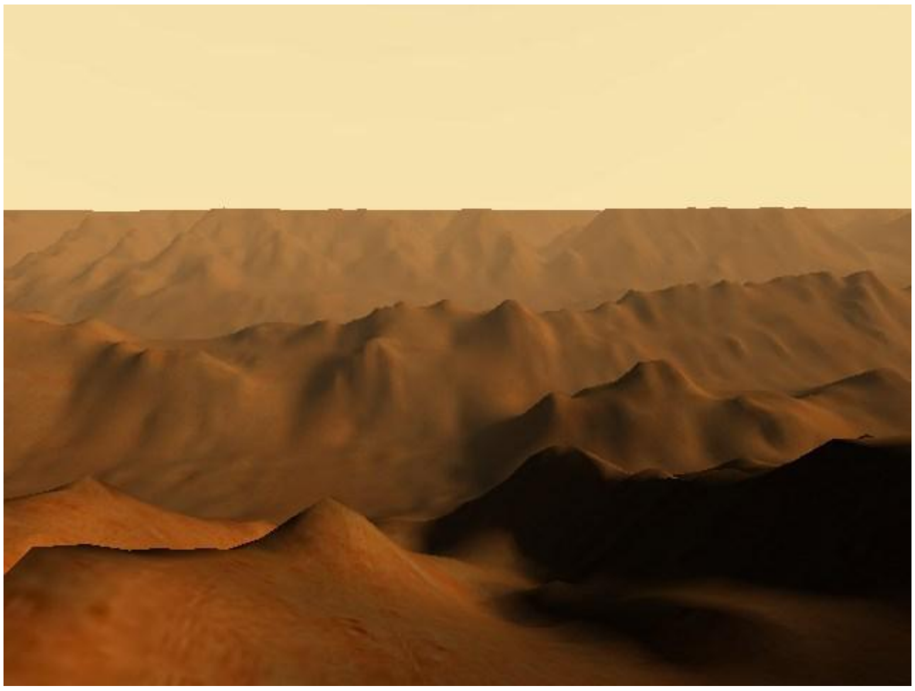


Figure 6.1. MOLA viewer

6.3 Terrain rendering algorithm

Once the terrain has been defined, the second step is to build its correct visualization during the motion. A terrain visualization depends on the observer: near objects appear more detailed than the farer ones; more difficulties are introduced with a moving viewer. This problem is solved in computer graphics by the "Level of detail" (LOD) algorithms. By definition, level of detail involves decreasing the complexity of a 3D object representation as it moves away from the viewer or according other metrics such as object importance, eye-space speed or position. There are several techniques that perform LOD algorithms, which main objectives are to reduce memory expenditure and have the best rendering of the terrain in real time. Several authors present these techniques ([**36**] [**37**] [**38**] [**39**] [**40**] [**41**]), which differ for the task they want to get (memory optimization, high resolution in moving fast, …).

In the EDL simulator, the main requirements for a terrain rendering are:

- High resolution near the surface.
- Low amount of memory.

The first one gives a good realistic description; in addition, a high resolution map helps the designer to better understand hazards near the surface and how to avoid them. The low amount of memory, instead, it is necessary to a good graphic

simulation. The time required to load graphic elements cannot be larger than the one that characterizes the motion, to have fluid real time visualization.
To accomplish tasks just presented, two guidelines characterize the terrain visualization and the rendering algorithm:

- Graphic elements not in the observer Field of View (FOV) are ignored.
- High resolution is avoided for farer elements.

The graphic simulation in the EDL tool is obtained by setting the properties of the camera (in particular its FOV) and creating two different maps: a low resolution and an high resolution one. During the re-entry trajectory, only the graphic elements in the camera FOV are displayed on the screen; when the RV is still far from the planet, the low resolution map is used, whereas at lower distance from the surface, the high resolution map is visualized.
Next subsections present the main elements needed for the terrain rendering algorithm and then a general description of it is given.

6.3.1 Tile definition

Once the MOLA map has been loaded, it is visualized on the screen by creating a mesh. The mesh is composed by independent elements called Tiles. These ones are generated by defining a grid on the planet surface (Figure 6.2); each element of the grid is a Tile (Figure 6.3) and it is defined by its vertices and its normal. The dimension of a Tile depends on the number of rows and columns set for the planet grid. Using the same resolution for the map, a Tile becomes more defined by increasing the number of elements in the grid.

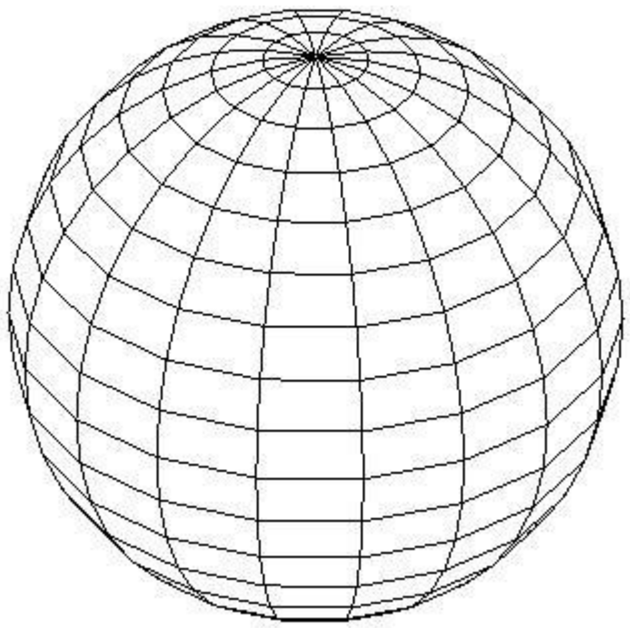


Figure 6.2. Tiles from a grid on the planet surface

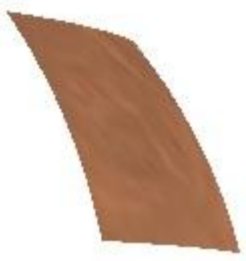


Figure 6.3. A single Tile

Once the tiles are defined, they are linked to the MOLA map, which gives the heights of the tiles' vertices respect to the mean radius of the planet. Then an image of the Mars map is added for a better visualization. The Tiles in the observer's FOV are displayed. An exemplificative image of Mars modeled with tiles is visible in Figure 6.4:

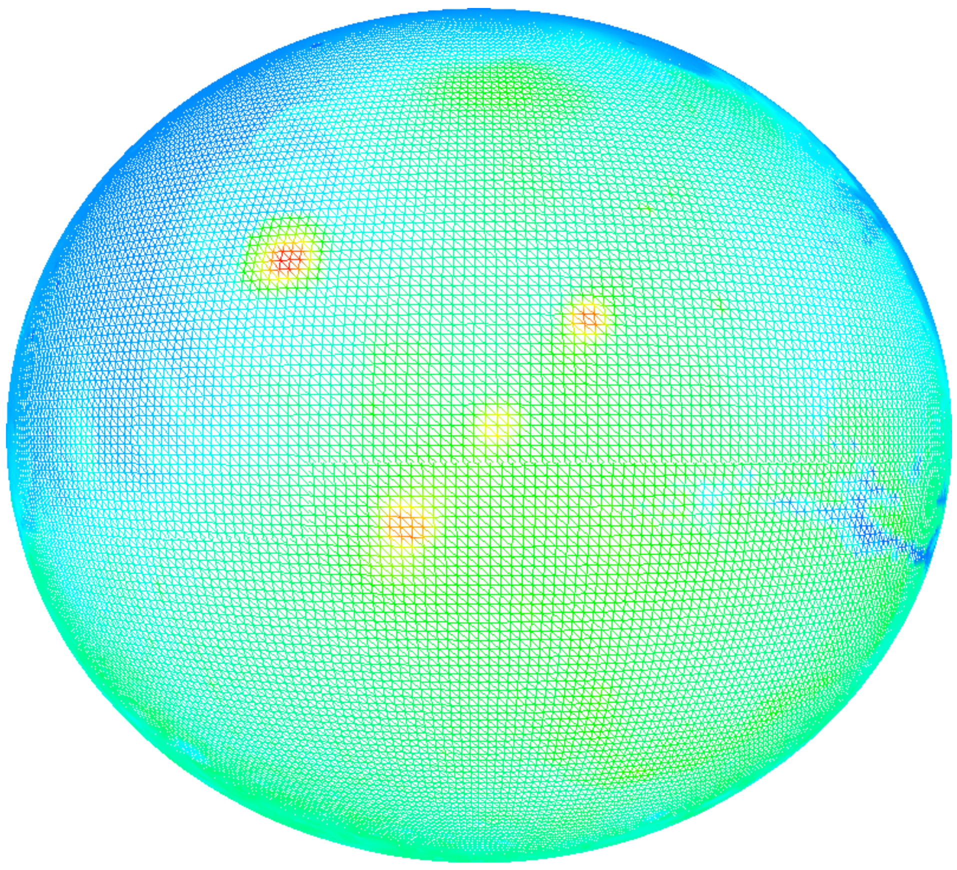


Figure 6.4. A wireframe vision of Mars modeled using tiles

Once the map is added, the planet is visualized as in Figure 6.5:

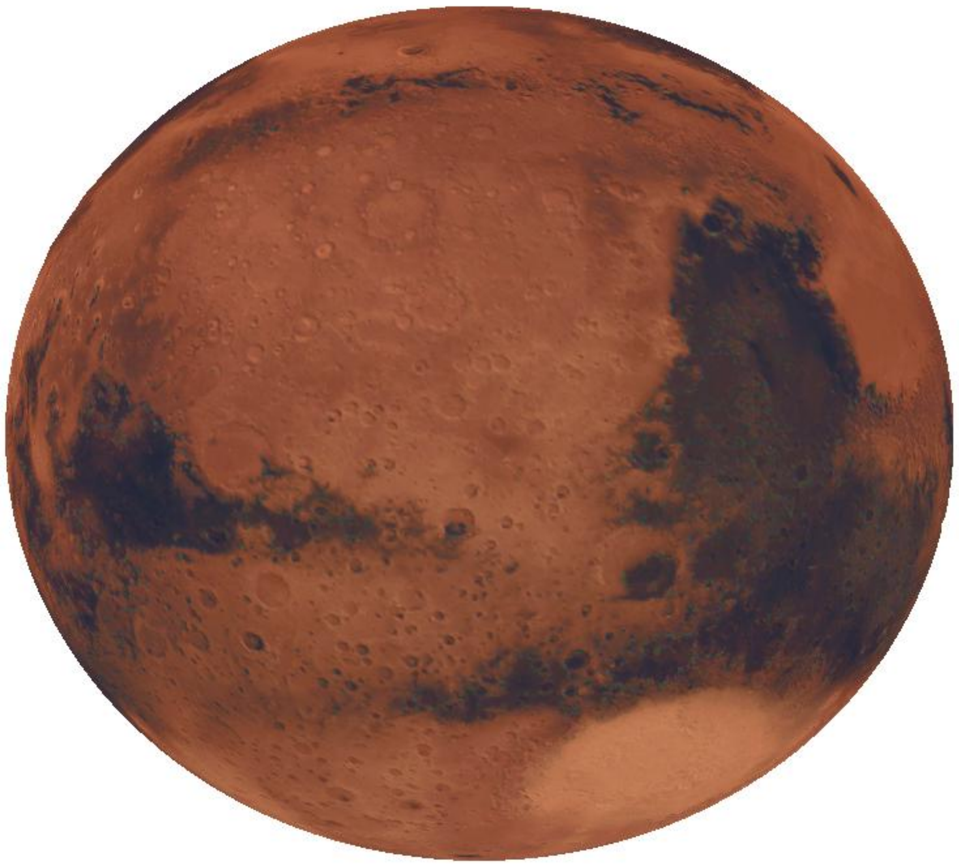


Figure 6.5. A Mars visual representation

6.3.2 Field Of View

For a better use of the memory, only the elements viewed by the observer are displayed on the screen. To select which of them have to be displayed, camera FOV is introduced as parameter. In the simulation, a FOV of 45° x 45° is considered. The FOV area on the planet surface identifies the region to display: tiles inside this region are visible on the screen (Figure 6.6).

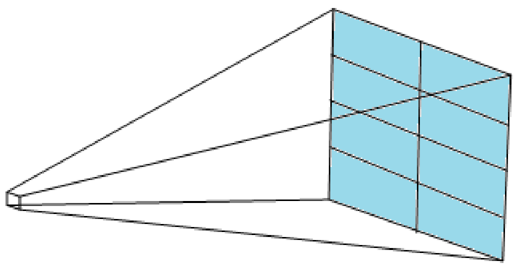


Figure 6.6. Camera FOV

If a tile is partially in the FOV, it is displayed on the screen, but there is an error. This error depends on the tile dimension, in particular the angle a tile covers on the planet (Figure 6.7). For large distances, this is not a big problem, because a precise definition of the FOV area boundaries is not requested. For smaller distance, tiles are very small (see subsection 6.3.3), and this automatically reduce the error.

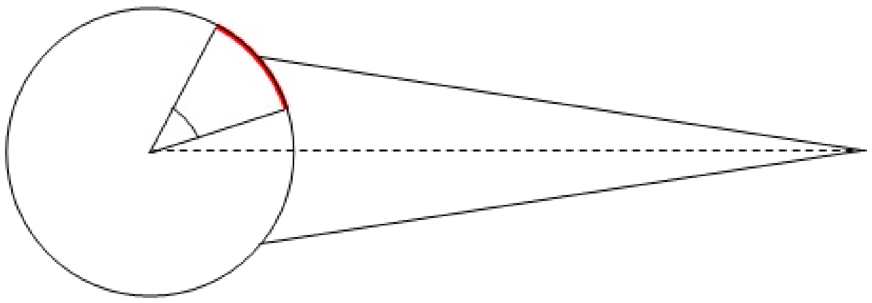


Figure 6.7. Tile partially in the FOV

6.3.3 Main terrain rendering algorithm

During the re-entry trajectory the camera image resolution of the landing site improves while the altitude of the RV decreases. This can be obtained by creating a more detailed map. Two ways are available to achieve this task:

- Increase the number of tiles. This means reducing the dimension of a single tile and thus using more points of the MOLA map (tiles are linked with MOLA points).
- Using maps with a better resolution. This means that two (or more) MOLA file have to be available and loaded.

The first solution is used to develop the rendering algorithm. With a more detailed map, the amount of memory required increases. To reduce it, this map doesn’t cover the whole planet surface, but only the region the RV passes over. Because of the trajectory is still unknown during re-entry, two observations are made:

- By observing simulation results, the maximum RV path in longitude and latitude is respectively of 10 and 85 degrees, obtained starting the re-entry at the higher latitude.
- Once the vehicle starts its re-entry trajectory, the velocity vector defines the direction it is moving towards.

Base on these considerations, the high resolution map considers a region of 90 x 15 degrees, respectively in longitude and latitude. This region is defined as a subset of the MOLA points (Figure 6.8), considering the RV initial position and the direction of the velocity vector.

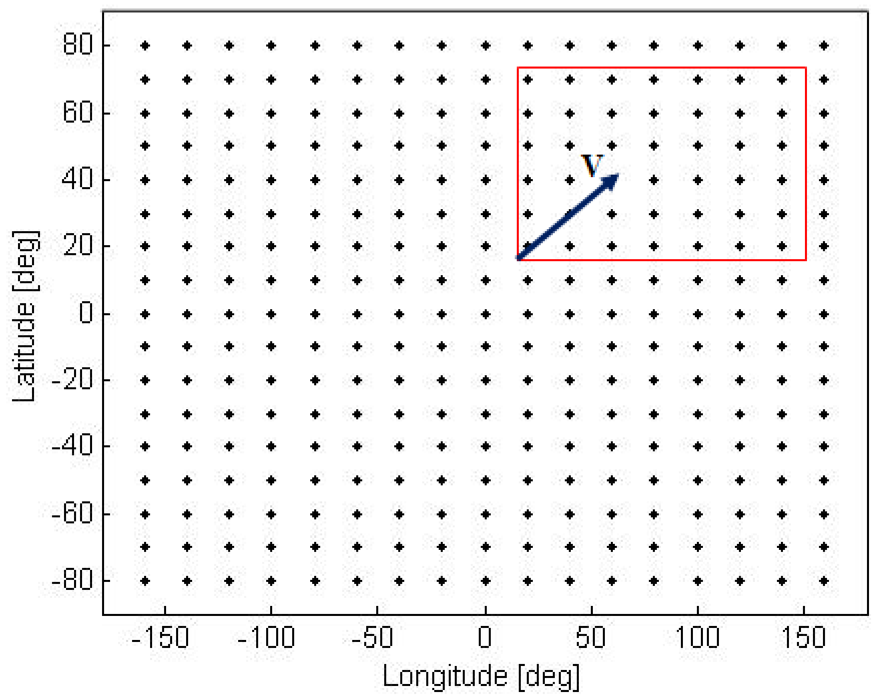


Figure 6.8. Region of MOLA points (not in scale)

At the beginning of the terrain rendering algorithm, the maps are loaded: a low resolution map, with a reduced number of tiles and a high resolution one. The low resolution map is used in a first time; then the high resolution map is used, displaying a larger number of tiles in the FOV as the altitude of the RV decrease.

6.4 Simulation of a re-entry trajectory

This section presents some images taken from the simulator after the EDL tool developed in this thesis has been integrated.

Figure 6.9 represents Mars as it appears when illuminated by the Sun.

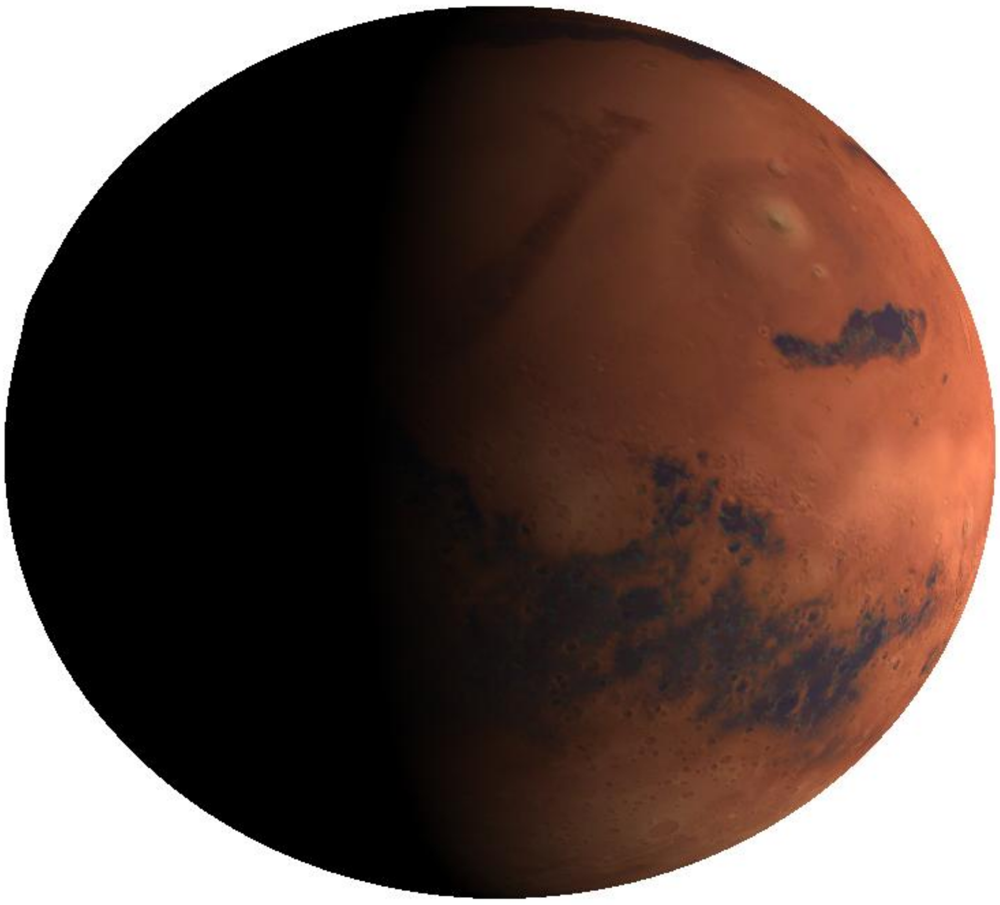


Figure 6.9. Mars illuminated by the Sun

Figure 6.10 presents the model of the RV (with a capsule shape in the 1st and 2nd phase) and in Figure 6.11 we can observe it during its approach to the planet.

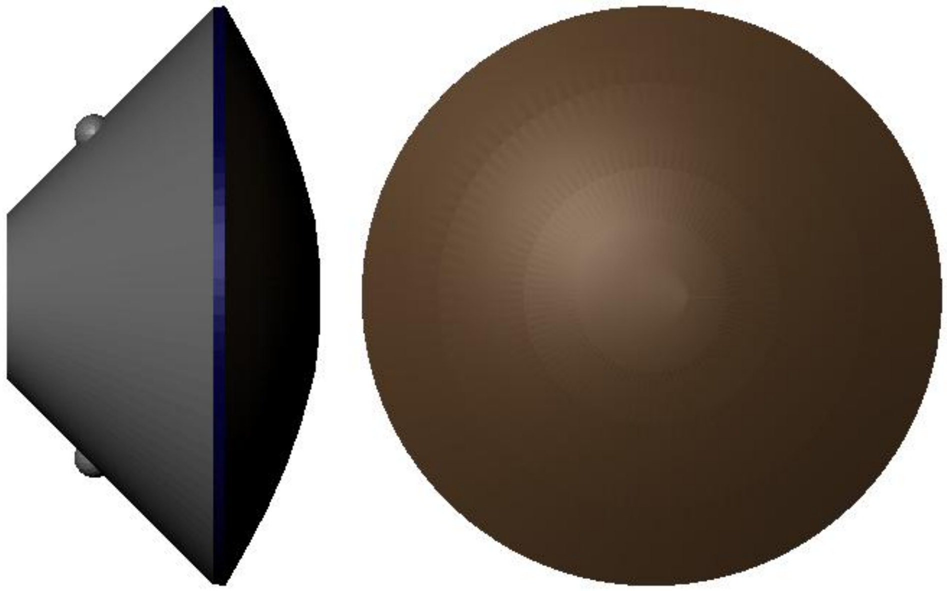


Figure 6.10. The re-entry vehicle

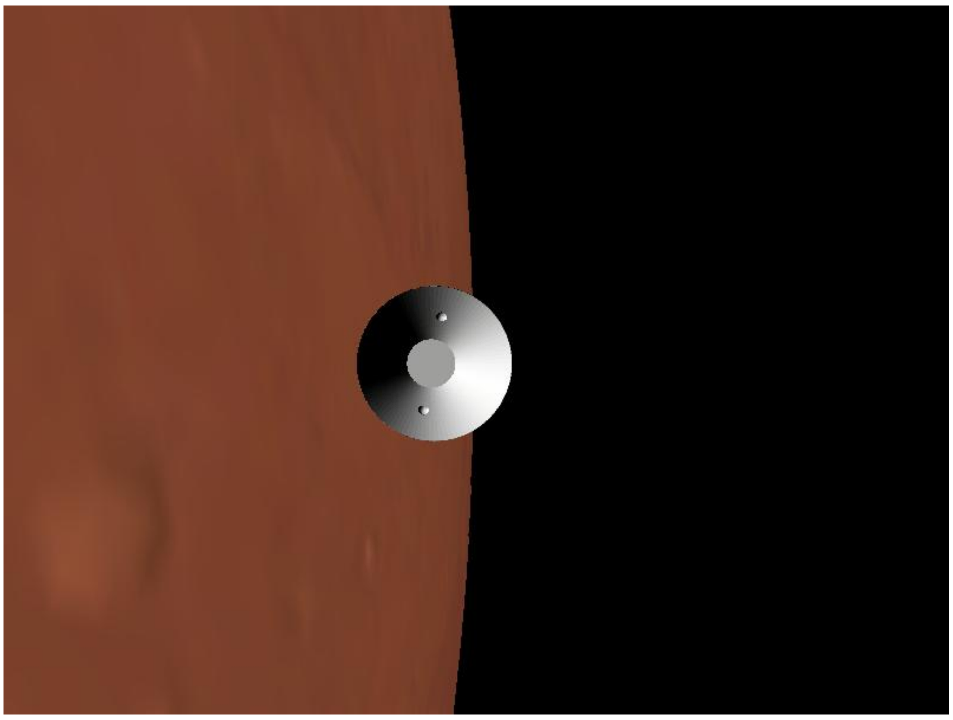


Figure 6.11. The re-entry vehicle approaches the planet

Next figures (Figure 6.12 and Figure 6.13) present the re-entry vehicle braked by the parachutes. Images are taken from the simulator by two different points of view.

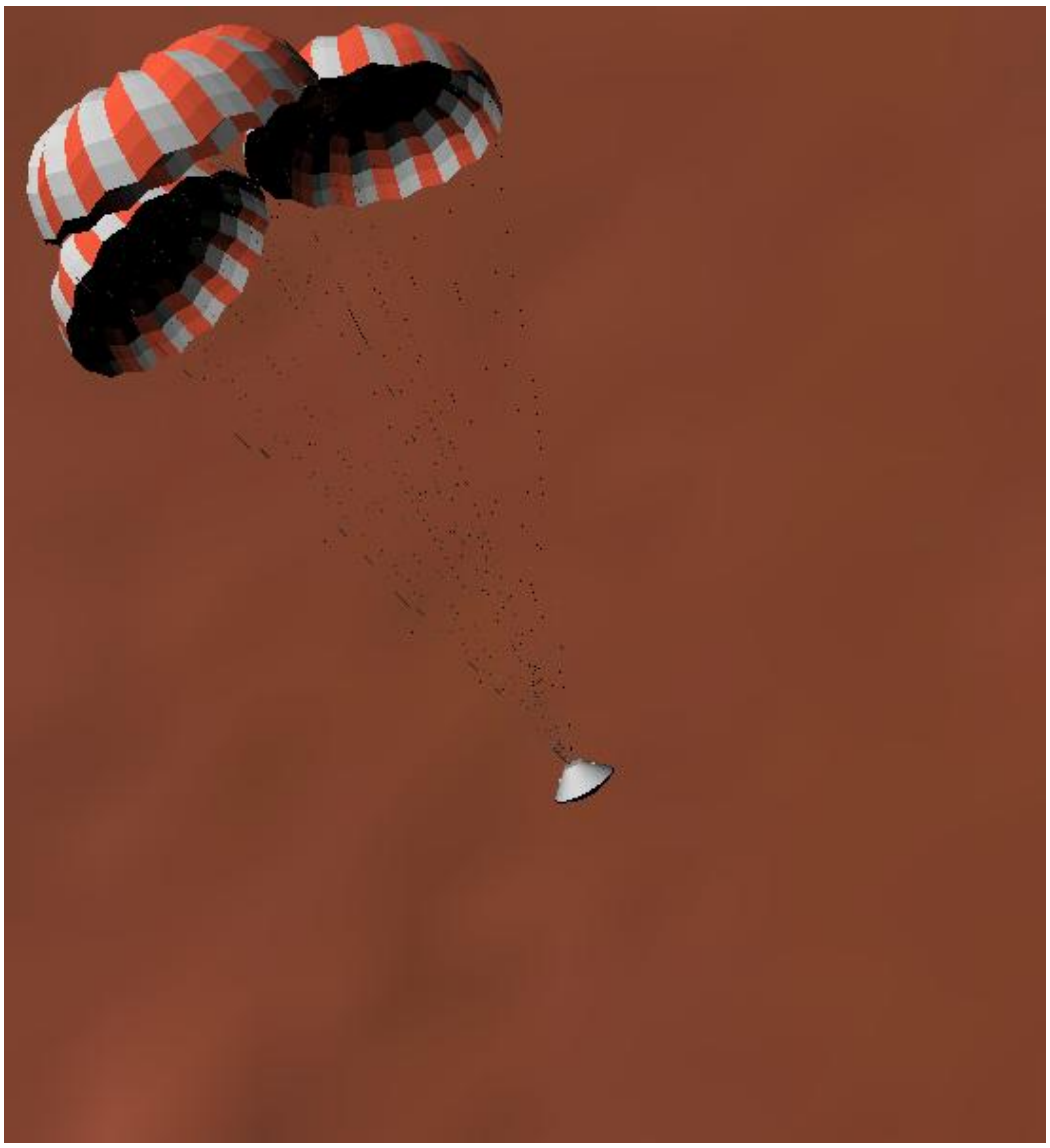


Figure 6.12. The re-entry vehicle is braked by parachutes (view 1)

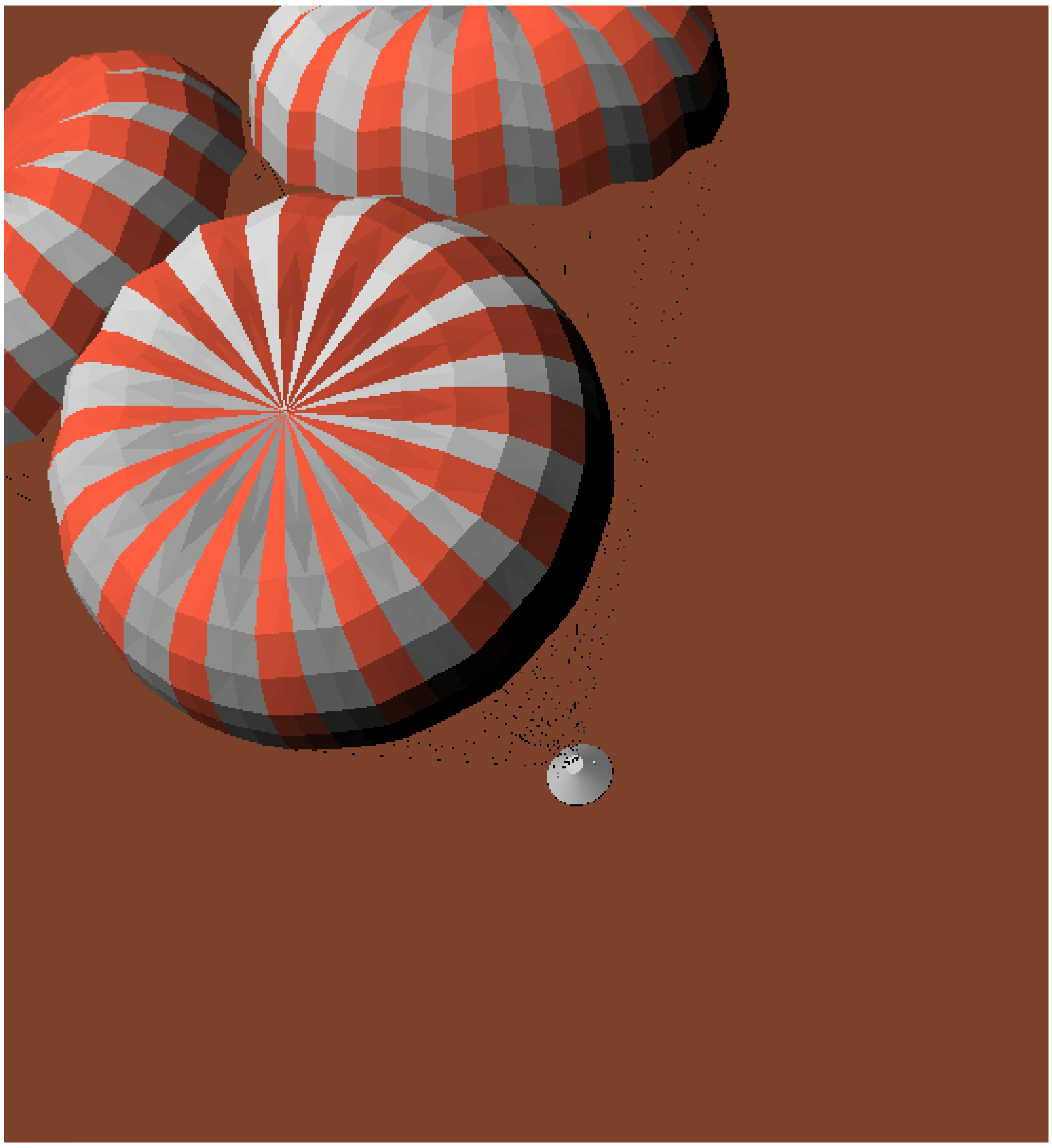


Figure 6.13. The re-entry vehicle is braked by parachutes (view 2)

Chapter 7

Conclusion

This chapter summarizes the main aspects of this work, outlines the possible uses of the tool and describes the key points for its future developments.

7.1 Final remarks

The main task of this work was to create an Entry Descent and Landing module to be connected with a graphical simulator.
Firstly, it has been presented an overview of the graphical simulators already available for re-entry trajectories; later the main differences and peculiarities of the tool developed have been underlined. A general description of an EDL system in an atmospheric environment has been given, with the description of the motion phases and typical vehicle configurations. After this general description, mathematical models are presented, considering the equations of motion (center of mass and attitude) of a rigid body subjected to aerodynamic and gravitational loads. A planar motion is considered, characterized by a single body in the first and third phase, whereas the second phase considers a multi-body approach with the simultaneous presence of the capsule and the parachute. The interaction between the two bodies has been described in two different ways: using a constraint force and a mass-less-spring dumper model. Gravitational acceleration is considered constant, whereas the environment is described by using both a mathematical model and experimental data. Aerodynamic forces are computed by interpolating existent data on aerodynamic coefficients for the re-entry vehicle. It was noticed a lack of a complete aerodynamic database to correctly describe the re-entry vehicle. Therefore, different aerodynamic databases have been considered, choosing the one more complete to test the models. The aerodynamic description of the parachute is obtained through a mathematical model. Once the mathematical models are set, the architecture for the EDL tool is presented. A modular approach has been followed, considering future upgrades. C++ language with an object oriented programming has been used. The tool has been tested, comparing the results with previous works and scientific papers with real data from past missions on Mars. Then, the EDL tool has been integrated in the whole graphical simulator. The main difficulties in the graphical vision of an EDL system are underlined, and a graphic simulation is performed.

7.2 Software utilization

Graphical view is the key feature of this work. This instrument is a valid support for mission design. The visualization can help designer to have both physical and a visual representation of what happens in reality.
Different situations can be simulated thanks to the robustness of the software and the ability to simulate failures. In this way, a robust mission design can be performed, reducing costs and time.
In addition, the simulator becomes a valid support to test flight software. The development of software to be mounted on a spacecraft is a critical phase. The possibility of testing the flight software by visualizing its effects is a valid aid to realize a robust design.

7.3 Future developments

In this work, the EDL tool for a graphical simulator is developed. To obtain a more detailed description of the motion and of the re-entry vehicle, several improvements could be introduced in future developments of the tool.
Considering the graphics, the terrain rendering algorithm can be optimized to improve the quality of the graphical representation, which is the most important element in a graphical simulator.
As described in subsection 3.2.1, a control law for the RV attitude is not modeled. This lack can be covered in two ways. Firstly by implementing a control law to achieve the correct attitude for the re-entry vehicle. Secondly, if a control law for an EDL system is already implemented, the simulator can be used to test it.
Control can also cover situations of failure or identifying and avoid obstacles during re-entry; this kind of control can be added to the simulator and tested.
As described in chapter 3, some hypotheses were made, to simplify the mathematical models adopted to describe the motion. For example, planetary rotation has been neglected and a planar motion has been assumed. These simplifications can be lifted in the future development of the tool.
Furthermore, aerodynamic forces are computed using the exponential model for density and data fitting for the aerodynamic coefficients. A more detailed model can be adopted for density computation for the different regimes of motion (free molecular, transitional, continuum). A complete aerodynamic database based on numerical simulations and wind tunnel tests could offer a more realistic description of the re-entry dynamics. Similar considerations can be done considering mass and inertia properties of the RV.
Actually, only the dynamics has been implemented in the EDL tool; further developments could include models for various subsystems. In particular, the

first subsystems to be added could be the thermal and the communications ones, to characterize loads acting on the vehicle and to link the RV to the orbiter spacecraft or directly to Earth. As a subsystem is developed, the configuration can be updated and detailed, considering the geometric parameters (e.g. Inertia) and the graphical view.

APPENDIX

Appendix A

In order to pass from a reference frames to another, in this section are presented the DCMs (Direction Cosine Matrices) between the different reference systems. Each matrix is identified as C_a^b, that means that it transform a vector in the "a-frame" to a vector in the "b-frame".

I-frame to l-frame:

$$C_I^l$$

$$\begin{bmatrix} -\sin\psi\sin\varphi\cos\lambda-\cos\psi\sin\lambda & -\sin\psi\sin\varphi\sin\lambda+\cos\psi\cos\lambda & \sin\psi\cos\varphi \\ \cos\psi\sin\varphi\cos\lambda-\sin\psi\sin\lambda & \cos\psi\sin\varphi\sin\lambda+\sin\psi\cos\lambda & -\cos\psi\cos\varphi \\ -\cos\varphi\cos\lambda & -\cos\varphi\sin\lambda & -\sin\varphi \end{bmatrix}$$

l-frame to v-frame

$$C_l^v = \begin{bmatrix} \cos\gamma & 0 & -\sin\gamma \\ 0 & 1 & 0 \\ \sin\gamma & 0 & \cos\gamma \end{bmatrix}$$

b-frame to v-frame

$$C_b^v = \begin{bmatrix} \cos\beta\cos\alpha & \sin\beta & \cos\beta\sin\alpha \\ -\sin\beta\cos\alpha & \cos\beta & -\sin\beta\sin\alpha \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix}$$

I-frame to b-frame

The b-frame is obtained from the I-frame by 3 successive rotations, listed below:

1. Rotation of an angle χ around the 3 axis ($\mathbf{Z^I}$)

$$A_3 = \begin{bmatrix} \cos\chi & \sin\chi & 0 \\ -\sin\chi & \cos\chi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

2. Rotation of an angle ξ around the 2 axis (**y'**)

$$A_2 = \begin{bmatrix} \cos\xi & 0 & -\sin\xi \\ 0 & 1 & 0 \\ \sin\xi & 0 & \cos\xi \end{bmatrix}$$

3. Rotation of an angle η around the 1 axis (**x''**)

$$A_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\eta & \sin\eta \\ 0 & -\sin\eta & \cos\eta \end{bmatrix}$$

The matrix to pass from the I-frame to the b-frame results:

$$C_I^b = A_1 A_2 A_3$$

ACRONYMS

ADCS - Attitude Determination and Control System
CFD - Computational Fluid Dynamics
DAC - Direct Simulation Monte Carlo Analysis Code
DCM - Direction Cosine Matrix
DEM - Digital Elevation Map
DSENDS - Dynamic Simulator for Entry, Descent and Surface Landing
DSMC - Direct Simulation Monte Carlo
EAGLE - Entry And Guided Landing Environment
EDL - Entry Descent and Landing
ESA - European Space Agency
FOV - Field Of View
FPA - Flight Path Angle
GAC - Global Aerospace Company
GSL - Gnu Scientific Library
HyperCMST - Hypersonic Control Modeling & Simulation Tool
HyperPASS - Hypersonic Planetary Aeroassist Simulation System
IAU - International Astronomic Union
JPL - Jet Propulsion Laboratory
LAURA - Langley Aerothermodynamics Upwind Relaxation Algorithm
LIDAR - Laser Imaging Detection And Ranging
LOD - Level Of Detail
MER - Mars Exploration Rover
MOLA - Mars Orbiter Laser Altimeter
MPF - Mars Pathfinder
MSL - Mars Scientific Laboratory
NASA - National Aeronautics and Space Administration
NED - North East Down
OBDH - On Board Data Handling
ODE - Ordinary Differential Equation
RV - Re-entry Vehicle

BIBLIOGRAPHY

[1] Peter A. Gnoffo, "Planetary-entry gas dynamics," *Annual review of fluid mechanics*, vol. 31, pp. 459-494, 1999.

[2] R. Austin, P. Banerjee, T.Bentley, D. Henriquez, B. Martin, E. McMahon, G. Sohl J. Balaram, "DSENDS - A High-Fidelity Dynamics and Spacecraft Simulator for Entry, Descent and Surface Landing," *IEEE Aerospace Conference Proceedings*, vol. 7, March 2002.

[3] David A. Henriquez, J. Bob Balaram, Garret A. Sohl, Marc I. Pomerants Bryan J. Martin, "System Engineering Challenges of Real-Time Simulation for Mars Smart Lander Entry, Descent and Landing," in *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, Monterey, California, 2002.

[4] Abhinandan Jain, Steven Myint Marc I. Pomerants, "Dspace: Real-time 3D Visualization System for Spacecraft Dynamics Simulation," in *Third IEEE International Conference on Space Mission Challenges for Information Technology*, Pasadena, 2009.

[5] Guy Johns, Alastair Pidgeon, Christian Philippe Ender John-Olcayto, "Eagle: an extensible, end to end simulation and evaluation framework for planetary EDLs," in *Internationa planetary probe workshop*, Barcelona, spain, 2010.

[6] Global Aerospace. (2010) Global Aerospace Corporation. [Online]. http://www.gaerospace.com/projects/Hypersonic/hypersonic_and_EDL.htm

[7] Robert M. Manning Robert D. Braun, "Mars Exploration Entry, Descent, and Landing Challenges," *Journal of spacecraft and rockets*, vol. 44, no. 2, March-April 2007.

[8] Robert C. Blanchard, Robert D. Braun, Pieter H. Kallemeyn, Sam W. Thurman David A. Spencer, "Mars Pathfinder Entry, Descent, and Landing Reconstruction," *Journal of spacecraft and rockets*, vol. 36, no. 3, May-June 1999.

[9] National Aerospace and Space Administration. (2010) NASA. [Online]. http://www.nasa.gov

[10] Euler. E.A., "Viking Mission overview - Lesson learned and challenges for the future," in *Mars: past, present and future*, Williamsburg, USA, 1991.

[11] Robert C. Blanchard, Sam W. Thurman, Robert D. Braun, Chia-Yen Peng, Pieter H. Kallemeyn David A. Spencer, "Mars Pathfinder Atmospheric Entry Reconstruction," *Advances in Astronautical Sciences*, vol. 99, pp. 663-692, 1998.

[12] Philip C. Knocke Prasun N. Desai, "Mars Exploration Rovers Entry,

Descent, and Landing Trajectory Analysis," *AIAA Astrodynamics specialist conference and exhibit*, 2004.

[13] Adler Mark, Erickson James Manning Robert M., "Mars Exploration Rover: Launch, Cruise, Entry, Descent, and Landing," NASA, Washington DC, 2008.

[14] Eric M. Queen Philip C. Calhoun, "Entry Vehicle Control System Design For The Mars Smart Lander," *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, August 2002.

[15] Robert D. Braun Soumyo Dutta, "Mars Entry, Descent, and Landing Trajectory and Atmosphere Reconstruction," in *48 th AIAA AErospece Sciences Meeting* , Orlando, Florida, 2010.

[16] Satya M.Anandakrishnan Frank J. Regan, *Dynamics of Atmospheric Re-Entry.*: AIAA Education Series, 1993.

[17] Patrick Sean Kenney Jason Richard Neuhaus, "A Generic Multibody Parachute Simulation Model," in *AIAA Modeling and simulation technologies conference and exhibit*, Keystone, Colorado, 2006.

[18] E. M. Queen B. Raiszadeh, "Partial Validation of Multibody Program to Optimize Simulated Trajectories II (POST II) Parachute Simulation With Interacting Forces," Hampton, Virginia, NASA TM-2002-211634, 2002.

[19] R. D. Braun C. G. Justus, "Atmospheric Environments for Entry, Descent and Landing (EDL)".

[20] Stephen R. Lewis, "Modelling the Martian Atmosphere," *Astronomy & Geophysics*, vol. 44, no. 4, August 2003.

[21] French Laboratoire de Météorologie Dynamique and Oxford groups. (2010) Martian climate database. [Online]. http://www-mars.lmd.jussieu.fr/

[22] COESA, "U.S. Standard Atmosphere 1976," U.S. Government Printing Office, Washington DC, 1976.

[23] F. McNeil Cheatwood, Prasun N. Desai Mark Schoenenberger, "Static Aerodynamics of the Mars Exploration Rover Entry Capsule," in *43rd AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, 2005.

[24] Glenn J. Bobskill, Paresh C. Parikh, Ramadas K. Prabhu, and Erik D. Tyler, "Aerodynamic Database Development for Mars Smart Lander Vehicle Configurations ," in *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, Monterey, California, 2002.

[25] Karl T. Edquist, "Computations of viking Lander Capsule Hypersonic Aerodynamics with Comparisons to Ground and Flight Data," in *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, Keystone, Colorado, 2006.

[26] Prasun N. Desay, Mark Schoenenberger Karl T. Edquist, "Aeodynamics for

the Mars Phoenix Entry Capsule," NASA Langley Research Center, Hampton, Virginia,.

[27] Thomas J. Horvath, Gary E. Erickson Kelly J. Murphy, "Supersonic Aerodynamic Characteristic of Proposed Mars '07 Smart Lander Configurations," in *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, Monterey, California, 2002.

[28] Wayne Hathaway, Leslie Yates, Pranun Desai Mark Schoenenberger, "Ballistic Range Testing of the Mars Exploration Rover Entry Capsule," in *43rd AIAA Aerospace Meeting and Exhibit*, Reno, Nevada, 2005.

[29] Robert D. Braun, K.James Weilmuenster , Robert A. Mitcheltree, Walter C. Engelud, Richard W. Powell Peter A. Gnoffo, "Prediction and validation of Mars Pathfinder Hypersonic Aerodynamic Database," *Journal of spacecraft and rockets*, vol. 36, no. 3, May-June 1999.

[30] Robert L. Kruse Robert I. Sammonds, "Viking entry vehicle aerodynamics at M=2 in air and some preliminary test data for flight in CO2 at M=11," Moffet Filed, California, NASA TN D-7974, 1975.

[31] Allison L. Hutchings, Christopher L. Tanner, Robert D. Braun Ian G. Clark, "Supersonic Inflatable Aerodynamic Descelerators for Use on Future Robotic Missions to Mars," *Journal of spacecraft and rockets*, vol. 46, no. 2, March-April 2009.

[32] Alex Ellery, Chris Welch Elie Allouis, "Parachutes and Inflatable Structures: Parametric Comparison of EDL systema for the proposed Vanguard Mars Mission," in *th International Astronautical Congress*, Bremen, Germany, 2003.

[33] Davide Ferraro, Robust Control for Entry, Descent and Landing in Mars Atmosphere, A.A. 2007-2008, Master thesis, Relator: ing. Michéle Lavagna.

[34] Irrlicht Community. (last visit November 2010) Irrlicht, lightning fast real time 3D engine. [Online]. http://irrlicht.sourceforge.net/

[35] EPFL. (last visit 2010) Biologically Inspired Robotics Groups Page. [Online]. http://birg.epfl.ch/Jahia/site/op/edit/pid/57461

[36] J. Edward , Swan Ii Baoquan Chen, "LOD-sprite technique for accelerated terrain rendering," *X. Held in* , 1999.

[37] Willem H. de Boer, "Fast Terrain Rendering Using Geometrical MipMapping," *in FlipCode featured Articles*, October 2000.

[38] Hugues Hoppe Frank Losasso, "Geometry clipmaps: terrain rendering using nested regular grids," *ACM SIGGRAPH*, 2004.

[39] Yuan-feng Yang, Sheng-rong Gong, Zhi-ming Cui Jian Wu, "A New quadtree-based Terrain LOD Algorithm," *Journal of software*, vol. 5, no. 7,

July 2010.

[40] David Koller, William Ribarsky, Larry F. Hodges, Nick Faust, Gregory A. Turner Peter Lindstrom, "Real-Time, Continuous Level of Detail Rendering of Height Fields," *ACM SIGGRAPH '96*, August 1996.

[41] Thatcher Ulrich. (2010) Continuous LOD Terrain Meshing Using Adaptative Quadtrees. [Online]. http://www.gamasutra.com/features/2000028/ulrich_01.htm

[42] Rodrigo R. Costa, Qi-Ping Chu, J.A. Mulder, Guillermo Ortega Shu-Fan Wu, "Nonlinear dynamic modeling and simulation of an atmospheric re-entry spececraft," *Aerospace Science and Technology*, pp. 365-381, 2001.

[43] Satya M. Anandarkrishnan Frank J.Regan, *Dynamics of atmospheric re-entry.*: AIAA education series, 1993.

[44] R. D. Braun C. G. Justus, "Atmospheric Environments for entry, Descent and Landing (EDL)," June 2007.

[45] Murray Wolinksy, David E. Sigeti, Mark C. Miller, Charles Aldrich, Mark B. Mineev-Weinstein Mark Duchaineau. (2010) ROAMing Terrain: Real Time Optimally Adapting Meshes. [Online]. http://www.lnl.gov/graphics/ROAM/

[46] johnson M.A., Pierre J.A.S. Haack B.R., "Reconstructing the entry, descent and landing of the Phoenix Mars Lander," in *32 Annual AAS Rocky Mountain Guidance and Control Conference* , Breckenridge, Colorado (USA), 2009.