Titulació:
Enginyeria Aeronàutica Superior

Alumne (nom i cognoms):
Arnau Pons Lorente

Títol PFC:
Study of Grain Burnback and Performance of Solid Rocket Motors

Director del PFC:
Dr. Ramon Carreras Planells

Co-Tutor del PFC:
Dr. Manel Soria Guerrero

Convocatòria de lliurament del PFC:
Quatrimestre de tardor 2012 - Defensa Gener 2013

Contingut d'aquest volum: **-**MEMÒRIA**-**

STUDY OF GRAIN BURNBACK AND PERFORMANCE OF SOLID ROCKET MOTORS

-PROJECT REPORT-

by

ARNAU PONS LORENTE

TUTOR: Dr. Ramon Carreras Planells

CO-TUTOR: Dr. Manel Soria Guerrero

January 2013

A dissertation submitted to the Department of Heat Engines,

ETSEIAT- *Universitat Politècnica de Catalunya*,

in partial fulfillment of the requirements for the degree of

Aeronautical Engineer

Escola Tècnica Superior d'Enginyeries Industrial i Aeronàutica de Terrassa

UNIVERSITAT POLITÈCNICA DE CATALUNYA

ABSTRACT

This project presents the development of a numerical code aimed to perform the grain burnback analysis and predict the performance of a solid rocket motor. The Level Set Method (LSM) has been applied to calculate the grain burnback evolution of 2D and 3D grain configurations introduced by the user as an input. In addition, in order to implement the 3D grain burnback analysis, an innovative approach called Quasi-3D has been implemented towards reducing the computational time. The Quasi-3D approach consists of modeling the 3D grain surface through few 2D reference cross-sections, which are propagated using the LSM in 2D. Then, the 3D surface is reconstructed by interpolating the grain shape between the reference cross-sections.

With respect to the internal ballistics simulation, 0D-unsteady and 1D quasi-steady models found from the literature have been used. This way, the Matlab® code developed studies three different cases: 2D grain burnback and 0D unsteady flow, Quasi-3D grain burnback and 0D unsteady flow and Quasi-3D grain burnback and 1D quasi-steady flow.

The results from 2D and Quasi-3D grain burnback and 0D unsteady flow tools have been compared against three reference cases, obtaining a high level of matching. The 1D quasi-steady flow model results are qualitative.

Keywords: Solid Rocket Motor, grain burnback, level set method, 3D grain geometry, 2D grain geometry, internal ballistics and performance.

ACKNOWLEDGEMENTS

Through this few lines I want to show my gratefulness to all that have supported me during the development of this study. Firstly, I would like to express my gratitude to my tutor Dr. Ramon Carreras Planells as well as to my co-tutor Dr. Manel Soria Guerrero, who gave me the golden opportunity to face big personal challenge with this project on the field of Solid Rocket Motors and guided me thorough the entire process.

Secondly, I would like to acknowledge the International Center for Numerical Methods in Engineering (CIMNE) [17] for providing freely the software GID. Particularly, I am grateful to Marti Coma Company, who aided me during the learning process of the GID software.

Finally, I would like to thank all my family, but specially my parents Baldiri and Núria and my sister Helena. Foremost, I want to thank my girlfriend Orzuri for her constant support and patience.

Sincerely,

Arnau Pons Lorente

Audaces fortuna iuvat.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

AIAA American Institute of Aeronautics and Astronautics

CAD Computer-aided Design

CSAR Center for Simulation of Advanced Rockets

DOD Department of Defense

ESA European Space Agency

GUI Graphical User Interface

HRM Hybrid Rocket Motor

HW Hardware

IGES Initial Graphics Exchange Specification

LRE Liquid Rocket Engine

LSM Level Set Method

MDF Minimum Distance Function

NASA National Aeronautics and Space Administration

NAWC Naval Air Warfare Center

ODS Ordinary Differential Equations

SFT Static Firing Test

SI International System of Units

SRM Solid Rocket Motor

SRB Solid Rocket Booster

SW Software

US United States

LIST OF SYMBOLS

Symbol	Definition	Units
$\dot{r}_b$	Burning rate.	m/s
a	Burning rate coefficient.	$\frac{m/s}{Pa^n}$
n	Burning rate pressure exponent.	$-$
P_c	Combustion chamber pressure.	Pa
F	Thrust	N
$\dot{m}_d$	Mass flow rate exiting the nozzle	kg/s
u_e	Exit velocity of the gas.	m/s
P_e	Static pressure of the gases at the nozzle exit.	Pa
P_a	Atmospheric pressure.	Pa
A_e	Area of the nozzle exit section.	m^2
F_c	Characteristic thrust.	N
γ	Heat specific value.	$-$
R_0	Universal gas constant.	$\frac{J}{mol\ K}$
$\bar{M}$	Mean molar mass of the gas.	kg/mol
R	Gas parameter, equal to $\frac{R_0}{M}$	$\frac{J}{kgK}$
T_c	Combustion chamber temperature.	K
C_F	Thrust coefficient.	$-$
A_t	Area of the nozzle throat.	m^2
c^*	Characteristic velocity.	m/s
ε	Expansion ratio.	$-$
I_s	Specific impulse.	m/s
ϕ	Level Set Method function.	$-$
$\vec{\tau}$	Tangent vector.	$-$
$\vec{n}$	Normal vector.	$-$

P_b	Perimeter of the cross-section.	m
L_{grain}	Length of the grain.	m
A_b	Burning area.	m^2
R_{case}	Motor case radius.	m
w	Grain burnt depth.	m
A_p	Port area.	m^2
M_t	Mach number at the nozzle throat.	$-$
ρ_t	Gas density at the nozzle throat.	kg/m^3
ρ_c	Gas density at the combustion chamber.	kg/m^3
T_t	Static temperature at the nozzle throat.	K
u_t	Velocity at the nozzle throat.	m/s
$\dot{m}_g$	Mass flow rate due to propellant combustion.	kg/s
V_c	Combustion chamber volume.	m^3
Δt	Time step.	s
E	Total energy.	J/kg
h_f	Enthalpy of combustion products.	J/kg
u_f	Injection velocity of the combustion products.	m/s
M	Mach number.	$-$
c_p	Specific heat capacity at constant pressure.	$\frac{J}{kgK}$
c_v	Specific heat capacity at constant volume.	$\frac{J}{kgK}$
h_c	Enthalpy of the reactants at the initial temperature of the chamber.	J/kg
h_e	Enthalpy of the products at the ejection temperature.	J/kg
u_c	Velocity at the combustion chamber.	m/s
M_p	Propellant mass.	kg
ρ_p	Propellant density.	kg/m^3
V_p	Propellant volume.	m^3

1. INTRODUCTION

1.1. AIM OF THE PROJECT

The aim of the project consists of developing a computer numerical code able to perform the grain burnback analysis and calculate the performance of a solid rocket motor.

1.2. SCOPE

Next, the scope of the different areas in which the project is divided is presented:

- **Grain burnback.**
  - Research of state of the art of grain burnback analysis.
  - Development of a numerical code to implement 2D grain burnback analysis.
  - Validation of the numerical code developed for 2D grain burnback analysis.
  - Development of a 3D grain burnback analysis model.
  - Development of a numerical code to implement the 3D grain burnback analysis model.
  - Validation of the numerical code developed for 3D grain burnback analysis.
  - Out of scope:
    - Consideration of erosive burning effects.
    - Consideration of the effects of the initial propellant temperature.
    - Consideration of rocket motion effects.
- **Internal ballistics**
  - Development of a numerical implementation of 0D unsteady flow model.
  - Validation of the numerical implementation of the 0D unsteady flow model.
  - Research of state of the art of 1D flow model.
  - Selection of the 1D flow model to be implemented numerically.
  - Development of a numerical code to implement the selected 1D flow model.
  - Realization of the coupling between the 2D grain burnback and 0D unsteady flow model.
  - Realization of the coupling between the 3D grain burnback and 0D unsteady flow model.

    - Out of scope:
        - Development of an own 1D quasi-steady flow model.
        - Complete coupling the 1D quasi-steady flow model with the Quasi-3D burnback analysis.
- **Economic aspects**
    - Estimation of the engineering work cost.
    - Estimation of the hardware and working infrastructure cost.
    - Estimation of the software cost.
    - Estimation of the project budget.
    - Out of scope: consideration of taxes in the project budget.
- **Environmental effects**
    - Research of the environmentally harmful effects of solid rocket motors testing.
    - Estimation of the environmental benefits reached through this study.
- **Documentation and marketing**
    - Composition of a Project Report.
    - Generation of 2D grain burnback videos.
    - Generation of 3D grain burnback videos.
    - Out of scope: development of a Graphical User Interface for the code.

1.3. REQUIREMENTS

- The numerical code shall be able to perform 2D the grain burnback analysis of a solid rocket motor.
- The numerical code shall be able to perform 3D the grain burnback analysis of a solid rocket motor.
- The numerical code shall be able to calculate the performance of a solid rocket motor.
- The numerical code shall be able to calculate the grain burnback and internal ballistics with reduced computational cost.
- The numerical code shall be able to deal with defined geometry files given by the user. These geometry files are obtained from the discretization of grain CAD designs.
- The numerical code shall be written in Matlab®.
- The 2D and 3D grain burnback models shall use a spatially constant burning rate.

1.4. JUSTIFICATION

Numerous parametric studies conducted in order to optimize the motor design have typically defined the design and development phase of Solid Rocket Motors (SRMs). Basically, technological goals as well as commercial aims have impelled the research of SRM. This way, these actuators have established the possible innovative configurations analyzed and the constraints for optimization problems. Once the design bases are defined, the optimum configuration is selected and then, the critical review of design configuration begins. In this phase, in order to predict the full-scale motor configuration, selected propellant, grain design and motor configuration are analyzed in detail. The basic tools to perform this detailed analysis combine the theoretical performance prediction methods with numerical models implemented in simulation codes that estimate the SRM grain burnback and motor performance. Nonetheless, in order to verify the results obtained, it is necessary to match with experimental data obtained from subscale motor test, or from similar SRMs designs or from previous firings on full-scale motors. This verification is a risky and expensive process. As it is indicated in [16], costly redesign, delays and loss of reliability in the SRM design program can be originated due to the incorrect consideration for the differences between the theoretical and the resulting real performance of the full-scale SRM.

Therefore, in the design and development of SRM, the increase of grain burnback as well as internal ballistics prediction capabilities can enhance SRM reliability. Meanwhile, this enhancement can also drop off design and development costs, mainly related with experimental activities (especially the ones related with Static Firing Tests (SFTs) cost [8]). The only way to reach these targets is through an accurate numerical simulation and a subsequent comprehension of the physics interacting the grain burnback and internal ballistics.

Nonetheless, these necessities have to face two challenging issues. Firstly, they must possess the capability to represent in an accurate manner the predominant physical phenomena. Secondly, the numerical codes have to face the problem with reduced computational times [8] . This way, the design modification can be implemented quickly and the capability to test several kinds of grain geometries rapidly is allowed. Particularly, this last constraint limits considerably the possibility to use fully developed 3D models of flow field and grain geometries. The bigger the detailed implemented in the numerical code, the greater the computational time required. Thereby, simpler modelization methods for 3D grain geometries are required. Additionally, the implementation of 3D flow field models is out of scope of numerical codes used to preliminary SRM designs or academic proposes.

In few words, as it has been aforementioned, in order to check and prove the reliability of a solid rocket motor design it is necessary to perform many experimental tests, which are expensive and time-consuming. Therefore, the ability to calculate and predict with precision the performance during the whole operation of the SRM is the best way to reduce the costs and time of the SRM development. With this philosophy, the better the simulations are, the lesser tests that are needed to check and validate the design. Consequently, the need of the numerical code here presented is completely justified in order to test quickly different configurations in a preliminary design phase. This way, the application of simple flow models in the numerical code is completely reasonable. Nonetheless, for preliminary designs is essential to include in the numerical code here presented a especial modeling of 3D grain surfaces. Finally, this study contributes with the numerical code here presented to the SRM academic field by proposing a fast way to perform the 3D grain burnback analysis, called Quasi-3D model.

2. SOLID ROCKET MOTORS OVERVIEW

A rocket motor is a propulsion system aimed to apply an impulse to a payload and typically, to do it within a specified period of time. It is a pure reaction system, in which the thrust is obtained from the mass variation of the system itself. Consequently, it can be considered as a typical energy transfer system, where high pressure and high temperature gases are generated into it due to the combustion process. The gases generated from this combustion, known as combustion products, are expanded through the nozzle. So, the differences in pressures and velocities between the chamber and the nozzle exit results in a propulsive force.

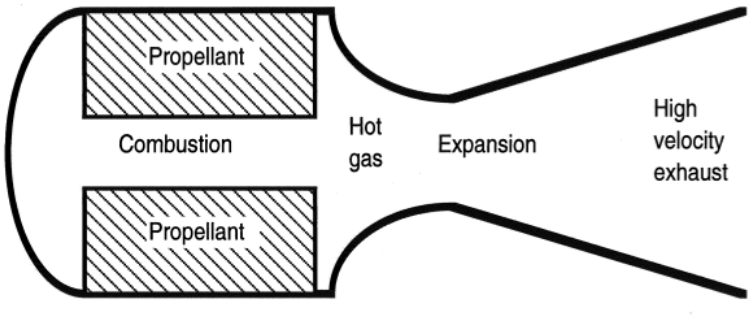

Figure 1. Basic scheme of a Solid Rocket Motor (source [1]).

Most of the rocket propulsion systems are based on chemical combustion. Considering which type of propellant is used, rocket motors can be classified in three groups: Solid Rocket Motor (SRM), Liquid Rocket Engine (LRE) and Hybrid Rocket Motor (HRM). As it name implies, in case of SRM the propellant is in solid state. SRM are very commonly used rocket motors due to being relatively simple, easy to apply and maintain whilst compared with LRE. As it is observed in Figure 1, the fuel and the oxidizer are contained and stored directly in the combustion chamber. It is therefore enough to ignite the propellant and no other chemical has to be added for combustion to take place. Note that there are several propellant compositions, usually double base or composite, having all of them different properties.

An example of the structure of a SRM is shown in Figure 2. The basic structure is formed by the motor case, solid propellant grain, igniter, nozzle, insulation and liner. If the nozzle of the SRM has thrust vector control (e.g. the SRM of the Figure 2), in the structure a thrust vector actuator is also included.

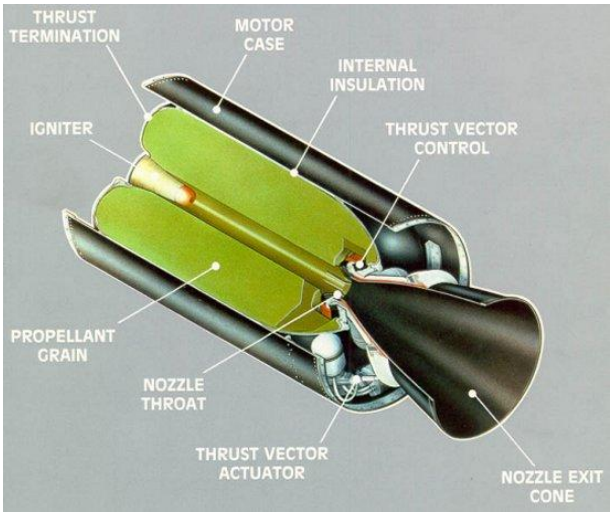

Figure 2. Basic nomenclature and parts of a Solid Rocket Motor (source [18]).

The simplicity presented by SRM is also a drawback. Requiring nothing external to burn, the burning of SRMs cannot be regulated in-flight by providing more or less fuel to the motor. This way, an active control form (real time) of thrust provided by a SRM is impossible. Indeed, there are two ways in which combustion stops: when the propellant has depleted or when it is not possible to sustain combustion temperature and pressure conditions. Thrust termination systems are used as failsafe systems to stop failing rockets or when a stage rocket is separated from the main rocket. Nevertheless, these systems usually destroy the motor and do not allow the re-ignition.

Due to impracticality of applying active control, the interest lies in a form of passive control. The passive regulation of the SRMs is reached through the propellant grain geometry. That is to say that considering the requirements of the mission, the thrust vs. time curve is determined. Then, the grain geometry is designed in order to provide this required thrust vs. time curve. Some examples of 2D grain geometries and associated thrust vs. time curves are shown in Figure 3.

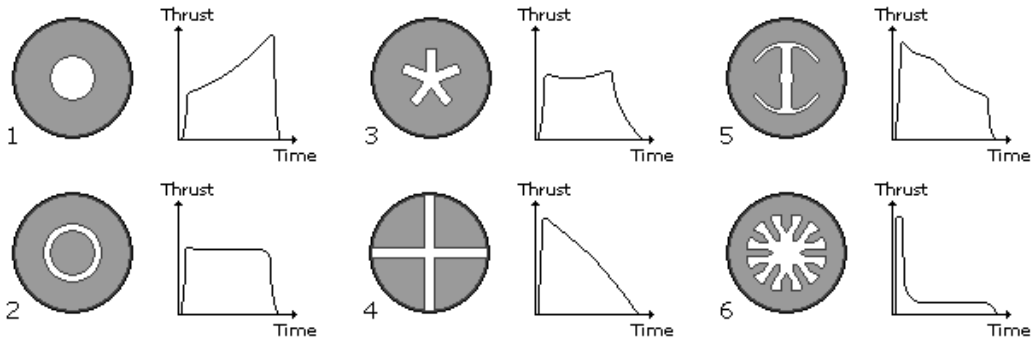

Figure 3. Thrust curve for several SRM 2D grain geometries (source [2]).

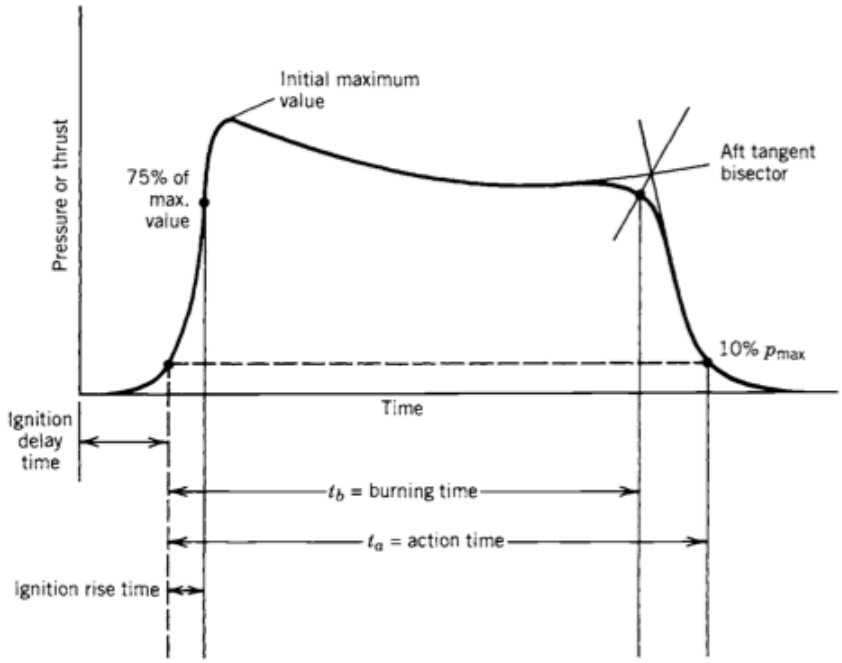

Figure 4. Example of a SRM thrust curve identifying main characteristic points (source [11]).

Consequently, the main goal of the grain designer is to fulfill the requirements of the thrust-time schedule of the mission. Note that the mission requirements are the requirements given directly or indirectly to the designer as inputs. Generally, these requirements encompass the desired range, the desired mass, the desired diameter and eventually, the desired flight time of the SRM. This way, the thrust vs. time graph of the SRM, such as the shown in Figure 4, is determined by considering these requirements inside the flight mechanics. During the determination of the thrust vs. time curve, to each characteristic point (e.g. the action time, the burning time and the ignition delay time shown in Figure 4) a specific value shall be assigned. Consequently, the inputs given to the grain designer are: the thrust-time graph itself, the diameter, the mass, the time and occasionally, the length of the motor.

One of the major variables to be considered in the design of a SRM is the burning rate. Many factors affect the burning rate. The composition of the propellant is the most important fact. Nonetheless, the composition tends to be predetermined and it is the same throughout the entire propellant mass. So, as the propellant properties are experimentally determined, the burn rate law can be predicted. Note that the main conditions affecting the burn rate are: the pressure in the combustion chamber, the initial temperature of the propellant, the gas flow along the burning surface (known as erosive burning) and the motion of the rocket (e.g. fast spinning).

Experimental testing of propellants provides burn rate's dependence on pressure. In case of SRM, the dependence is expressed through the simplified result known as the Saint-Robert's or Vieille's law. Its expression is:

$$\dot{r}_b = aP_c^n \tag{2.1}$$

where P_c is the combustion chamber pressure expressed in Pa, n is the exponent of the chamber pressure which is experimentally adjusted and a is the constant multiplying the value P_c^n. Note that the constant a is determined experimentally; so that its units shall be adjusted in such a way that when P_c is expressed in Pa, the burning rate $\dot{r}_b$ is obtained in m/s.

In case of temperature, it affects the rate at which chemical reactions take place; consequently, the initial propellant temperature affects the burning rate. This way, it is common to place the rocket in a temperature controlled space or at least, to protect from the sun prior to ignition. Furthermore, the initial temperature requirements are determined; so that, if operating conditions exceed the limits, the launch of the rocket is delayed.

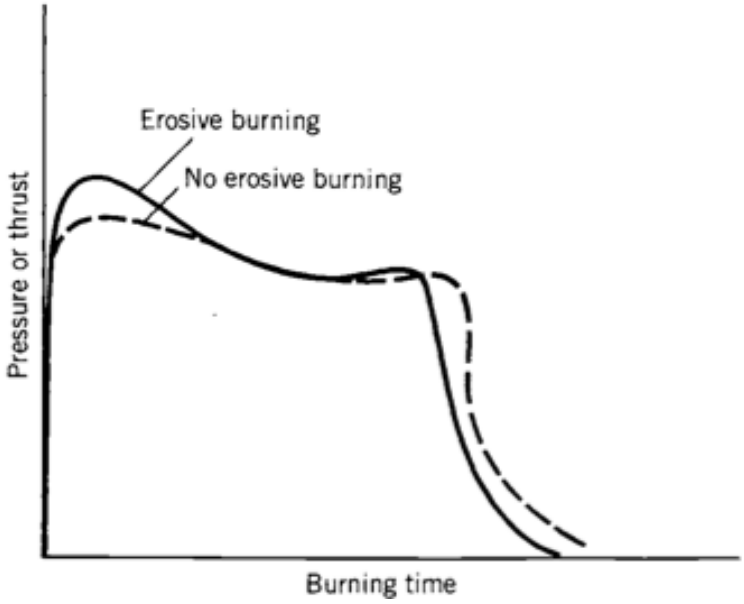

Figure 5. Pressure or thrust curve vs. burning time when erosive burning is considered or not (source [11]).

Erosive burning consists of the increase in burning rate due to the fast flow of hot gases along the burning surface. It mainly exists near the nozzle, where gas flow is fastest and it is mainly a consequence of the pressure gradient within the combustion chamber. This, as it has been shown in Figure 5, results in an irregular burning along different sections of the grain, being strongest when the flow is fastest. The erosive burning affects most to the SRM with a big L/D ratio

(Grain length/Motor case diameter). This fact makes the prediction of the thrust vs. time profile more complex.

Finally, the equations that define the performance of SRM should be specified. For that, the nomenclature shown in Figure 6 is followed. Precisely, the subscript "c" makes reference to the combustion chamber, the subscript "t" makes reference to the throat of the nozzle and the subscript "e" makes reference to the exhaust section of the nozzle. Additionally, the atmosphere outside the SRM is named with the subscript "a". Furthermore, while describing the performance of SRM five parameters have to be considered: the static pressure usually identified as "P", the gas density known as "ρ", the static temperature identified as "T", the cross-section area known as "A" and gas velocity named as "u".

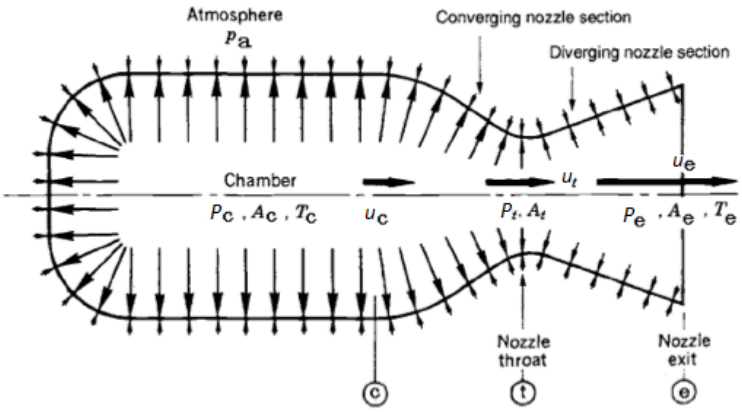

Figure 6. Scheme of the combustion chamber and the rocket nozzle.

Once the performance main parameters and the subscript letters implications are defined, following the scheme of Figure 6, the thrust F in Newton given by a SRM is defined as:

$$F = \dot{m}_d u_e + (P_e - P_a)A_e \tag{2.2}$$

where $\dot{m}_d$ is the mass flow rate exiting the nozzle expressed in kg/m^3, u_e is the exit velocity of the gases expressed in m/s, P_e static pressure of the gases at the exit section of the nozzle in Pa, P_a is the atmospheric pressure in Pa and A_e is the exit cross-section area of the nozzle in m^2. Additionally, when the nozzle is adapted, the static pressure of the gases at the exit section of nozzle is equal to the atmospheric pressure ($P_e = P_a$). Then, the characteristic thrust F_c is obtained, which is equal to:

$$P_e = P_a \text{ (adapted nozzle)} \rightarrow F_c = \dot{m}_d u_e \tag{2.3}$$

where the different parameters have been indicated before. Note that the thrust is maximum when the nozzle is adapted.

Furthermore, when applying the first law of the thermodynamics to a unit of fluid, it is possible to ascertain that:

$$\frac{u_c^2}{2} + h_c = \frac{u_e^2}{2} + h_e \tag{2.4}$$

where u_c is the velocity of the gas at the combustion chamber in m/s, h_0 is the enthalpy of the reactants at the initial temperature of the chamber in J/kg and h_e is the enthalpy of the products at the ejection temperature in J/kg. In case of a solid rocket motor, the velocity at the combustion chamber can be considered to be almost zero. Therefore, if $u_c \approx 0$, from the equation (2.4), u_e can be isolated as:

$$h_c = \frac{u_e^2}{2} + h_e \rightarrow u_e = \sqrt{2(h_c - h_e)} \tag{2.5}$$

Although the previous expression (2.5) is strictly rigorous, it possesses an important drawback: a priori, it is very difficult to directly estimate the enthalpy of the gasses at the exit of the nozzle. Precisely, it is hard to ascertain which is their temperature and composition at the exit of the nozzle. This way, when the computational calculation methods were not very popular among the SRM scientific communities, expressions that allowed to obtain approximated equations for the velocity of the gas at the exit of the nozzle, u_e, were developed. These equations are based on the admission of the following premises:

- The gases perform ideally and they can be characterized through a simple equation of state such as:

$$P \cdot V = \frac{m}{\overline{M}} R_0 T \quad \text{with} \quad \frac{P \cdot V}{T} = \text{constant} \tag{2.6}$$

  where P is the static pressure in Pa, V is the volume of the gas in m^3, m is the mass of the gas in kg, $\overline{M}$ is the mean molar mass of the gas in kg/mol, R_0 is the universal gas constant equal to $R_0 = 8.3144621 \frac{J}{mol\,K}$ and T is the static temperature.
- The expansion that takes place in the nozzle follows the expression:

$$P \cdot V^{\gamma} = \text{constant} \tag{2.7}$$

  where γ is the heat specific value.

This way, the enthalpic difference of expression (2.5) can be approximated through:

$$h_c - h_e = \overline{c}_p (T_c - T_e) = \overline{c}_p T_c \left(1 - \frac{T_e}{T_c}\right) \tag{2.8}$$

where $\bar{c}_p$ is the mean specific heat capacity at constant pressure in J/kg/K, T_c is the combustion chamber static temperature in K and T_e is the static temperature in the exit of the nozzle in K. Additionally, assuming that the fluid does not react in the nozzle and the expansion in the nozzle is adiabatic, it can be used that:

$$\frac{T_e}{T_c} = \left(\frac{P_e}{P_c}\right)^{\frac{\gamma-1}{\gamma}} \tag{2.9}$$

where P_c is the static pressure at the combustion chamber expressed in Pa and P_e is the static pressure at the exit of the nozzle in Pa. Finally, considering the relation of Mayer $\left(c_p - c_v = R_0/\bar{M}\right)$ and the ratio of specific heat capacities $\frac{c_p}{c_v} = \gamma$, if the expression (2.9) is substituted into (2.8) and then, the obtained value in (2.5), the following formula for the exit velocity can be calculated:

$$u_e = \sqrt{\frac{2\gamma}{\gamma-1} R T_c \left[1 - \left(\frac{P_e}{P_c}\right)^{\frac{\gamma-1}{\gamma}}\right]} \tag{2.10}$$

This equation is known as the equation of Saint Venant-Wantzel. Moreover, in order to study and compare the performance of different SRM, it is recommended to use the thrust coefficient C_F, which is defined as:

$$C_F = \sqrt{\gamma}\left(\frac{2}{\gamma+1}\right)^{\frac{\gamma+1}{2(\gamma-1)}} \sqrt{\frac{2\gamma}{\gamma-1}\left[1 - \left(\frac{P_e}{P_c}\right)^{\frac{\gamma-1}{\gamma}}\right]} + \frac{A_e}{A_t}\left(\frac{P_e}{P_c} - \frac{P_a}{P_c}\right) \tag{2.11}$$

where A_t is the area of the throat of the nozzle in m^2. Note that the parameter Γ is defined as:

$$\Gamma = \sqrt{\gamma}\left(\frac{2}{\gamma+1}\right)^{\frac{\gamma+1}{2(\gamma-1)}} \tag{2.12}$$

Consequently, considering the value given for the parameter Γ (2.12), the equation (2.11) can be rewritten as:

$$C_F = \Gamma \sqrt{\frac{2\gamma}{\gamma-1}\left[1 - \left(\frac{P_e}{P_c}\right)^{\frac{\gamma-1}{\gamma}}\right]} + \frac{A_e}{A_t}\left(\frac{P_e}{P_c} - \frac{P_a}{P_c}\right) \tag{2.13}$$

In addition, one of the interesting advantages of the thrust coefficient is that enables to write the expression (2.2) of the thrust as:

$$F = C_F P_c A_t \tag{2.14}$$

Consequently, while using the thrust coefficient through the expression (2.14), the thrust can be related directly with the combustion chamber pressure and with the nozzle throat area. Another helpful parameter for the analysis of the SRM performance is the characteristic velocity c^*, which is defined as:

$$c^* = \frac{\sqrt{RT_c}}{\Gamma} = \frac{\sqrt{\frac{R_0}{M}T_c}}{\Gamma} \tag{2.15}$$

Note that the units of the characteristic velocity are m/s. The characteristic velocity can be related with the mass flow rate exiting the nozzle through:

$$\dot{m}_d = \frac{P_c A_t}{c^*} \tag{2.16}$$

Moreover, the nozzle expansion ratio ε is defined as:

$$\varepsilon = \frac{A_e}{A_t} \tag{2.17}$$

From the theory of the one-dimensional steady nozzle flow, the relationship between the nozzle expansion ratio and the nozzle pressure ratio P_e/P_c, considering the expression (2.12), can be written as:

$$\varepsilon = \frac{A_e}{A_t} = \frac{\sqrt{\gamma}\left(\frac{2}{\gamma+1}\right)^{\frac{\gamma+1}{2(\gamma-1)}}}{\left(\frac{P_e}{P_c}\right)^{\frac{1}{\gamma}}\sqrt{\frac{2\gamma}{\gamma-1}\left[1-\left(\frac{P_e}{P_c}\right)^{\frac{\gamma-1}{\gamma}}\right]}} = \frac{\Gamma}{\left(\frac{P_e}{P_c}\right)^{\frac{1}{\gamma}}\sqrt{\frac{2\gamma}{\gamma-1}\left[1-\left(\frac{P_e}{P_c}\right)^{\frac{\gamma-1}{\gamma}}\right]}} \tag{2.18}$$

To sum up, the last important SRM parameter to be defined is the specific impulse, also known as I_s. Indeed, the specific impulse is a measure of the impulse or momentum change that can be produced per unit of mass of the propellant consumed. In terms of grain design, the I_s is the ratio of the motor thrust to mass flow rate. Thereby, its value is most significant in the determination of the propellant weight necessary to meet the mission requirements. Generally, the specific impulse, expressed in m/s, is defined as:

$$I_s = c^* C_F \tag{2.19}$$

where the term c^* is defined in equation (2.15) and the term C_F in (2.13).

3. STATE OF THE ART

In solid rocket motors, the grain burnback analysis consists of calculating the evolution of the grain burning surface during the combustion process of the rocket. This way, the problem to solve in the grain burnback analysis consists of predicting the evolution of an interface (grain burning surface) subject to a given propagation velocity field. This velocity field depends on space, time and the normal direction of the boundary. In solid propellant combustion, the grain surface recedes normal to itself, as it is shown in Figure 7.

Figure 7. Process of grain burnback of a 2D grain design (source [11])

Figure 7 shows how the grain burning surface evolves during the operation of a SRM with a 2D grain design. To illustrate the problem to solve in the prediction of the grain burning surface evolution, Figure 8 shows an arbitrary 2D interface curve that propagates normal to itself with a speed F.

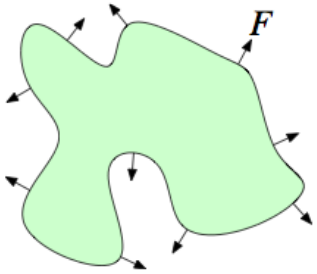

Figure 8. Interface propagating normal to itself with a speed F (source [7]).

The problem of the front propagation is stated as follows (see [7]):

$$\frac{d\vec{x}}{dt} = F \cdot \vec{n}(\vec{x}) \tag{3.1}$$

Which is also known as the Eikonal's equation, given the initial interface curve Γ. Where $\vec{x}$ is the position vector of the points that form the interface, F is the

propagation speed and $\vec{n}(\vec{x})$ is the local normal vector to the interface at each point of the interface. The initial condition is stated as follows:

$$\vec{x}(t = 0) = \Gamma \tag{3.2}$$

This way, this problem can be solved using many methodologies. Moreover, these methods can be divided mainly into two categories: explicit and implicit techniques. The explicit techniques are methods that represent the interface explicitly through a parameterized function or expression. As well, among the explicit methods there are the analytical methods, which define the interface shape through an analytical function, and the numerical methods, which represent the interface by discretizing it in a finite number of points. On the contrary, the implicit methods are characterized for representing the interface implicitly through a given function. In these latter methods, there is no direct expression that defines the interface, but it is necessary to reconstruct the interface by finding a given level from the implicit function.

3.1. EXPLICIT TECHNIQUES

The explicit techniques are based on parameterizing the interface (analytically or numerically) and integrating the system of ODE's (3.1) departing from the initial condition (3.2). For instance, one of the most used numerical methods is the marker particle method. This technique is based on parameterizing the discretized interface through marker points that for instance are equally spaced. Then, the position of each marker point is propagated using the equations of motion (3.1). However, as it is stated in [3] and [7], this explicit method has some important drawbacks, which are shown in Figure 9. This method fails when representing sharp corners, which indeed form very usually in SRM grain burnback process. As well, this method fails when the interface topology changes unless many exceptions are considered in the programming. Nevertheless, despite of the exceptions many problems arise when the topology of the interface changes. As well, after some propagation time the nodes distribution becomes uneven, thereby representing wrongly the interface. Thus, this method is not suitable for complex motions such as the case of grain burnback in solid rocket motors.

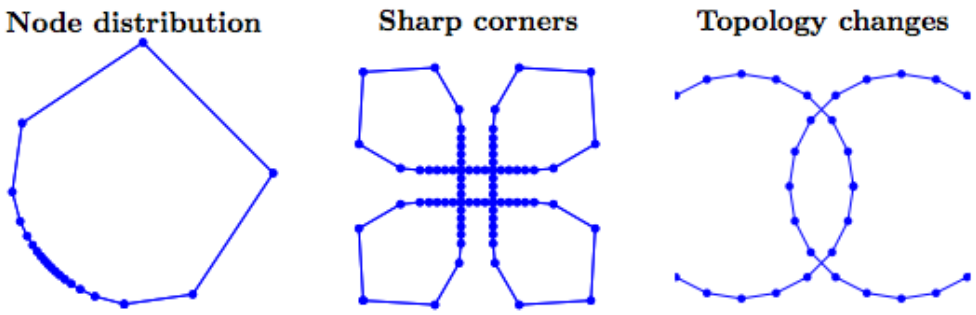

Figure 9. Typical problems from explicit techniques (source [7]).

3.2. IMPLICIT TECHNIQUES

On the other side, in the implicit techniques the interface is represented implicitly using a given function. The most common and powerful techniques used to date are the Level Set Method and the Fast Marching Method. Both methods were developed by Sethian and Osher [3];[5], as an innovative way to solve the problem of evolving fronts.

The idea of the level set method is to represent the interface implicitly through the zero level set of a $\phi(\vec{x}(t), t)$ function. Basically, this function represents the distance of a point $\vec{x}$ to the interface. Hence, the points where $\phi(\vec{x}(t), t) = 0$ represent the interface. It is important to remark that this method is a Initial Value Problem. One of the great advantages of the Level Set Method, is that it can afford the formation of cusps and corners, even though the differentiability is lost. As well, it can handle easily topology changes as it is shown in Figure 11.

On the other hand, the fast marching method also represents implicitly the interface through a $T(\vec{x})$ function. This function represents the time of arrival of the interface to a given studied point. It should be pointed out that the fast marching method is a Boundary Value Problem, in contrast with the level set method. The philosophy of this method is shown in Figure 10.

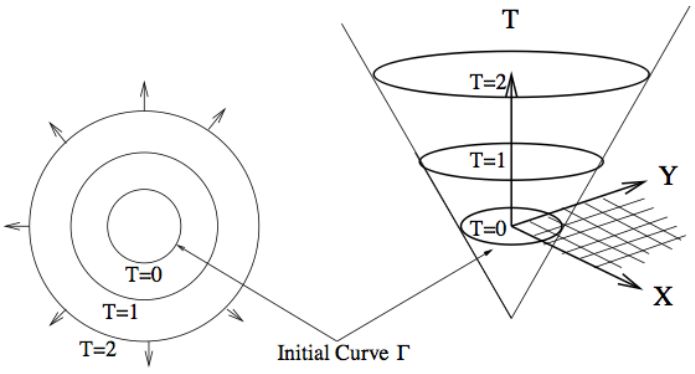

Figure 10. Representation of the Fast Marching Method (source [4]).

4. GRAIN BURNBACK ANALYSIS

This chapter develops the 2D and Quasi-3D grain burnback analysis formulation, modeling and algorithms applied in this study. Although this project deals both with the simulation of the grain burnback analysis and the internal ballistics of the SRM, the core of the project is the first one.

Following, a brief organization of the chapter is presented. Firstly, an introduction of the relation between the internal ballistics, grain burning surface evolution and SRM performance is exposed. Next, the Level Set Method is introduced, developing its formulation and numerical scheme for this application. Later on, the section is divided into the modeling of 2D and Quasi-3D grain burnback analysis. In the 2D section, the necessary steps and calculations required to perform the grain regression simulation are explained. Finally, the Quasi-3D burnback analysis is presented, accounting also for the common steps shared with the 2D simulation.

4.1. INTRODUCTION TO GRAIN BURNBACK ANALYSIS

The grain burnback analysis consists of calculating the evolution of the grain burning surface during the operation of a solid rocket motor. Due to the combustion process, the grain burning surface recedes, propagating itself towards the inward normal direction. In this way, the propellant below the grain burning surface layer, which is in solid state, burns continually becoming a hot gas that flows across the combustion chamber and finally escapes through the rocket nozzle.

As it has been aforementioned, the evolution of the grain burning surface has a determining influence on the performance of a solid rocket motor. Actually, the SRM performance is linked to the grain burnback analysis through the internal ballistics simulation (internal flow in the SRM). In fact, the internal ballistics simulation requires inputs from the grain burnback analysis such as the burning area, port area, chamber volume, inter-alia. Yet, this relation is reciprocal because in turn, the propagation speed of the burning surface is determined by the internal flow of the SRM. Thus, the grain burnback analysis is coupled with the internal ballistics simulation, as it is explained in the section 6.1. As a concluding remark, in order to predict the SRM performance it is necessary to perform a precise grain burnback analysis.

From the mathematical perspective, the problem to solve in the grain burnback analysis consists of predicting the evolution of an interface (grain burning surface) subject to a given propagation velocity field. The velocity field, called

burning rate in SRM terminology, depends on space, time and the normal direction of the boundary. Moreover, this dependence on space and time comes from the coupling with the internal ballistics simulation, which uses the interface geometrical parameters. For instance, the burning rate depends on the combustion chamber pressure, while this latter depends in turn on the burning surface area.

4.2. LEVEL SET METHOD

After considering several methods for performing the grain burnback analysis, it has been chosen to use the level set method (LSM). This method was invented by J.A. Sethian and S. Osher in 1988 (original paper [1]), and since then it has been applied to many fields such as crystal growth, flame propagation, grain burnback analysis [8],[9], shortest paths calculation, image segmentation, inter-alia.

The level set method is a numerical algorithm very suitable for solving problems related with the propagation of fronts. Precisely, this method is able to predict the motion of a boundary or interface given the propagation velocity field. The motion considered in the LSM is a front propagation along its normal vector field allowing the speed to depend on the curvature.

The main advantage of the LSM lies on its ability to easily handle sharp gradients and shape cusps, as well as topological changes in the interface. Moreover, the characteristic feature of the LSM consists of representing the interface implicitly, by using the zero level set of a time-dependent function, ϕ. However, this is an opposite approach to that used for older algorithms, which in many cases were based on a parametrization of the discretized interface. The philosophy of those algorithms was to use marker points of the discretized interface and then propagate it using the equations of motion. Nevertheless, these explicit methods proved to be unsuitable for complex motions such as the case of grain burnback in solid rocket motors.

It should be pointed out that the level set method formulation is Eulerian while the approach of those older algorithms is Lagrangian. Thus, in the Lagrangian formulation the equations follow the interface, while on the Eulerian formulation the equations are “fixed” on the frame and the interface is tracked implicitly. This difference on the formulation philosophy is the main reason why the level set method works were the explicit methods fail. Furthermore, as it is stated in [5]: *“To summarize, Lagrangian approximations suffer instability and topological limitations because they follow a local representation of the front”*. Indeed, the Lagrangian approach has the same problem experienced by the Burgers’ equation without the viscosity term. Namely, in the inviscid Burgers’ equation the

solution can develop discontinuities (shock waves) even if the initial data is smooth, just like it might happen in the explicit methods. To fix this problem, in the Burgers' equation an entropy condition is used to select the proper way to continue the solution after the shock is formed. In the same way, the level set method formulates an entropy condition [6] to provide an explicit construction for a weak solution beyond the formation of discontinuities (blow up). Thanks to this entropy condition, the level set method can afford the formation of cusps and corners, even though the differentiability is lost. Thereby, this ability of the LSM becomes a key factor, bearing in mind that in many cases cusps form during the grain burnback evolution of a SRM. Summarizing, all these features make the level set method very appropriate for performing the grain burnback analysis of SRM's.

In the construction of the level set method, the entropy condition plays a very important role, as it is exposed next. Basically, the entropy condition consists of stating the following: "*once a particle burns, it remains burnt"*. This way, *"if two ignition curves cross at a particular point, whichever one arrives first will ignite the particle located there"* (both citations come from [6]). Therefore, from this entropy condition the ϕ function is created as an indicator function of the propagating interface, from which the boundary can be determined. Thereby, as it has been stated before, in the level set method rather than following the interface through an explicit way, the front interface is represented implicitly using the 0 level set of the function ϕ:

$$\phi(\vec{x}(t), t) = 0 \tag{4.1}$$

The equation (4.1) expresses that for any time the interface points are defined by the zero level set of the ϕ function. Note that ϕ is defined for all x, not just for the points of the boundary. A scheme of the level set method representation is shown in Figure 11. This example illustrates the LSM philosophy for the case of the propagation of a circle subject to a propagating speed normal to the boundary (not dependent on the curvature of the boundary). This figure shows that for each time step, ϕ is a surface in which its zero level set is the interface boundary, namely the circle at each time step.

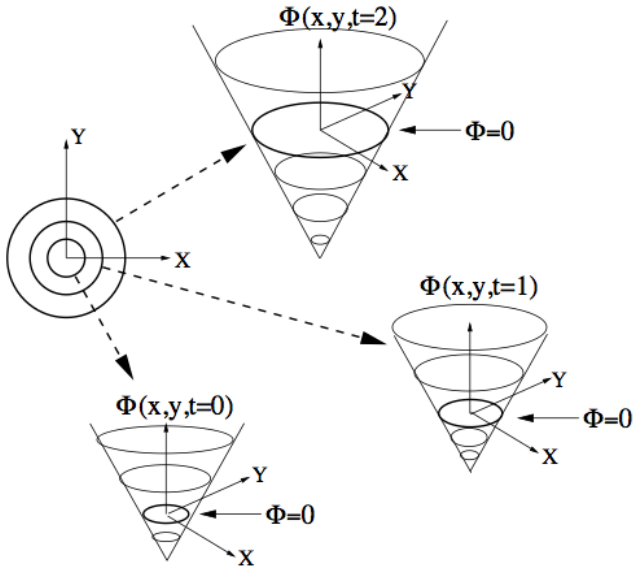

Figure 11. The level set method representation for the propagation of a circle (source [4]).

4.2.1. GOVERNING EQUATION

Recalling (4.1), in the level set method, at any time, the front is given by the zero level set of the ϕ function (for more detailed formulation development see [4]). Differentiating (4.1) with respect to time an applying the chain rule:

$$\frac{\partial \phi}{\partial t} + \nabla \phi \cdot \frac{\partial \vec{x}}{\partial t} = 0 \tag{4.2}$$

On the other side, let F be the propagation speed of the front in the normal outward direction:

$$\frac{d\vec{x}}{dt} \cdot \vec{n} = F \Leftrightarrow \frac{d\vec{x}}{dt} = F \cdot \vec{n} \tag{4.3}$$

Due to ϕ is constant at a level set, ∇ϕ is normal to the contour given by the points defined by that level set. So, the normal direction is:

$$\vec{n} = \frac{\nabla \phi}{|\nabla \phi|} \tag{4.4}$$

Substituting (4.3) and (4.4) into (4.2) we obtain:

$$\frac{\partial \phi}{\partial t} + \nabla \phi \cdot F \cdot \frac{\nabla \phi}{|\nabla \phi|} = 0 \tag{4.5}$$

Given that $\nabla \phi \cdot \nabla \phi = |\nabla \phi|^2$, the level set equation is obtained:

$$\frac{\partial \phi}{\partial t} + F\,|\nabla \phi| = 0 \tag{4.6}$$

The initial condition is:

$$\text{Given} \qquad \phi(\vec{x}, t = 0) \tag{4.7}$$

Hence, the initial condition consists of knowing the values of ϕ for all x at $t = 0$. So that, departing from the initial condition the interface can be propagated by solving equation (4.6) for a given velocity F. Note that all level sets are propagated, not only the zero level set. Additionally, the interface can be tracked by extracting the points given by the zero level set for each time step.

4.2.2. DISCRETIZATION AND NUMERICAL SCHEME

Bearing in mind that ϕ is defined for all $\vec{x}$, in order to represent it numerically on a computer, it is discretized by using a background mesh. In this study, a Cartesian grid is used for this purpose but other types of mesh can be used.

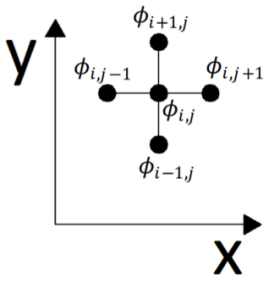

Figure 12. Values of the discretized ϕ function at the grid points surrounding an i,j-th point

In the Matlab® code developed in this study for the 2D grain burnback analysis, the matrix containing the values of the discretized ϕ function ($\phi(x, y, t)$), uses the index "j" for the x-direction, the index "i" for the y-direction and the index "n" for the time. Where "i" is the index for the matrix rows, "j" is the index for the matrix columns and "n" is the index for the matrix third dimension. The discretized ϕ function matrix has the dimensions $(N_{grid} + 1) \cdot (N_{grid} + 1) \cdot (N_{time_{steps}} + 1)$. Where $(N_{grid} + 1)$ is the number of nodes per side, in x and y direction, and $(N_{time\_steps} + 1)$ is the number of time steps in the simulation. In Figure 13 an example of the ϕ function discretized on a Cartesian grid is shown.

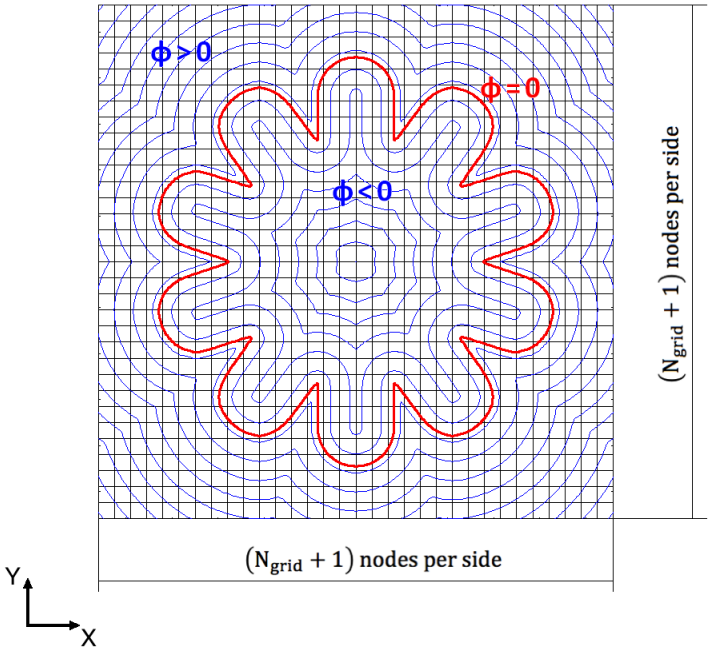

Figure 13. ϕ function discretized on a Cartesian grid.

Note that in Figure 13 the regions are delimited as follows:

- $\phi > 0$: the grid points are outside the interface delimited region. In the case of grain burnback analysis this region is the propellant side.
- $\phi < 0$: the grid points are inside the interface delimited region. In the case of grain burnback analysis this region is the hot gas side.

It should be pointed out that it is very unlikely to happen that a grid point has an exact value of $\phi = 0$. Hence, in order to extract the interface from the discretized ϕ function, one can interpolate between the cells as it is explained in the section 4.3.3.

On the other side, the numerical scheme of the LSM should satisfy the entropy condition as well as to give the physical correct solution despite of the fact that this solution may not be differentiable. Hence, the numerical scheme should select the entropy-satisfying weak solution among the viscosity solutions from the associated partial differential equations. To achieve this, the level set method uses techniques from the numerical solutions of hyperbolic conservation laws. To illustrate the relation of the level set equation and the hyperbolic conservation laws, consider the one-dimensional level set equation:

$$\phi_t + F \cdot |\phi_x| = 0 \Leftrightarrow \phi_t + F \cdot \sqrt{(\phi_x)^2} \tag{4.8}$$

Considering now a scalar conservation law equation in conservative form:

$$u_t + [f(u)]_x = 0 \tag{4.9}$$

On the other side, the one-dimensional Hamilton–Jacobi equation for hyperbolic conservation law:

$$g_t + [H(g)]_x = 0 \Rightarrow f(u) = H(g) \tag{4.10}$$

Where H is the Hamiltonian. Then by substituting $g = \phi_x$ in (4.13) we obtain:

$$(\phi_x)_t + [H(\phi_x)]_x = 0 \tag{4.11}$$

So, from these equations it is clear that the level set equation is related with the hyperbolic conservation laws. Hence, in the level set method it is possible to use known numerical schemes for solving hyperbolic conservation laws.

Finally, the numerical scheme used in this study is a first order upwind scheme (further information about the construction of this numerical scheme can be found in [4]):

$$\phi_{i,j,k}^{n+1} = \phi_{i,j,k}^{n} - \Delta t^{n}\left(\max\left(F_{i,j,k}, 0\right) \cdot \nabla_{i,j,k}^{+} + \min\left(F_{i,j,k}, 0\right) \cdot \nabla_{i,j,k}^{-}\right) \tag{4.12}$$

Where $F_{i,j,k}$ is the interface propagation speed. As well, $\nabla_{i,j,k}^{+}$ and $\nabla_{i,j,k}^{-}$ are given by:

$$\begin{aligned}\nabla_{i,j,k}^{+} = \Big[&\max\left(D_{i,j,k}{}^{-x}\phi_{i,j,k}^{n}, 0\right)^2 + \min\left(D_{i,j,k}{}^{+x}\phi_{i,j,k}^{n}, 0\right)^2 \\ &+ \max\left(D_{i,j,k}{}^{-y}\phi_{i,j,k}^{n}, 0\right)^2 + \min\left(D_{i,j,k}{}^{+y}\phi_{i,j,k}^{n}, 0\right)^2 \\ &+ \max\left(D_{i,j,k}{}^{-z}\phi_{i,j,k}^{n}, 0\right)^2 + \min\left(D_{i,j,k}{}^{+z}\phi_{i,j,k}^{n}, 0\right)^2\Big]^{1/2}\end{aligned} \tag{4.13}$$

$$\begin{aligned}\nabla_{i,j,k}^{-} = \Big[&\min\left(D_{i,j,k}{}^{-x}\phi_{i,j,k}^{n}, 0\right)^2 + \max\left(D_{i,j,k}{}^{+x}\phi_{i,j,k}^{n}, 0\right)^2 \\ &+ \min\left(D_{i,j,k}{}^{-y}\phi_{i,j,k}^{n}, 0\right)^2 + \max\left(D_{i,j,k}{}^{+y}\phi_{i,j,k}^{n}, 0\right)^2 \\ &+ \min\left(D_{i,j,k}{}^{-z}\phi_{i,j,k}^{n}, 0\right)^2 + \max\left(D_{i,j,k}{}^{+z}\phi_{i,j,k}^{n}, 0\right)^2\Big]^{1/2}\end{aligned} \tag{4.14}$$

Moreover, $D_{i,j,k}{}^{-x}$ and $D_{i,j,k}{}^{+x}$ are respectively the backward and forward difference operators in the x-direction:

$$D_{i,j,k}{}^{-x}\phi_{i,j,k}^{n} = \frac{\phi_{i,j,k}^{n} - \phi_{i,j-1,k}^{n}}{\Delta x} \tag{4.15}$$

$$D_{i,j,k}{}^{+x}\phi_{i,j,k}^{n} = \frac{\phi_{i,j+1,k}^{n} - \phi_{i,j,k}^{n}}{\Delta x} \tag{4.16}$$

As it has been stated before, in the Matlab® code developed in this study, the matrix containing the values of the discretized ϕ function, uses the index "j" for the x-direction and the index "i" for the y-direction. The formulation here presented follows the nomenclature applied in the code. Analogously, the backward and forward difference operators in the y-direction are:

$$D_{i,j,k}{}^{-y}\phi^{n}_{i,j,k} = \frac{\phi^{n}_{i,j,k} - \phi^{n}_{i-1,j,k}}{\Delta y} \tag{4.17}$$

$$D_{i,j,k}{}^{+y}\phi^{n}_{i,j,k} = \frac{\phi^{n}_{i+1,j,k} - \phi^{n}_{i,j,k}}{\Delta y} \tag{4.18}$$

Finally, the backward and forward difference operators in the z-direction are:

$$D_{i,j,k}{}^{-z}\phi^{n}_{i,j,k} = \frac{\phi^{n}_{i,j,k} - \phi^{n}_{i,j,k-1}}{\Delta z} \tag{4.19}$$

$$D_{i,j,k}{}^{+z}\phi^{n}_{i,j,k} = \frac{\phi^{n}_{i,j,k+1} - \phi^{n}_{i,j,k}}{\Delta z} \tag{4.20}$$

4.3. 2D GRAIN BURNBACK ANALYSIS

The solid rocket motor grain designs can be divided into two categories: 2D and 3D. This distinction refers to the number of dimensions that determine the grain regression evolution. In 2D grains, the cross-section is constant along the motor length while on 3D grains the cross-section varies along the motor longitudinal axis (z). Thus, in 2D grains, studying only the evolution of the grain cross-section in 2 dimensions (x-y) is enough to determine the whole grain regression. In contrast, in 3D grains it is necessary to study the 3 dimensions of the grain geometry to perform the grain burnback analysis. At this point, it should be pointed out that the complexity involved in performing the grain burnback analysis in 3D is considerably higher than doing so in 2D. Hence, whenever possible, 2D grains are preferred for their simplicity and relative easiness of calculation. Figure 14 shows a typical 2D grain design, namely a star-shaped grain, which is very commonly used due to its neutral thrust curve.

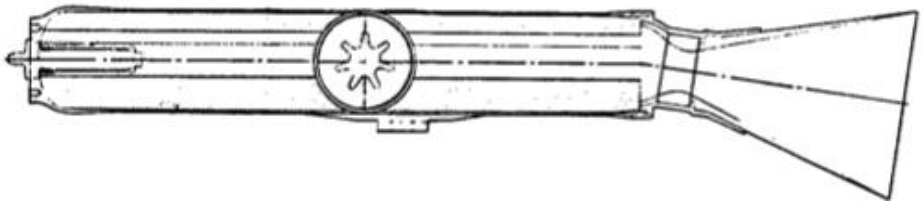

Figure 14. Solid rocket motor with a 2D star grain design (source [10])

On the other side, due to the 2D grain burnback analysis deals with the grain only in 2 dimensions (x and y), the interface to propagate is a 2D curve. Thereby, in order to obtain the grain burning surface, one only needs to multiply the interface perimeter by the grain length (considering inhibited grain ends (4.35)). Thus, in 2D grains the interface perimeter and the burning surface evolve in the same way. So, in 2D grain burnback analysis the interface perimeter is commonly taken as the geometrical indicator for the grain evolution, as it is shown in Figure 15.

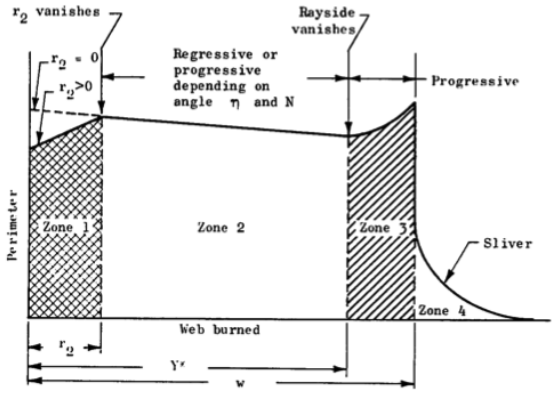

Figure 15. Star configuration evolution of the perimeter vs. web burnt (source [10])

The philosophy of the 2D grain burnback analysis is shown in Figure 16. The grain cross-section is constant along the motor longitudinal axis (z-axis). As well, the interface is a 2D curve (in white color in Figure 16) belonging to the x-y plane. Hence, the 2D grain burnback analysis consists of propagating this 2D curve given a burning rate. At the same time, the burning rate is determined by the internal ballistics simulation, which uses the geometrical properties of the grain such as the burning surface area or the port area. Due to the treatment is 2D, the inner grain surface is constructed by extruding the interface calculated in the x-y plane through the z-axis.

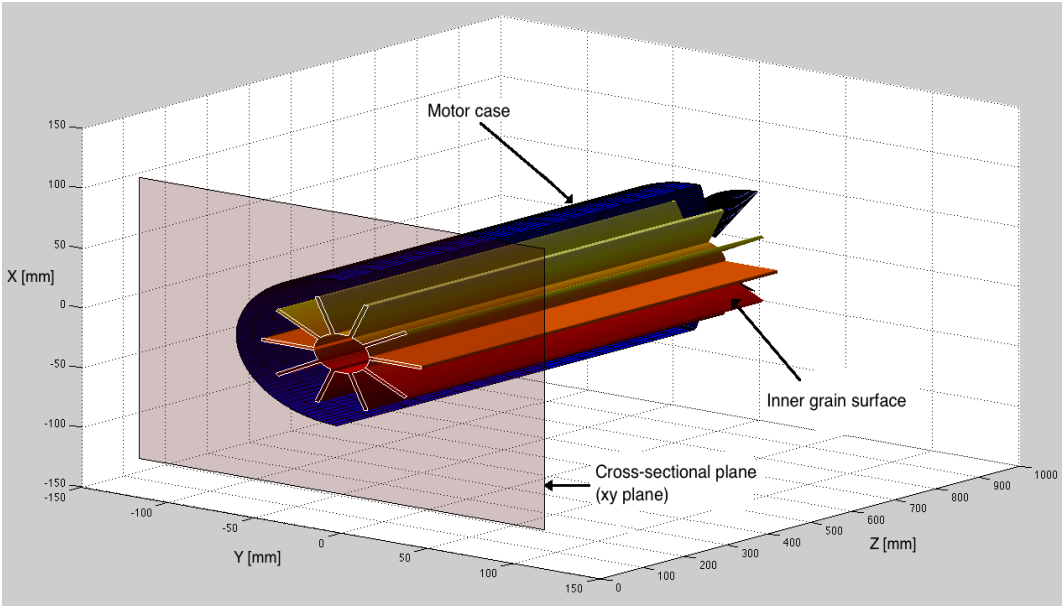

Figure 16. Scheme of a 2D grain configuration (star of 10 slotted tips).

4.3.1. LOCAL NORMAL VECTOR CALCULATION

The initialization of the ϕ function requires distinguishing whether each background grid point is inside or outside the interface. Therefore, in order to make this distinction, it is necessary to calculate the local normal vectors at each point of the discretized interface.

On the other side, it should be pointed out that in the code developed in this study, the user introduces the initial grain geometry through a text file with the coordinates x, y and z of each point. This text file contains the geometry of the initial grain curve as well as the shape of the motor case (points forming a circle of a given radius). Figure 17 is an example of an input geometry file introduced by the user; precisely, it is the geometry file of the grain design of Figure 16.

```
%Star of 10 slotted tips Grain geometry
%       X                Y                Z
  -4.33013e+00   -2.50000e+00    0.00000e+00
  -4.56773e+00   -2.03368e+00    0.00000e+00
  -4.04508e+00   -2.93893e+00    0.00000e+00
  -4.75528e+00   -1.54509e+00    0.00000e+00
  -3.71572e+00   -3.34565e+00    0.00000e+00
  -3.52783e+00   -1.27770e+00    0.00000e+00
  -3.53642e+00   -1.25128e+00    0.00000e+00
  -3.54500e+00   -1.22486e+00    0.00000e+00
  -3.55359e+00   -1.19844e+00    0.00000e+00
  -3.45973e+00   -1.25557e+00    0.00000e+00
  -3.56217e+00   -1.17202e+00    0.00000e+00
  -3.57075e+00   -1.14560e+00    0.00000e+00
  -4.89074e+00   -1.03956e+00    0.00000e+00
  -3.34565e+00   -3.71572e+00    0.00000e+00
  -3.57934e+00   -1.11919e+00    0.00000e+00
  -3.39163e+00   -1.23344e+00    0.00000e+00
  -3.58792e+00   -1.09277e+00    0.00000e+00
  -3.59651e+00   -1.06635e+00    0.00000e+00
  -3.60509e+00   -1.03993e+00    0.00000e+00
  -3.32353e+00   -1.21131e+00    0.00000e+00
  -3.53699e+00   -1.01780e+00    0.00000e+00
  -3.25542e+00   -1.18918e+00    0.00000e+00
  -3.46888e+00   -9.95676e-01    0.00000e+00
  -3.18732e+00   -1.16706e+00    0.00000e+00
  -3.40078e+00   -9.73548e-01    0.00000e+00
  -3.11922e+00   -1.14493e+00    0.00000e+00
  -3.33268e+00   -9.51420e-01    0.00000e+00
  -3.05111e+00   -1.12280e+00    0.00000e+00
```

Figure 17. Example of a geometry text file of a 2D grain design.

Once the code reads this geometry file, it separates the points belonging to the grain geometry and the points that correspond with the motor case, and stores each group of points into two separated matrices. After this, the points are ordered in counter-clockwise sense (this is just the criterion chosen in this study).

In order to calculate the local normal vectors at each point of the discretized interface, it is necessary to compute first the tangent vector. This way, the tangent vector to a curve can be approximated with a central difference approximation as follows [7]:

$$\vec{\tau}_i = \frac{d\vec{x}_i}{ds} \approx \frac{\vec{x}_{i+1} - \vec{x}_{i-1}}{2\Delta s} \tag{4.21}$$

Where "i" is the index of the studied point in the discretized curve. Also, $\vec{x}_i$ is the position vector of the ith point of the discretized curve and Δs is the spacing between the discretized points in the interface. Furthermore, the normal vector can be approximated using the same process (central difference approximation):

$$\vec{n}_i = \frac{d^2\vec{x}_i}{ds^2} \approx \frac{\vec{x}_{i+1} - 2\vec{x}_i + \vec{x}_{i-1}}{\Delta s^2} \tag{4.22}$$

This equation computes the inward normal vector; hence the vector has to be inverted to obtain the outward normal vector. However, this approximation fails when for instance $\vec{x}_{i+1}$, $\vec{x}_i$ and $\vec{x}_{i-1}$ are aligned. As well, this approximation fails when the curvature goes in a different sense to that in which the points are ordered (recall that the points are ordered in counter-clockwise sense). Nevertheless, it is possible to use the tangent vector to construct the normal vector due to both vectors are perpendiculars. Additionally, if the points of the discretized interface are ordered, the tangent vector approximation is always

correct. Then, by changing the tangent vector components in order to rotate it 90º clockwise, the outward normal vector can be obtained:

$$\vec{\tau}_{\mathrm{i}} = \left[\tau_{i_x}\ ,\ \tau_{i_y}\right] \Rightarrow \vec{\mathrm{n}}_{\mathrm{i}} = \left[\tau_{i_y}\ , -\tau_{i_x}\right] \tag{4.23}$$

In Figure 18, an example of the calculation of the local normal (outward direction) vectors to the interface is shown. In this case, the interface is the grain initial geometry of a “dogbone” 2D grain design.

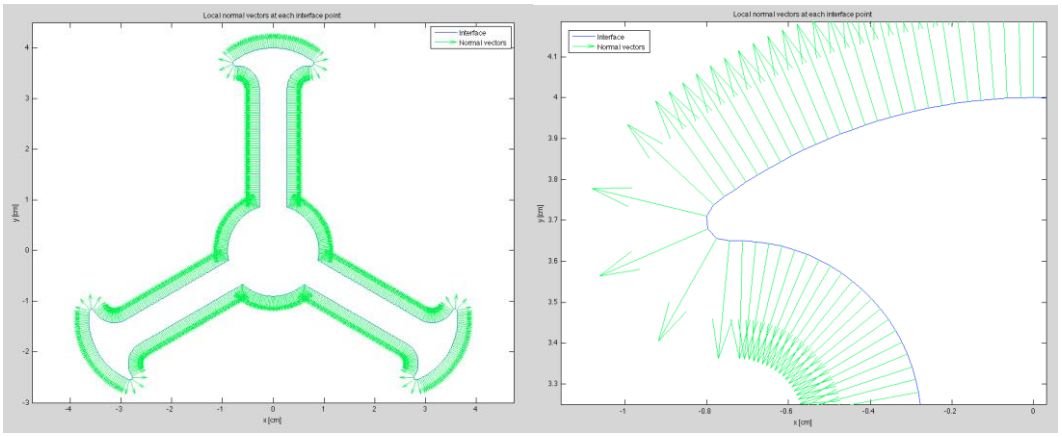

Figure 18. Local normal vectors at each point of the discretized interface.

4.3.2. MINIMUM DISTANCE FUNCTION

In order to initialize the ϕ function, it is necessary to compute the minimum distance function to the interface for each point of the background grid. The so-called minimum distance function is the value of the minimum distance ($\mathrm{d_{min}}$) from a background grid point to the interface. As well, the minimum distance function (MDF) is a signed distance function, as follows:

$$\mathrm{MDF} = \begin{cases} -|\mathrm{d_{min}}| & \textit{if} \quad \text{the background grid point is inside the interface region} \\ 0 & \textit{if} \quad \text{the background grid point coincides with the interface} \\ |\mathrm{d_{min}}| & \textit{if} \quad \text{the background grid point is outside the interface region} \end{cases}$$

The minimum distance of the j-th grid point to the interface ($\mathrm{d^j_{min}}$), is calculated by doing a loop for all the points of the discretized interface (from $\mathrm{k} = 1$ to $\mathrm{k} = \mathrm{N_{interface}}$) and computing (4.24):

$$\mathrm{d_{jk}} = \sqrt{\left(\mathrm{X^j_{grid}} - \mathrm{X^k_{int}}\right)^2 + \left(\mathrm{Y^j_{grid}} - \mathrm{Y^k_{int}}\right)^2} \tag{4.24}$$

Where $\mathrm{X^j_{grid}}$ and $\mathrm{Y^j_{grid}}$ are the coordinates of the studied j-th grid point, and $\mathrm{X^k_{int}}$ and $\mathrm{Y^k_{int}}$ are the coordinates of the k-th point of the discretized interface. Then,

after the loop in k is completed, d^{j}_{min} is obtained ($d^{j}_{min} = \min(d_{jk})$), and the closest interface point to the j-th grid point is identified ($k_{closest}$). It should be remarked that although d^{j}_{min} has been computed, it is necessary to determine whether the studied j-th grid point is inside or outside the interface. To do so, it is necessary to use the normal vectors of the interface, as shown in Figure 19. The condition used to distinguish if the studied grid point is inside or outside the interface is the next:

IN-OUT condition

$$(\vec{P}_{grid} - \vec{P}_{int}) \cdot \vec{n}_{int} \begin{cases} \geq 0 \rightarrow \text{the background grid point is outside the interface region} \\ < 0 \rightarrow \text{the background grid point is inside the interface region} \end{cases}$$

Where $\vec{P}_{grid}$ ($\vec{P}_{grid} = \left[X^{j}_{grid}, Y^{j}_{grid}\right]$) is the position vector of the j-th grid point studied, $\vec{P}_{int}$ ($\vec{P}_{int} = \left[X^{k_{closest}}_{int}, X^{k_{closest}}_{int}\right]$) is the position vector of the closest interface point to the grid point studied, and $\vec{n}_{int}$ is the local normal vector at this closest interface point. Then, applying the IN-OUT condition the sign of the MDF for the j-th studied grid point is determined.

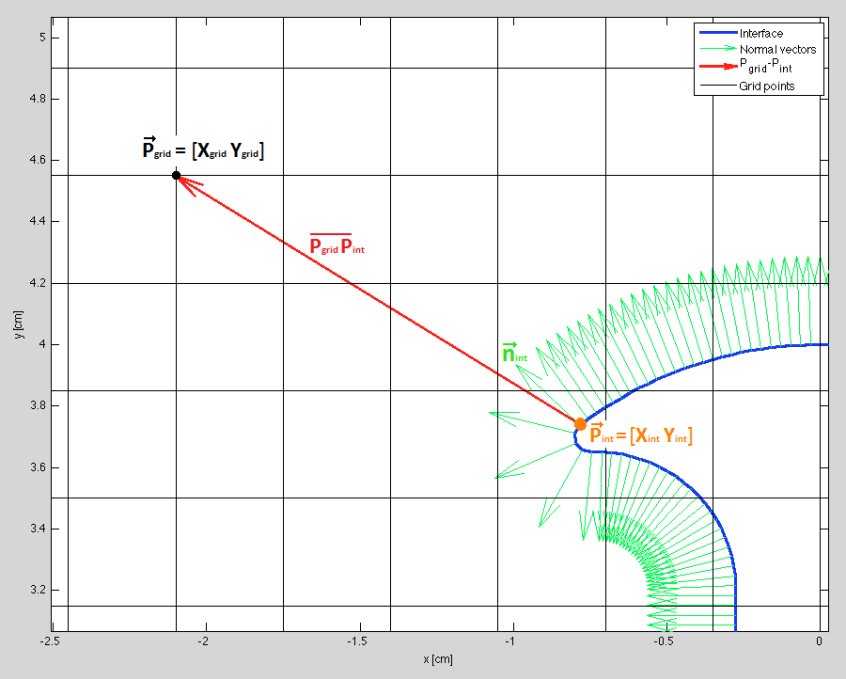

Figure 19. Graphical example of the computation of the minimum distance function.

These operations are repeated in a j loop for all the background grid points. Therefore, once the MDF has been computed for all the background grid points, the discretized ϕ function is initialized:

$$\phi(x, y, t = 0) = \mathrm{MDF}(x, y) \qquad \forall x, y \in \text{Background grid} \tag{4.25}$$

This way, the initial value problem stated by the level set method can be propagated departing from this initial condition. This initial condition is the numerical equivalence of (4.7). It should be pointed out that the initialization of ϕ function is a very time-consuming operation for the computer.

An example of the MDF result for a given 2D grain design (initial grain geometry) is shown in Figure 20 and Figure 21. Recall how this result coincides with Figure 11 (for $t = 0$), which is a representation of the level set method.

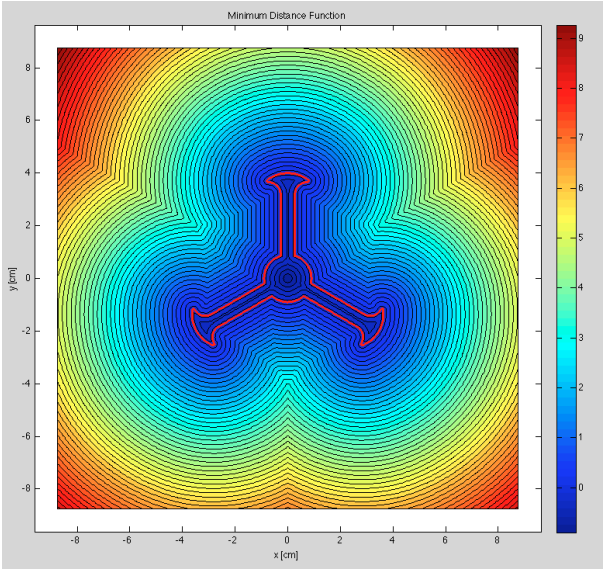

Figure 20. Contour plot of the MDF for a given 2D grain design (dogbone type).

As it is shown in Figure 20 and Figure 21, the grid points inside the interface (red line shape) have a negative MDF value, and so their color is deep blue. On the other side, the grid points outside the interface have positive values of MDF and their colors go from blue to red. This example has been done using a background grid of 250x250 points, which for this example provides a resolution good enough to track the geometry complexities.

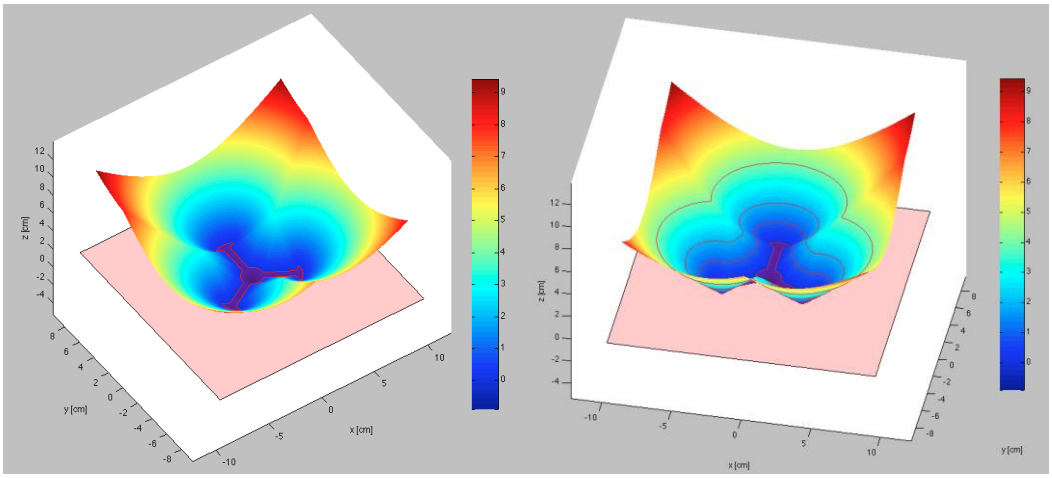

Figure 21. 3D plot of the surface defined by the MDF for a given 2D grain design (dogbone type)

In a 2D case the ϕ function is a 3D surface as it is shown in Figure 21; in contrast, in a 3D case the ϕ function is a hypersurface.

4.3.3. INTERFACE RECONSTRUCTION

As it has been stated former, in the level set method the interface is represented implicitly through the ϕ function. Therefore, in order to extract the interface curve, it is necessary to do a reconstruction of the front by means of an interpolation of the discretized ϕ function between the grid nodes. Moreover, reconstructing the interface is necessary for plotting purposes such as showing the grain burning surface evolution.

On the other side, the simulation of the SRM internal ballistics requires some interface geometrical properties such as the perimeter, burning area and port area. Yet, it is not strictly necessary to reconstruct the interface for this purpose. For instance, in [8] Heaviside functions are used to compute the geometrical properties of the interface. Nevertheless, in this study the interface geometrical properties are calculated from the reconstructed interface shape (2D curve). So, at each time step, the interface is reconstructed and then it is used to calculate the grain geometrical properties.

4.3.3.1. Interface shape, perimeter and burning area

In order to extract the interface shape from the discretized ϕ function in the background grid, it is necessary to find the points where the ϕ function is equal to 0. Recall the equation (4.1), which states that the interface is defined by the zero level set of the ϕ function. However, it is very unlikely that any grid point has an exact value of $\phi = 0$. Thus, with the aim of finding the points where $\phi = 0$ an interpolation between the grid nodes is done.

The interpolation between the grid nodes is known as the cut-cell method. Basically, the method consists of studying the grid cells an interpolating the values of the ϕ function between the cell nodes. On the other hand, with the aim of finding the interface it is necessary to identify the cells where the sign of the ϕ function changes. Those are the boundary cells. In fact, due to the level set to seek is $\phi = 0$, the interface will pass through the cells in which the sign of ϕ is not the same in all the nodes. An example of this property is shown in Figure 22.

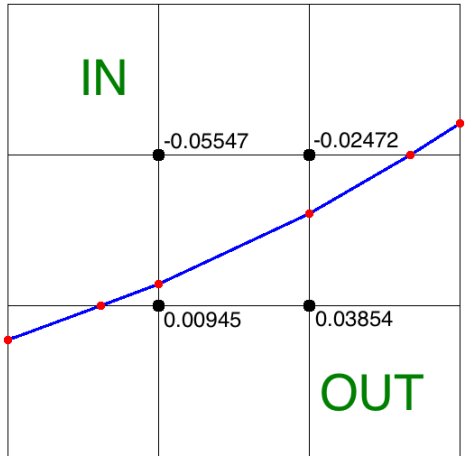

Figure 22. Example of interface reconstruction from the discretized ϕ function.

In Figure 22 the interface is the blue line, the interface points are red and the studied cell nodes are the black markers. Also, the numbers that appear in the figure are the ϕ values at the nodes of the cell studied (the center cell). As it shown in this figure, in the cell studied the sign of ϕ is not the same in all the nodes. Indeed, in the two upper nodes ϕ is negative while on the two lower nodes ϕ is positive. This means that the inner region of the interface (gas side) is above, while the outer region of the interface (propellant) is below. Recall that $\phi < 0$ inside the interface delimited region while $\phi > 0$ outside, as it is shown in Figure 13.

This way, the philosophy of the cut-cell method consists of using the ϕ values at the nodes of the boundary cells to find out approximately where the interface is. Furthermore, the next considerations are done in the interface reconstruction process:

- Each edge of a boundary cell can only be intersected once by the interface.
- Inside each boundary cell, the interface shape is assumed to be a straight line (two lines in case of double-crossing, see Figure 24).
- In case of double-crossing the interface cannot intersect itself.

Graphically, Figure 23 shows the list of cases considered within these assumptions.

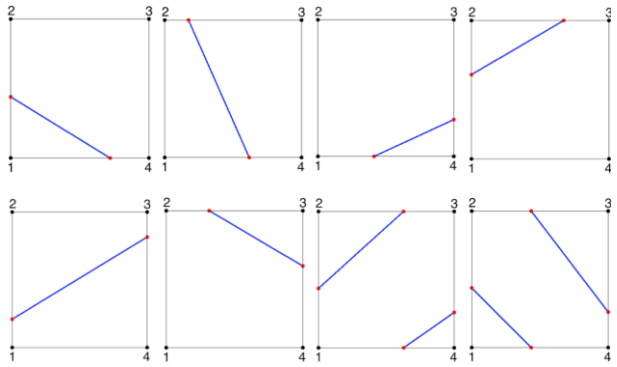

Figure 23. List of cases considered in the interface reconstruction process.

In the first 6 cases of Figure 23 the interface crosses the cell only once, so that it enters from one edge and leaves for another. However, in the last two cases the interface intersects the cell twice. Here, this configuration is called double-crossing. In this configuration the interface enters from one edge, leaves for another and after some time it enters again to the cell as it is shown in Figure 24. The double-crossing configuration can occur next to sharp tips in the grain.

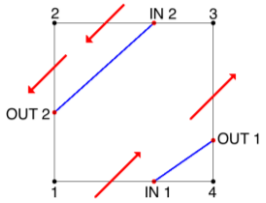

Figure 24. Example of double-crossing configuration.

Next, the algorithm for the interface reconstruction in 2D developed in this study is presented.

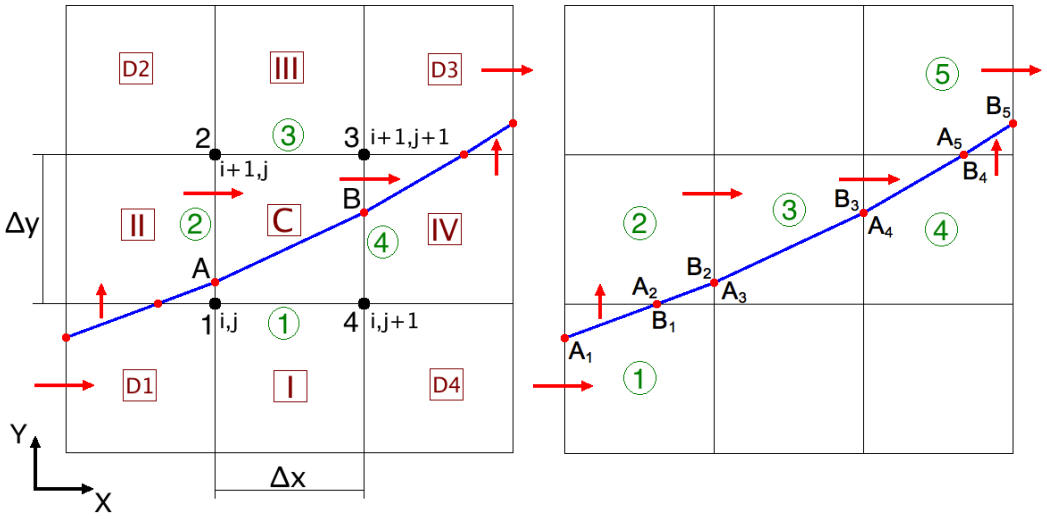

Figure 25. Scheme of the interface tracker algorithm.

The core idea of this algorithm consists of constructing the interface from cell to cell by tracking the zero level set at each cell, until the curve closes itself. To do so, the interface tracker algorithm uses the background grid points' coordinates and values of the discretized ϕ function.

Figure 25 (L) shows how the algorithm works at each cell. In this figure, the cell studied is called C, and the interface points intersecting this cell are A and B. The cell names are in brown color, the edges of the studied cell are in green color, the nodes of the studied cell are the black markers and the red arrows show the sense of the interface reconstruction process. It should be pointed out that the location of A is already known from the previous cell studied i.e. $\mathrm{A^k = B^{k-1}}$ (as it is shown in Figure 25 (R)). Thus, the objective when studying a cell is to find out the position of B. Once B is found, the next cell to be studied is the one adjacent to the edge where B is. In this example, B is located in the edge 4 (right edge) of the C cell, so the next cell to be studied is the cell IV. At the same time, the position of A in the next cell to be studied is the location of B in the currently studied cell (C in this case). Thereby, this procedure is repeated until the curve closes itself i.e. $\mathrm{B^{last} = A^1}$. It should be pointed out that $\mathrm{A^1}$ is found by doing a scan of the grid cells until a cell in which the sign of ϕ is not the same in all the nodes is found. Then, $\mathrm{A^1}$ is calculated and so the reconstruction process starts.

In order to show how the algorithm works, this section presents only the case when A is located in the lower edge of the studied cell (edge 1). The rest of the cases (A is in the left edge, A is in the upper edge and A is in the right edge) behave exactly the in the same way. Additionally, for each of the 4 location cases of A, there are 3 sub-cases for the location of B. Then, for the case presented here where A is in the lower edge of the studied cell, there are 3 sub-cases for the location of B: B is in the left edge, B is in the upper edge and B is in the right edge. Therefore, the objective is to find the location of B in the studied cell; the location of A is already known from the previous calculation. In order to find B, the x and y coordinates as well as the value of ϕ for each of the studied cell nodes (1,2,3,4) are used:

Cell information

$$\mathrm{x_1 = X(i,j);} \qquad \mathrm{y_1 = Y(i,j);} \qquad \mathrm{\phi_1 = \phi(i,j,n)}$$

$$\mathrm{x_2 = X(i+1,j);} \qquad \mathrm{y_2 = Y(i+1,j);} \qquad \mathrm{\phi_2 = \phi(i+1,j,n)}$$

$$\mathrm{x_3 = X(i+1,j+1);} \qquad \mathrm{y_3 = Y(i+1,j+1);} \qquad \mathrm{\phi_3 = \phi(i+1,j+1,n)}$$

$$\mathrm{x_4 = X(i,j+1);} \qquad \mathrm{y_4 = Y(i,j+1);} \qquad \mathrm{\phi_4 = \phi(i,j+1,n)}$$

Where X and Y are the matrices with the x and y coordinates of the background grid points, and ϕ is the matrix with the values of the discretized $\phi(x, y, t)$ function.

On the other side, the three sub-cases belonging to the case when A is located in the lower edge of the studied cell are identified as follows:

If $\phi_1\phi_2 < 0$	*then*	B is located in the left edge of the studied cell
If $\phi_2\phi_3 < 0$	*then*	B is located in the upper edge of the studied cell
If $\phi_3\phi_4 < 0$	*then*	B is located in the right edge of the studied cell

- Sub-case B is located in the left edge of the studied cell (A is located in the lower edge):

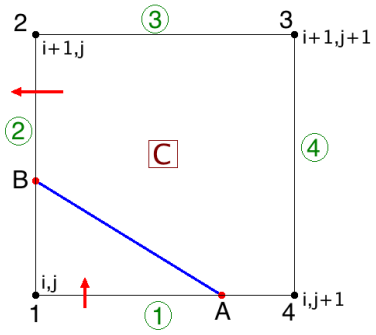

Figure 26. Sub-case B is located in the left edge of the studied cell.

The position of B can be calculated as follows by interpolating with the values of the discretized ϕ function and the coordinates of the nodes 1, 2, 3 and 4.

$$x_B = x_1 \tag{4.26}$$

$$y_B = y_1 + \frac{|\phi_1|}{|\phi_1| + |\phi_2|}\Delta y \tag{4.27}$$

As well, note that $\Delta x = \Delta y$ due to the background grid is Cartesian. Moreover, due to B is located in the edge 2 (left edge), the next cell to be studied is the cell adjacent to the edge 2 i.e. the cell II (using the denomination of the Figure 25). The subscripts of the node 1 of the next cell to be studied are $(i, j-1)$. Also, the position A of the next cell to be studied is assigned as the position of B calculated in the current studied cell:

$$x_A{}^{next} = x_B; \qquad y_A{}^{next} = y_B \tag{4.28}$$

- Sub-case B is located in the upper edge of the studied cell (A is located in the lower edge):

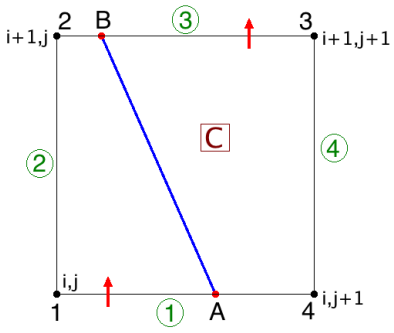

Figure 27. Sub-case B is located in the upper edge of the studied cell.

Analogously to the previous sub-case, the position of B can be calculated as follows:

$$x_B = x_1 + \frac{|\phi_2|}{|\phi_2| + |\phi_3|}\Delta x \tag{4.29}$$

$$y_B = y_2 \tag{4.30}$$

Due to B is located in the edge 3 (upper edge), the next cell to be studied is the cell adjacent to the edge 3 i.e. the cell III (using the denomination of the Figure 25). The subscripts of the node 1 of the next cell to be studied are $(i+1, j)$. As well, the position A of the next cell to be studied is assigned as in (4.28).

- Sub-case B is located in the right edge of the studied cell (A is located in the lower edge):

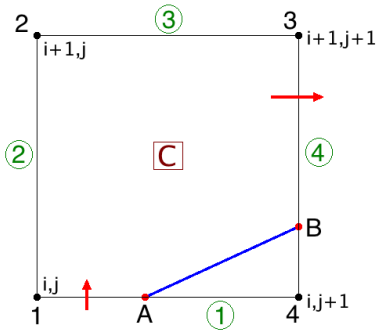

Figure 28. Sub-case B is located in the right edge of the studied cell.

Same as the previous sub-cases, the position of B can be calculated as follows:

$$x_B = x_4 \tag{4.31}$$

$$y_B = y_1 + \frac{|\phi_4|}{|\phi_3| + |\phi_4|}\Delta y \tag{4.32}$$

Due to B is located in the edge 4 (right edge), the next cell to be studied is the cell adjacent to the edge 4 i.e. the cell IV (using the denomination of the Figure

25). The subscripts of the node 1 of the next cell to be studied are $(i, j+1)$. As well, the position A of the next cell to be studied is assigned as in (4.28).

On the other side, the double-crossing configuration case is presented next. Firstly, the double-crossing configuration occurs if the next conditions are satisfied:

$$\phi_1\phi_2 < 0 \quad \& \quad \phi_2\phi_3 < 0 \quad \& \quad \phi_3\phi_4 < 0 \quad \& \quad \phi_1\phi_4 < 0$$

As well, it should be pointed out that the double-crossing configuration is treated with the same reconstruction procedure but applying a certain condition to it. In this configuration, it is necessary to discriminate what is the right B point from the two possible locations available. As an example of this, as it is shown in Figure 24, when the interface enters the cell from the point IN 1, it has two possible locations for B: OUT1 or OUT 2. The point IN 2 is discarded because the interface cannot intersect itself. Hence, it is necessary to choose which of the two possible locations of B are correct. This discrimination is done by choosing the B location that better follows the trend direction of the interface. Due to the interface shape before the studied cell is known, by comparing the direction of the previous cell studied and the two possible directions in the current cell, it is possible to know what is the right B location. This comparison is done by doing a dot product as follows:

$$\begin{cases} \textit{If } (\overline{BA})_{prev\_cell} \cdot \overline{B_1A} > (\overline{BA})_{prev\_cell} \cdot \overline{B_2A} & \text{then} \quad B_1 \text{ is the right B point.} \\ \textit{If } (\overline{BA})_{prev\_cell} \cdot \overline{B_1A} < (\overline{BA})_{prev\_cell} \cdot \overline{B_2A} & \text{then} \quad B_2 \text{ is the right B point.} \end{cases}$$

Where $(\overline{BA})_{prev\_cell}$ is the tangent vector to the interface in the previous studied cell, and $\overline{B_1A}$ and $\overline{B_2A}$ are the two possible tangent vectors in the current studied cell. Therefore, using this condition it is possible to choose the right location of B and continue tracking the interface with the same procedure explained before in the three sub-cases.

In summary, these cases show the philosophy and the calculation procedure of the interface tracker algorithm. Now, it is possible to calculate the interface perimeter using the results from the interface reconstruction. Indeed, while calculating the location of the interface points in the reconstruction process, it is possible to compute the length of each interface segment as follows:

$$L_i = \sqrt{\left(x_B^i - x_A^i\right)^2 + \left(y_B^i - y_A^i\right)^2} \tag{4.33}$$

Where "i" is the index of the cell studied. By summing the length of each interface segment, the burning perimeter is obtained:

$$P_b = \sum_{i=1}^{N} L_i \tag{4.34}$$

As well, the grain burning area is calculated by multiplying the burning perimeter by the length of the grain:

$$A_{b_{IGE}} = P_b \cdot L_{grain} \tag{4.35}$$

The value of this area is calculated considering that the grain ends are inhibited (IGE=Inhibited Grain Ends), and hence do not burn. However, in most of the cases the grain ends burn (EB=Ends Burn) thereby contributing to the total burning area as follows:

$$A_{b_{EB}} = P_b\left(L_{grain} - 2 \cdot w\right) + 2\left(\pi \cdot {R_{case}}^2 - A_p\right) \tag{4.36}$$

Where w is the grain burnt depth (or web burned), R_{case} is the motor case radius and A_p is the port area (that is calculated in the next section). The 2D grain burnback analysis simulation tool uses the value of $A_{b_{EB}}$ for the burning area calculation.

Finally, the grain burning area is later used in the simulation of the solid rocket motor internal ballistics.

4.3.3.2. Interface port area

The port area is the cross-sectional area in which the gas flow passes through. As it is shown in Figure 13, the port area is the region delimited by the interface, i.e. the region where $\phi \leq 0$ (the boundary is included). Thereby, the calculation of the port area is done using the discretized ϕ function on the background grid. There are 7 cases considered in the calculation of the port area. Here, the double-crossing case is neglected; however the error caused by this assumption is irrelevant. Next, the seven cases of the interface situation in the cell are presented.

- Case 1, when $\phi_1 < 0 \quad \& \quad \phi_2 < 0 \quad \& \quad \phi_3 < 0 \quad \& \quad \phi_4 < 0$:

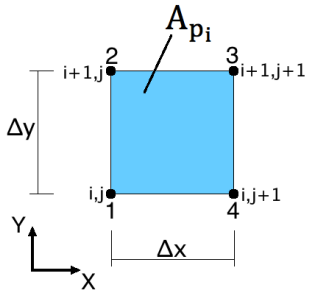

Figure 29. Case 1, the whole cell belongs to the interface delimited region.

$$A_{p_{i\_cell}} = A_{p_i} = \Delta x \cdot \Delta y \tag{4.37}$$

Where $A_{p_{i\_cell}}$ is the port area contribution of the cell studied.

- Case 2, when $\phi_1\phi_2 < 0 \quad \& \quad \phi_1\phi_4 < 0$:

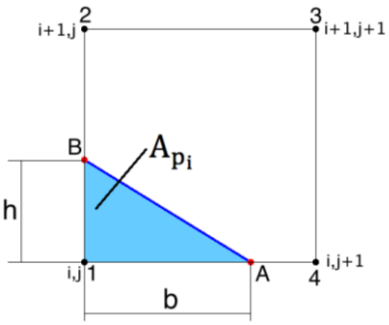

Figure 30. Case 2 of the interface crossing the cell.

$$A_{p_i} = \frac{1}{2} b \cdot h = \frac{1}{2}\left(\frac{|\phi_1|}{|\phi_1| + |\phi_4|}\Delta x\right)\left(\frac{|\phi_1|}{|\phi_1| + |\phi_2|}\Delta y\right) \tag{4.38}$$

$$\begin{cases} \textit{If} \quad \phi_1 < 0 \quad \textit{then} \quad A_{p_{i\_cell}} = A_{p_i} \\ \textit{If} \quad \phi_1 > 0 \quad \textit{then} \quad A_{p_{i\_cell}} = \Delta x \cdot \Delta y - A_{p_i} \end{cases}$$

- Case 3, when $\phi_1\phi_4 < 0 \quad \& \quad \phi_2\phi_3 < 0$:

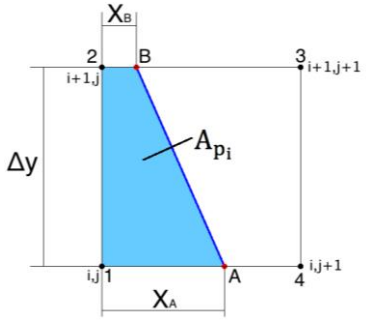

Figure 31. Case 3 of the interface crossing the cell.

$$A_{p_i} = \frac{1}{2}(x_A + x_B)\Delta y = \frac{1}{2}\left(\frac{|\phi_1|}{|\phi_1| + |\phi_4|}\Delta x + \frac{|\phi_2|}{|\phi_2| + |\phi_3|}\Delta x\right)\Delta y \tag{4.39}$$

$$\begin{cases} \textit{If} \quad \phi_1 < 0 \quad \textit{then} \quad A_{p_{i\_cell}} = A_{p_i} \\ \textit{If} \quad \phi_1 > 0 \quad \textit{then} \quad A_{p_{i\_cell}} = \Delta x \cdot \Delta y - A_{p_i} \end{cases}$$

- Case 4, when $\phi_1\phi_4 < 0 \quad \& \quad \phi_3\phi_4 < 0$:

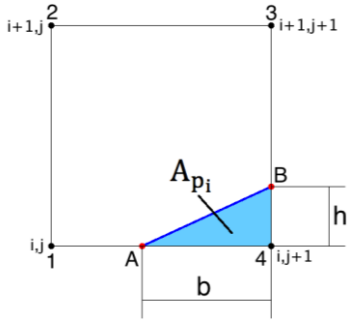

Figure 32. Case 4 of the interface crossing the cell.

$$A_{p_i} = \frac{1}{2} b \cdot h = \frac{1}{2}\left(\frac{|\phi_4|}{|\phi_1| + |\phi_4|}\Delta x\right)\left(\frac{|\phi_4|}{|\phi_3| + |\phi_4|}\Delta y\right) \tag{4.40}$$

$$\begin{cases} \textit{If} \quad \phi_4 < 0 \quad \textit{then} \quad A_{p_{i\_cell}} = A_{p_i} \\ \textit{If} \quad \phi_4 > 0 \quad \textit{then} \quad A_{p_{i\_cell}} = \Delta x \cdot \Delta y - A_{p_i} \end{cases}$$

- Case 5, when $\phi_1\phi_2 < 0 \quad \& \quad \phi_2\phi_3 < 0$:

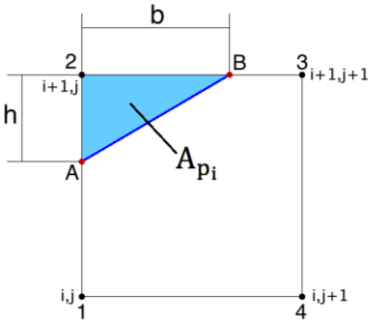

Figure 33. Case 5 of the interface crossing the cell.

$$A_{p_i} = \frac{1}{2} b \cdot h = \frac{1}{2}\left(\frac{|\phi_2|}{|\phi_2| + |\phi_3|}\Delta x\right)\left(\frac{|\phi_2|}{|\phi_1| + |\phi_2|}\Delta y\right) \tag{4.41}$$

$$\begin{cases} \textit{If} \quad \phi_2 < 0 \quad \textit{then} \quad A_{p_{i\_cell}} = A_{p_i} \\ \textit{If} \quad \phi_2 > 0 \quad \textit{then} \quad A_{p_{i\_cell}} = \Delta x \cdot \Delta y - A_{p_i} \end{cases}$$

- Case 6, when $\phi_1\phi_2 < 0 \quad \& \quad \phi_3\phi_4 < 0$:

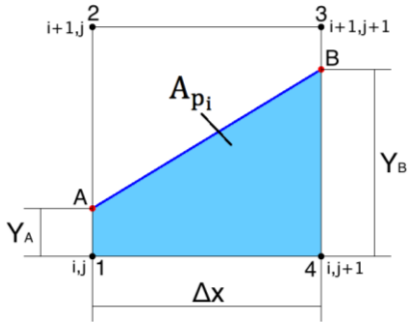

Figure 34. Case 6 of theinterface crossing the cell.

$$A_{p_i} = \frac{1}{2}(y_A + y_B)\Delta x = \frac{1}{2}\left(\frac{|\phi_1|}{|\phi_1| + |\phi_2|}\Delta y + \frac{|\phi_4|}{|\phi_3| + |\phi_4|}\Delta y\right)\Delta x \tag{4.42}$$

$$\begin{cases} \textit{If} \quad \phi_1 < 0 \quad \textit{then} \quad A_{p_{i\_cell}} = A_{p_i} \\ \textit{If} \quad \phi_1 > 0 \quad \textit{then} \quad A_{p_{i\_cell}} = \Delta x \cdot \Delta y - A_{p_i} \end{cases}$$

- Case 7, when $\phi_2\phi_3 < 0 \quad \& \quad \phi_3\phi_4 < 0$:

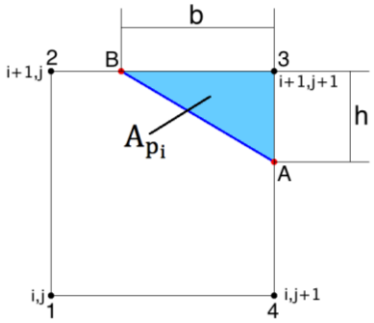

Figure 35. Case 7 interface crossing the cell.

$$A_{p_i} = \frac{1}{2}b \cdot h = \frac{1}{2}\left(\frac{|\phi_3|}{|\phi_2| + |\phi_3|}\Delta x\right)\left(\frac{|\phi_3|}{|\phi_3| + |\phi_4|}\Delta y\right) \tag{4.43}$$

$$\begin{cases} \textit{If} \quad \phi_3 < 0 \quad \textit{then} \quad A_{p_{i\_cell}} = A_{p_i} \\ \textit{If} \quad \phi_3 > 0 \quad \textit{then} \quad A_{p_{i\_cell}} = \Delta x \cdot \Delta y - A_{p_i} \end{cases}$$

Finally, by summing the port area contributions from all the cells, the total port area is obtained:

$$A_p = \sum_{i=1}^{N} A_{p_{i\_cell}} \tag{4.44}$$

The port area is later used in the simulation of the solid rocket motor internal ballistics.

4.3.4. INTERFACE PROPAGATION

Once the discretized ϕ function is initialized with the values of the minimum distance function (4.25), it can be propagated by calculating (4.12) in a loop for all the background grid points. Note that in (4.12) there are three dimensions, x, y and z, however in 2D grain burnback analysis z is not used. So, all the terms containing z disappear from (4.12) and its internal terms. On the other side, the value of F in (4.12) is the burning rate, which is determined by the internal ballistics simulation. Moreover, in 2D grain burnback analysis only a 0D flow model can be used for the internal ballistics simulation, hence the burning rate is spatially constant i.e. $F_{i,j} = F = \dot{r}_b$, but varies with the time. Finally, the interface can be extracted at any time by finding the zero level set of the discretized ϕ function, namely by reconstructing the interface (see 4.3.3.1).

4.4. QUASI-3D GRAIN BURNBACK ANALYSIS

In solid rocket motors with 3D grain designs, the grain cross-section varies along the motor longitudinal axis (z). Therefore, it is necessary to study the 3 dimensions of the grain geometry to perform the grain burnback analysis. Moreover, as it has been aforementioned, in 3D grain burnback analysis the interface to propagate is a surface. This way, this surface is evolved using the burning rate calculated in the internal ballistics simulation. Depending on the flow model used, the burning rate can be either spatially constant (0D model) or variable (1D or more) in a given time step. On the other side, the geometrical properties obtained from the grain regression analysis are sent to the internal flow simulation as inputs. So, the grain burnback analysis is coupled with the internal ballistics simulation.

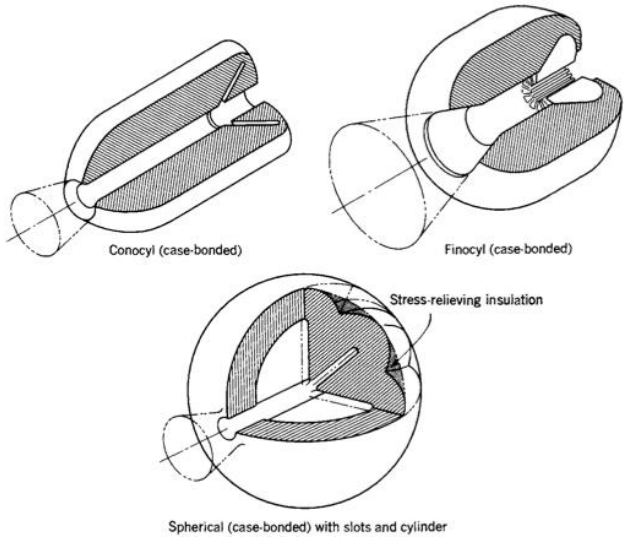

Figure 36. Typical 3D grain configurations (source [11]).

To sum up, all nowadays advanced solid rocket motors use 3D grain designs because of their higher performance and capabilities. Also, 3D grains can provide more complex thrust curves than 2D grains, hence allowing for more efficient trajectories. Some representative examples of 3D grain designs are the Space Shuttle Solid Rocket Boosters (SRBs) from NASA, as well as the ESA Ariane 4 and Ariane 5 SRB's and the first three stages of VEGA.

4.4.1. QUASI-3D MODEL

In this study, the 3D grain burnback analysis has been developed using an innovative approach, called Quasi-3D model, with the aim of reducing the overall computational time. The philosophy of the Quasi-3D approach consists of modeling the 3D grain burning surface through few 2D reference cross-sections. These reference cross-sections contain the grain surface main geometric characteristics. As well, a part from these, additional cross-sections are generated by interpolating the grain shape between the reference cross-sections. Then, the grain burning surface shape is reconstructed by doing a triangulation process between the points of each cross-section.

In this way, instead of propagating a 3D surface with the level set method in 3-dimensions, only the reference cross-sections are propagated using the LSM in 2D. Thereby, in the Quasi-3D model, the interfaces propagated with the LSM are 2D curves belonging to each cross-sectional plane. After all the 2D curves are propagated in a time step, the 3D grain surface is reconstructed using the triangulation process. Then, the 3D surface geometrical parameters (burning area, propellant volume, etc.) needed for the ballistics simulation are computed. The aim of this approach is to reduce the computational time by operating only in few grain cross-sections. In order to illustrate the Quasi-3D philosophy, two examples are shown in Figure 37 and Figure 38.

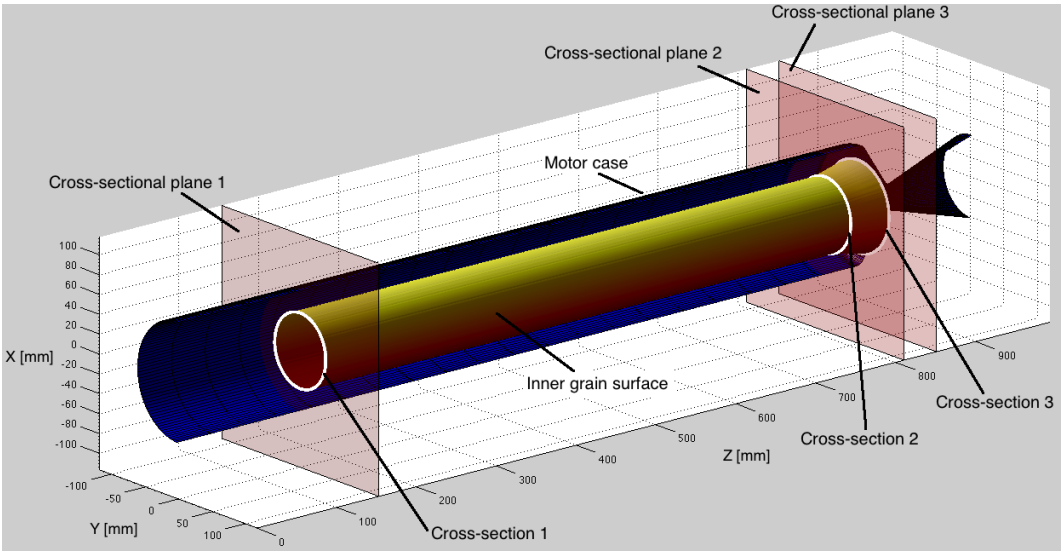

Figure 37. Reference cross-sections treatment with a 3D grain geometry (NAWC motor No. 13, 3 cross-sections)

As it is shown in Figure 37, the grain geometry of the NAWC motor No. 13 is tri-dimensional, having three reference cross-sections. Hence, the 3D grain surface is modeled using only 3 cross-sections, each belonging to a cross-sectional plane. So, each grain reference cross-section is the intersection between the 3D grain surface and its cross-sectional plane. Therefore, the Quasi-3D grain burnback analysis consists of propagating each reference cross-section curve using the LSM in 2D, given a burning rate. Then, the 3D grain burning surface is reconstructed through a triangulation process between the points of each cross-section.

On the other side, this example, for instance, has to be treated with a 0D flow model because with so few reference cross-sections the effect of erosive burning and longitudinally variable burning rate would be badly modeled. In order to account for this 1D flow model effects, it is necessary to use more reference cross-sections such as in the example from Figure 38.

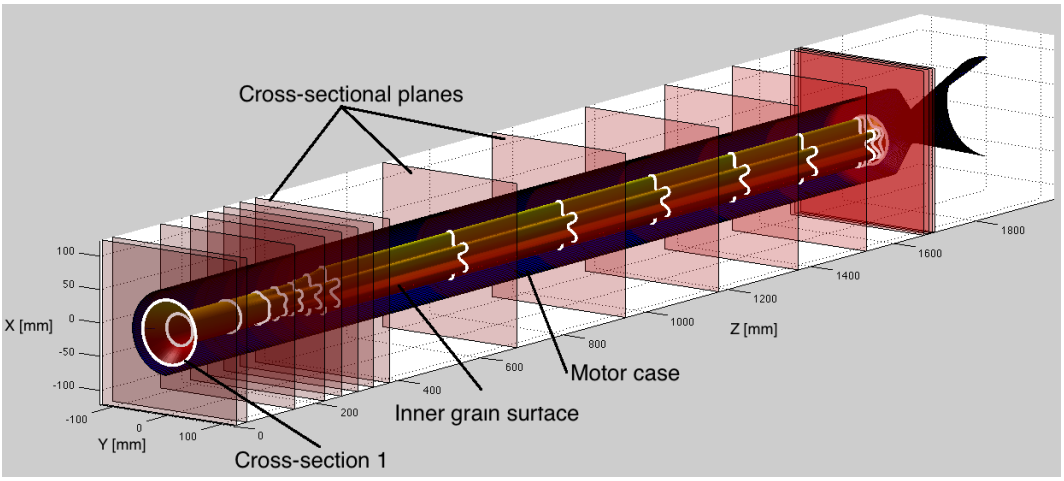

Figure 38. Reference cross-sections treatment with another 3D grain geometry (NAWC motor No. 6, 17 cross-sections)

As it is shown in Figure 38, the NAWC motor No. 6 has a 3D grain, which is modeled here with 17 cross-sections. The grain geometry departs from a circular cross-section at the head end (cross-section 1) and evolves to a 6 point star at the opposite end. Due to this grain geometry is modeled with more cross-sections it could allow the use of a 1D flow model. With a 1D model, each cross-section would be propagated with a different burning rate defined by the erosive burning and the local pressure at each cross-section. However, this option has not been conducted in this study and only a 0D flow model has been coupled with the Quasi-3D grain burnback.

In order to propagate the 3D grain burning surface, the Quasi-3D model uses the level set method in 2D for each reference cross-section. This way, for each reference cross-section there is a 2D discretized $\phi_k(x,y,t)$ function which is propagated using the burning rate calculated in the internal ballistics simulation. Where k is the subscript for identifying each reference cross-section. As well, for each reference cross-section the discretized $\phi_k(x,y,t)$ function is initialized using a minum distance function, as it is done in the section 4.3.2. So for each cross-section, the minimum distance function is calculated and then the discretized $\phi_k(x,y,t)$ function is initialized as follows:

$$\phi_k(x,y,t=0) = \mathrm{MDF}_k(x,y) \qquad \forall x,y \in \text{Background grid}; \quad k = 1,\ldots..,N_{rcs} \qquad (4.45)$$

Where N_{rcs} is the number of reference cross-sections.

4.4.2. INTERFACE RECONSTRUCTION

In the Quasi-3D model, the 3D grain burning surface, namely the interface, is reconstructed in 2 phases: reconstruction of the zero level set of each reference cross-section and triangulation between the points of each cross-section to form

the 3D surface. In this way, the first step consists of finding out the points of the zero level set of the $\phi_k(x, y, t)$ function for each reference cross-section. These points represent the intersection of the 3D grain burning surface with the cross-sectional planes. After this, the 3D grain surface is reconstructed by joining the points of each cross-section through a triangulation process.

On the other side, the simulation of the SRM internal ballistics requires some interface geometrical properties such as the burning area and the propellant volume. So that, in this study the interface geometrical properties are calculated from the reconstructed interface shape, which in this case is a 3D surface. So, at each time step, the 3D grain surface is reconstructed and then it is used to calculate the grain geometrical properties.

4.4.2.1. Cross-sections shape, perimeter and port area

As it has been aforementioned, in the Quasi-3D approach the grain burning surface is modeled using the reference cross-sections. Each reference cross-section is a 2D curve belonging to a cross-sectional plane (parallel to the x-y plane), and represents the intersection of the 3D grain burning surface with this plane. Then for each reference cross-section there is a $\phi_k(x, y, t)$ function discretized on a Cartesian background grid. This is the level set method ϕ function, and represents the propagating interface implicitly through its zero level set. Therefore, in order to reconstruct the interface of a reference cross-section, it is necessary to find the zero level set by means of an interpolation of the discretized ϕ function between the Cartesian grid nodes. This reconstruction of the zero level set is done in the same way that has been used for 2D grain burnback analysis (see 4.3.3.1). In the case of Quasi-3D model, this operation is repeated for each reference cross-section.

The interface perimeter of each reference cross-section is calculated using the equation (4.34). On the other side, the port area is calculated with the same procedure used for 2D grain burnback analysis (see 4.3.3.2), by computing the equation (4.44). The perimeter and the port area of each cross-section are calculated for each time step because they are necessary for computing the burning surface area and the propellant volume.

4.4.2.2. Cross-sections triangulation and interpolation

The second phase of the interface reconstruction consists of doing a triangulation between the points of each cross-section in order to form the 3D grain surface. The triangulation is the process of generation of triangular elements that join each point from all the cross-sections. Moreover, in order to represent better the 3D grain surface, additional cross-sections are generated by interpolating the

grain shape between the reference cross-sections. It should be pointed out that these additional cross-sections are not propagated using the level set method as it is done with the reference cross-sections. Indeed, the additional cross-sections are not propagated but are created at each time step interpolating the grain shape between the reference cross-sections. The purpose of these additional cross-sections is to improve the construction of the 3D grain surface, as well as to provide more cross-sections for the burning surface area calculation. In the Matlab® code developed, the user introduces as an input the number of additional cross-sections that are created between each reference cross-section.

With respect to the triangulation operation, a function named tri_generation.m (section A.2.2.16 ANNEX A) has been developed in this study with the aim of generating the triangular elements that connect the points of two given cross-sections. This way, this function receives the coordinates of the points belonging to two cross-sections and generates the triangular elements connections. Through the coordinates of the points of the cross-sections and the triangular elements connections, a surface can be created as it is shown Figure 39.

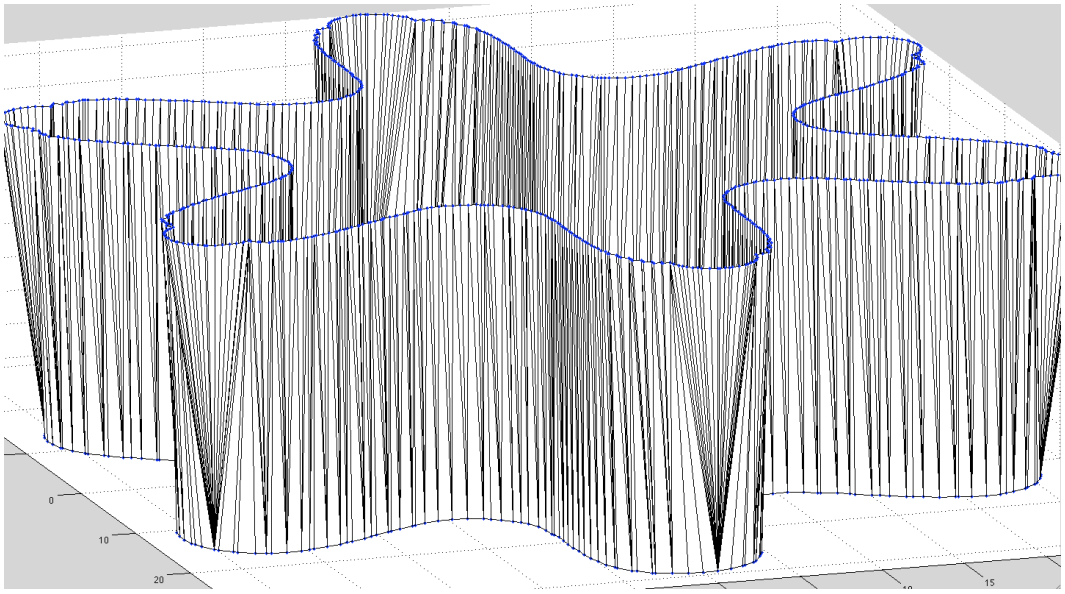

Figure 39. Triangulation between two cross-sections.

In Figure 39, the points belonging to the two cross-sections are in blue color and the triangular elements have black edges and white faces. The algorithm of this developed function is based on finding the closest connections between the points of the two cross-sections. On the other side, by repeating this operation from cross-section to cross-section, the 3D grain surface can be constructed.

After presenting the triangulation generation process between two cross-sections, the following paragraphs are dedicated to explain how a new cross-section between two existing cross-sections can be generated. The aim is to generate a

cross-section by interpolating the grain shape between two given cross-sections. To do so, a function named interpolate_cross_section.m (section A.2.2.15 ANNEX A) has been developed in this study.

Firstly, before the cross-section interpolation begins, it is necessary to create the triangular elements that connect the points of the two existing cross-sections. This operation is done using the aforementioned tri_generation function. Then, once the triangles are generated, the interpolated cross-section is created by intersecting the surface formed by the triangular elements with the interpolated cross-section plane. This process is shown in Figure 40.

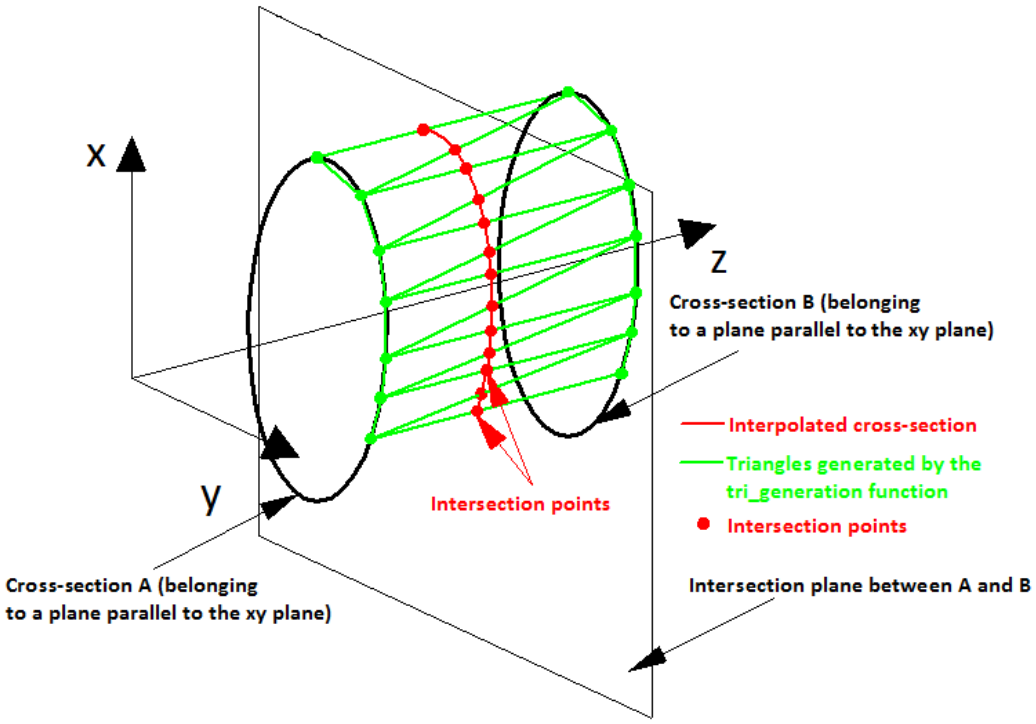

Figure 40. General view of the creation of an interpolated cross-section.

In Figure 40, the two existing cross-sections (A and B) are colored in black; the triangular elements that form the 3D surface are in green color and the interpolated cross-section is colored in red. As it is shown in this figure, the interpolated cross-section is generated by finding the intersection points between the triangles and the intersection plane (interpolated cross-section plane).

In order to detail the process, Figure 41 shows the operation performed by the interpolate_cross_section function for a given triangular element. Recall that the triangular elements connect the points from the cross-sections A and B, thereby generating a 3D surface.

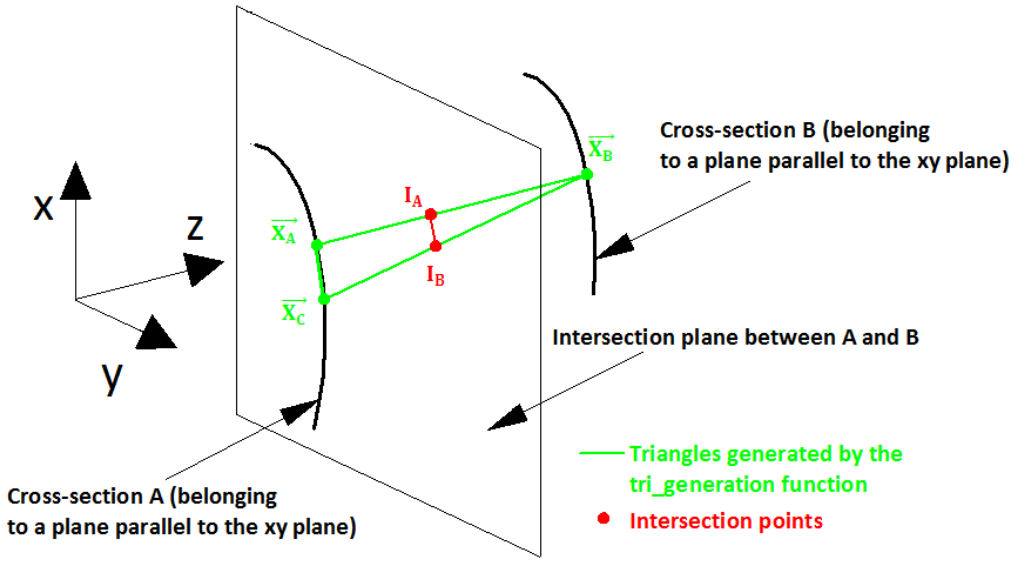

Figure 41. Detail of the procedure followed to create the interpolated cross-section.

Hence, once the triangles connecting the points of the cross-sections A and B are generated, the cross-section interpolation begins. Basically, the goal of this interpolation is to find the intersection points between the interpolated cross-section plane and the triangles joining the cross-sections A and B, as it is shown in Figure 41. In this case, in order to create cross-sections with similar number of points, only the point I_1 is searched. This way, one intersection point for triangle is enough to represent correctly the intersection curve. Consequently, it is necessary to find the intersection between the triangle edges and the intersection plane. For that, firstly the edge of the triangle joining the vertexes $\overrightarrow{X_A}$ and $\overrightarrow{X_B}$ should be defined. Following the nomenclature used in Figure 41, the general expression for this edge is the same that the equation of a line:

$$\vec{X}(x,y,z) = \overrightarrow{X_A} + (\overrightarrow{X_B} - \overrightarrow{X_A})s \tag{4.46}$$

where $\overrightarrow{X_A}$ is the position vector of the i-th point of the cross-section A, $\overrightarrow{X_B}$ is the position vector of the j-th point of the cross-section B and s is a parametric variable, which is limited as follows $s \in [0,1]$. Note that when $s = 0$, $\vec{X}(x,y,z) = \overrightarrow{X_A}$ and when $s = 1$ $\vec{X}(x,y,z) = \overrightarrow{X_B}$. This expression (4.46) can be written vectorially:

$$\vec{X}(x,y,z) = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} x_A \\ y_A \\ z_A \end{bmatrix} + \begin{bmatrix} x_B - x_A \\ y_B - y_A \\ z_B - z_A \end{bmatrix} s \tag{4.47}$$

Once the expression for the triangle edge is defined, it is required to express mathematically the intersection plane. Note that the general formula for a plane is given by:

$$\vec{X}(x,y,z) = \vec{P} + \lambda\vec{u} + \mu\vec{v} \tag{4.48}$$

where $\vec{P}$ is the position vector of a point belonging to the plane, $\vec{u}$ is a vector from the plane, $\vec{v}$ is another vector belonging to the plane and λ and μ are arbitrary parametric variables. In this case, note that the cross-sections A and B of the grain belong to planes that are parallel to the plane xy (so, perpendicular to the longitudinal axis of the rocket motor, which is the z-axis of the Figure 41). Consequently, the intersection plane is also parallel to the plane xy. This property eases the calculation of the intersection plane equation. Consequently:

$$\vec{P} = \begin{bmatrix} 0 \\ 0 \\ z_C \end{bmatrix} ; \ \vec{u} = \vec{x} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} ; \ \vec{v} = \vec{y} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{4.49}$$

where z_C corresponds with the z-coordinate of the intersection plane (interpolated cross-section plane), the vector $\vec{u}$ has been equated to the $\vec{x}$ axis and the vector $\vec{v}$ has been equated to $\vec{y}$ axis. Therefore, if the values given in (4.49) are substituted in the general equation of the plane given in (4.48):

$$\vec{X}(x,y,z) = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \vec{P} + \lambda\vec{u} + \mu\vec{v} = \begin{bmatrix} 0 \\ 0 \\ z_C \end{bmatrix} + \lambda \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + \mu \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} \lambda \\ \mu \\ z_c \end{bmatrix} \tag{4.50}$$

When the plane defined in (4.50) intersects the triangle edge given by (4.47), the intersection is a point located at the edge as well as in the plane. Therefore, $\vec{X}_{plane} = \vec{X}_{triangle\_edge}$. This way, applying this equality with expressions (4.47) and (4.50):

$$\vec{X}_{plane} = \vec{X}_{triangle\_edge} \Rightarrow \begin{bmatrix} x_A \\ y_A \\ z_A \end{bmatrix} + \begin{bmatrix} x_B - x_A \\ y_B - y_A \\ z_B - z_A \end{bmatrix} s = \begin{bmatrix} 0 \\ 0 \\ z_C \end{bmatrix} + \lambda \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + \mu \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} \lambda \\ \mu \\ z_C \end{bmatrix} \tag{4.51}$$

From the third row of the equation (4.51) rapidly can be deduced that:

$$z_C = z_A + (z_B - z_A)s \tag{4.52}$$

Consequently, isolating the variable s from the expression (4.52):

$$s = \frac{z_C - z_A}{z_B - z_A} \tag{4.53}$$

Once the value of the variable is defined through (4.53), it can be substituted in expression (4.47). This way, the intersection point coordinate values are found as:

$$\vec{X}_I = \begin{bmatrix} x_I \\ y_I \\ z_I \end{bmatrix} = \begin{bmatrix} x_A \\ y_A \\ z_A \end{bmatrix} + \begin{bmatrix} x_B - x_A \\ y_B - y_A \\ z_B - z_A \end{bmatrix} \left(\frac{z_C - z_A}{z_B - z_A} \right) \tag{4.54}$$

This operation is repeated for all the triangular elements connecting A and B. Once all the I_1 intersection points from all the triangles are found, the interpolated cross-section curve is obtained by joining all these intersection points, as it is shown in Figure 40.

4.4.2.3. Interface geometrical properties

The internal ballistics simulation requires as inputs some grain geometrical parameters such as the grain burning surface and the propellant volume. The grain burning surface area and the propellant volume are calculated using respectively the perimeter and the port area of each cross-section.

Firstly, the mean burning area between two given cross-sections ith and i+1-th can be calculated using these cross-sections' burning perimeter:

$$A_{b\_cs}{}^{i} = \left(\frac{P_{b_{cs}}{}^{i} + P_{b_{cs}}{}^{i+1}}{2} \right) \left| z_{cs}{}^{i+1} - z_{cs}{}^{i} \right| \tag{4.55}$$

Where $P_{b_{cs}}{}^{i}$ and $P_{b_{cs}}{}^{i+1}$ are the burning perimeters of the ith and i+1-th cross-sections respectively. These perimeters are calculated using (4.34). As well, $z_{cs}{}^{i}$ and $z_{cs}{}^{i+1}$ are the z-coordinates of the cross-sectional planes of the ith and i+1-th cross-sections. Then, the total grain burning surface area is the sum of all the couples of cross-sections contributions to the burning area:

$$A_b = \sum_{i=1}^{N_{cs}-1} A_{b\_cs}{}^{i} \tag{4.56}$$

Where N_{cs} is the total number of cross-sections. On the other side, the mean port volume between two given cross-section ith and i+1-th can be calculated using the port area as follows:

$$V_{p\_cs}{}^{i} = \left(\frac{A_{p_{cs}}{}^{i} + A_{p_{cs}}{}^{i+1}}{2} \right) \left| z_{cs}{}^{i+1} - z_{cs}{}^{i} \right| \tag{4.57}$$

Where, ${A_{p_{cs}}}^{i}$ and ${A_{p_{cs}}}^{i+1}$ are the port area of the ith and i+1-th cross-sections respectively. These port area values are calculated using (4.44). Finally, the total propellant volume is the sum of all the couples of reference cross-sections contributions to the propellant volume:

$$V_{prop} = \pi \cdot R_{case}^2 \cdot L_{grain} - \sum_{i=1}^{N_{rcs}-1} {V_{p\_cs}}^{i} \tag{4.58}$$

Where N_{rcs} is the total number of reference cross-sections. Recall that the port area can only be calculated using the reference cross-sections, which have a discretized ϕ function. In contrast, the perimeter can be calculated for all the cross-sections, the reference ones and the additional ones added through the interpolate_cross_section function.

4.4.3. INTERFACE PROPAGATION

In the Quasi-3D model, the interface propagation is done in the same that has been used for the 2D grain burnback analysis, but doing a loop for all the reference cross-sections. This way, the first step is to initialize for each reference cross-section the discretized ϕ function with the values of the minimum distance function (4.25). Then, the ϕ function of each reference cross-section can be propagated by calculating (4.12) in a loop for all the background grid points. Note that in (4.12) there are three dimensions, x, y and z, however in Quasi-3D grain burnback analysis, the z-coordinate of the ϕ function is not used. Instead of using the z-coordinate, there is a discretized ϕ function for each reference cross-section. So, all the terms containing z disappear from (4.12) and its internal terms. On the other side, the value of F in (4.12) is the burning rate, which is determined by the internal ballistics simulation. Moreover, in this study the Quasi-3D grain burnback analysis is coupled only with a 0D flow model, so that the burning rate is spatially constant i.e. $F_{i,j} = F = \dot{r}_b$, but varies with the time. Finally, the interface is extracted for each time step using the procedure explained in the section 4.4.2.

5. SRM INTERNAL BALLISTICS & PERFORMANCE

This chapter develops the internal ballistics of the SRM. Precisely, it presents the 0D quasi-steady, 0D-unsteady and 1D quasi-steady flow formulation applied in this study. These flow models are complementary to the grain burnback explained before. Consequently, in order to obtain the final performance of the SRM both the grain burnback and the internal ballistics have to be considered. Therefore, in order to ease the comprehension of this chapter, it is recommended to read the chapter 4 as well as the basic formulation of the chapter 2. Additionally, in order to see the coupling of the grain burnback and the internal ballistics, see the chapter 6.

Following, a brief organization of the chapter 5 is presented. Firstly, the formulation required to describe the 0D quasi-steady flow model, also known as Lumped model, is presented. Then, the mathematical background of the 0D unsteady model is described. Finally, a model for the 1D quasi-steady flow is presented. Note that, from the literature, different ways to describe the 1D quasi-steady flow can be found. For this study, two models can be considered: the conservative form [13] and the non-conservative form [9]. As it has been mentioned in the scope of this study, the development of a 1D flow model is out of scope. Thereby, the model here presented is found from the literature.

5.1. QUASI-STEADY 0D FLOW MODEL (LUMPED MODEL)

One the one side, the mass flow rate exiting the nozzle $\dot{m}_d$ expressed in kg/m^3 is:

$$\dot{m}_d = \rho_t u_t A_t \tag{5.1}$$

where ρ_t is gas density at the throat expressed in kg/m^3, u_t is the gas velocity at the throat expressed in m/s and A_t is the throat are expressed in m^2. Additionally, the static temperature at the throat, T_t, can be related with the static temperature at the combustion chamber, T_c which is equal to the total temperature because the flow inside is considered to have $u_c \approx 0$, through:

$$\frac{T_c}{T_t} = 1 + \frac{\gamma - 1}{2} M_t^2 \tag{5.2}$$

If an adiabatic reversible process of a perfect gas is considered, the ratio of (5.2) static temperatures can be related with the ratio of densities as follows:

$$\frac{\rho_c}{\rho_t} = \left(\frac{T_c}{T_t}\right)^{\frac{1}{\gamma-1}} = \left(1 + \frac{\gamma - 1}{2} M_t^2\right)^{\frac{1}{\gamma-1}} \tag{5.3}$$

where ρ_c is gas density at the combustion chamber expressed in kg/m^3, M_t is the Mach number at the throat and γ is the heat specific value. Taking into account that at the throat $M_t = 1$, then, the expression (5.3) can be rewritten as:

$$\frac{\rho_c}{\rho_t} = \left(\frac{\gamma + 1}{2}\right)^{\frac{1}{\gamma - 1}} \tag{5.4}$$

So, from the equation (5.4) the gas density at the throat can be isolated as:

$$\rho_t = \rho_c \left(\frac{2}{\gamma + 1}\right)^{\frac{1}{\gamma - 1}} \tag{5.5}$$

Additionally, considering the definition of the Mach number and that $M_t = 1$, the gas velocity at the throat can be written as:

$$u_t = M_t \sqrt{\gamma R T_t} = \sqrt{\gamma R T_t} \tag{5.6}$$

where R is the universal gas constant R_o divided by the mean molar mass of the gas, $\overline{M}$ $(R = R_o/\overline{M})$ expressed in J/kg/K. In addition, if in the expression (5.2) T_t is isolated, equality $M_t = 1$ is applied and then, the expression obtained is substituted in the formula (5.6), the u_t can be written as:

$$u_t = \sqrt{\gamma R T_t} = \sqrt{\gamma R T_c \left(\frac{2}{\gamma + 1}\right)} \tag{5.7}$$

So, substituting expressions (5.5) and (5.7) into the expression (5.1), the mass flow rate exiting the nozzle is equivalent to:

$$\dot{m}_d = A_t \rho_c \left(\frac{2}{\gamma + 1}\right)^{\frac{1}{\gamma - 1}} \sqrt{\gamma R T_c \left(\frac{2}{\gamma + 1}\right)} \tag{5.8}$$

In addition, considering that the gas at the combustion chamber is ideal, the gas density, ρ_c, can be related with the static temperature, T_c, and static pressure at the combustion chamber P_c, which is also equal to the total pressure because the flow inside is considered to have $u_c \approx 0$, applying the ideal gas formulas as:

$$\rho_c = \frac{P_c}{R T_c} \tag{5.9}$$

where R has been defined before. Consequently, substituting the result from (5.9) into (5.8) and developing the expression, it is obtained that:

$$\dot{m}_d = A_t \frac{P_c}{R T_c} \left(\frac{2}{\gamma + 1}\right)^{\frac{1}{\gamma - 1}} \sqrt{\gamma R T_c \left(\frac{2}{\gamma + 1}\right)} = \sqrt{\gamma} \left(\frac{2}{\gamma + 1}\right)^{\frac{\gamma + 1}{2(\gamma - 1)}} \frac{P_c}{\sqrt{R T_c}} A_t \tag{5.10}$$

Note the definition given to the parameter Γ in (2.12). Then, considering (2.12), the expression (5.10) can be rewritten as:

$$\dot{m}_d = \left(\frac{2}{\gamma+1}\right)^{\frac{\gamma+1}{2(\gamma-1)}} \frac{P_c}{\sqrt{RT_c}} A_t = \Gamma \frac{P_c A_t}{\sqrt{RT_c}} \tag{5.11}$$

Furthermore, the characteristic velocity, c^*, is defined in (2.15). Consequently, substituting the formula (2.15) into (5.11), the mass flow rate exiting the nozzle is expressed as:

$$\dot{m}_d = \frac{P_c A_t}{c^*} \tag{5.12}$$

Note that the formula (5.12) corresponds with the expression (2.16). On the other side, the mass flow rate due to propellant combustion, $\dot{m}_g$, is:

$$\dot{m}_g = \rho_p A_b \dot{r}_b \tag{5.13}$$

where ρ_p is the propellant density in kg/m^3, A_b is the burning area in m^2 and burning rate in m/s. In steady state and neglecting the variation of the chamber volume, the mass flow rate due to propellant combustion is equivalent to the mass flow rate exiting the nozzle, which means, $\dot{m}_g = \dot{m}_d$. Thereby, equating expressions (5.12) and (5.13), it is obtained that:

$$\dot{m}_g = \dot{m}_d \rightarrow \rho_p A_b \dot{r}_b = \frac{P_c A_t}{c^*} \tag{5.14}$$

Finally, substituting the Vieille's burning law of (2.1) into the expression (5.14), the averaged combustion chamber pressure $\overline{P_c}$ can be isolated as:

$$\overline{P_c} = \left(a \rho_p c^* \frac{A_b}{A_t}\right)^{\frac{1}{1-n}} \tag{5.15}$$

5.2. UNSTEADY 0D FLOW MODEL

The transient mass conservation equation is defined as:

$$\dot{m}_g = \dot{m}_d + \frac{d(\rho_c V_c)}{dt} \tag{5.16}$$

where $\dot{m}_g$ is the mass flow rate due to propellant combustion expressed in kg/s and defined in (5.13); $\dot{m}_d$ is the mass flow rate exiting the nozzle in kg/m^3 defined previously in expressions (5.1), (5.8), (5.10), (5.11) and (5.12); ρ_c is the gas density in the combustion chamber expressed in kg/m^3 and V_c is the combustion

chamber volume expressed in m^3. Developing the derivative part of the expression (5.16) it can be obtained that:

$$\frac{d(\rho_c V_c)}{dt} = \rho_c \frac{dV_c}{dt} + V_c \frac{d\rho_c}{dt} \tag{5.17}$$

Note that the temporal variation of the combustion chamber volume is equal to:

$$\frac{dV_c}{dt} = A_b \dot{r}_b \tag{5.18}$$

where A_b is the burning area expressed in m^2 and $\dot{r}_b$is the burning rate in m/s and defined through the expression (2.1).

In order to define the temporal variation of the combustion chamber gas density, first, it is necessary to isolate the value of T_c from equation (2.15) as:

$$c^* = \frac{\sqrt{RT_c}}{\Gamma} \rightarrow T_c = \frac{(c^*\Gamma)^2}{R} \tag{5.19}$$

So, substituting the result from (5.19) into the equation (5.9), the combustion chamber gas density is:

$$\rho_c = \frac{P_c}{RT_c} = \frac{P_c}{(c^*\Gamma)^2} \tag{5.20}$$

Now, applying the temporal variation of the combustion chamber gas density in both sides of the equation (5.20), it is obtained that:

$$\frac{d\rho_c}{dt} = \frac{1}{(c^*\Gamma)^2} \frac{dP_c}{dt} \tag{5.21}$$

Finally, the expression (5.18) and (5.21) can be substituted into (5.17) reaching:

$$\frac{d(\rho_c V_c)}{dt} = \rho_c A_b \dot{r}_b + V_c \frac{1}{(c^*\Gamma)^2} \frac{dP_c}{dt} \tag{5.22}$$

Consequently, substituting expressions (5.13) instead of $\dot{m}_g$, (5.12) instead of $\dot{m}_d$ and (5.22) instead of $\frac{d(\rho_c V_c)}{dt}$ into (5.16):

$$\rho_p A_b \dot{r}_b = \frac{P_c A_t}{c^*} + \rho_c A_b \dot{r}_b + V_c \frac{1}{(c^*\Gamma)^2} \frac{dP_c}{dt} \tag{5.23}$$

Operating with equation (5.23), the temporal variation of combustion chamber pressure can be isolated as:

$$\frac{dP_c}{dt} = \frac{(c^*\Gamma)^2}{V_c} \left[(\rho_p - \rho_c) A_b \dot{r}_b - \frac{P_c A_t}{c^*} \right] \tag{5.24}$$

Note that in the previous equation the parameters ρ_c and $\dot{r}_b$ depend on P_c as it is shown in the equations (5.20) and (2.1).

In order to obtain the equation of combustion chamber pressure as a function of time, the expression (5.24) should be integrated obtaining:

$$P_c^{n+1} - P_c^n = \int_{P_c^n}^{P_c^{n+1}} dP_c = \int_{t^n}^{t^{n+1}} \frac{(c^*\Gamma)^2}{V_c}\left[(\rho_p - \rho_c)A_b\dot{r}_b - \frac{P_c A_t}{c^*}\right]dt \tag{5.25}$$

where P_c^n is the combustion chamber pressure in Pa in the time step t^n and P_c^{n+1} is the combustion chamber pressure in Pa in the time step t^{n+1}. Indeed, the value of P_c^{n+1} can be isolated from (5.25), finding:

$$P_c^{n+1} = P_c^n + \int_{t^n}^{t^{n+1}} \frac{(c^*\Gamma)^2}{V_c}\left[(\rho_p - \rho_c)A_b\dot{r}_b - \frac{P_c A_t}{c^*}\right]dt \tag{5.26}$$

In order to study the SRM performance through the Matlab® code, it is required to integrate numerically the previous expression (5.26). For that, an upwind scheme is used, finding:

$$P_c^{n+1} = P_c^n + \frac{(c^{*n}\Gamma^n)^2}{V_c^n}\left[(\rho_p - \rho_c^n)A_b^n\dot{r}_b^n - \frac{P_c^n A_t}{c^*}\right]\Delta t \tag{5.27}$$

where the superscript "n" indicates the evaluation of the parameter in the time step t^n, the superscript "$n+1$" indicates the evaluation of the parameter in the time step t^{n+1} and Δt is the time step. In this case, the parameters at the instant "n" are known and the only unknown is the parameter P_c^{n+1}.

Due to the initial equilibrium pressure is not known, it is necessary to initiate the calculation of (5.27) with an initial guess $P_c^{n=0} \rightarrow P_c(t=0)$. Moreover, the time step Δt shall be so small for the first second of calculation because of the transient. If the time step is not small enough, the expression (5.27) becomes unstable and divergent. Thereby, for the first phase of the rocket performance (ignition transient) it is necessary to use a very small time step. Once the ignition transient has passed, the time step can be increased. In order to consider this fact in the code of the 0D unsteady flow simulation, two different time steps are implemented. The first time step is very small and controls the first phase of the rocket performance. This first phase lasts less than a second. The second time step is greater and it is used to calculate the rest of the SRM operation.

5.3. QUASY-STEADY 1D FLOW MODEL

In this section the treatment of the 1D quasi-steady flown is put forward. As it has been mentioned in the scope of the project, this study does not contemplate the development of a 1D flow model. Therefore, the models here attached are found from the literature. This way, different manners to describe the 1D quasi-steady flow can be found. For this study, two models can be considered: the conservative form developed by Stewart et al [13] and the non-conservative form developed by Yildirim [9].

In case of conservative form, Stewart et al [13] propose a simplified model to study SRM. This way, the authors determine a cross-section (gas core) of the rocket given by $A(z,t)$, where z is the distance along the axis of the rocket and t is the time. So, the proposed governing equation for a SRM with slowly varying cross-section is:

$$\frac{\partial}{\partial t}\begin{bmatrix}\rho A\\ \rho u A\\ \rho E A\end{bmatrix}+\frac{\partial}{\partial z}\begin{bmatrix}\rho u A\\ (\rho u^2+P)A\\ u(\rho E+P)A\end{bmatrix}=\begin{bmatrix}\rho_p \dot{r} A\\ P\dfrac{\partial A}{\partial z}\\ \rho_p \dot{r}_b P_b\left(h_f+\dfrac{1}{2}u_f^2\right)\end{bmatrix} \tag{5.28}$$

where E is the total energy and given by $E=P/\big(\rho(\gamma-1)\big)+u^2/2$, P_b is the burning perimeter, u is the gas velocity, P is the static pressure of the gas, ρ is the density of the gas, ρ_p is the propellant density, $\dot{r}$ is the burning rate, h_f is the enthalpy of the combustion products and u_f is the injection velocity of the combustion products.

Due to the burning rate is so slow compared to the flow field velocity, it can be assumed that for each time step the grain geometry is frozen, perform the flow calculation, and after that, propagate the grain geometry another step. Hence, from the flow point of view, the grain geometry is frozen. The grain burning rate is of the order of 10^{-2}m/s while the flow speed of sound is of the order of 10^{3}m/s, so the time a flow particle spends inside the combustion chamber is too short to feel the effect of the variation in the grain geometry. The scale time of the flow and the grain burnback are so different that they can be handled separately. So that, it can be considered that the flow is in steady state, solving it for a given grain geometry, and once it is solved, perform another grain burnback step, and calculate the flow again with this frozen configuration. Thereby, considering $A(z,t)\approx A(z)$, the equation (5.28) is rewritten as:

$$\frac{\partial}{\partial z}\begin{bmatrix}\rho uA\\(\rho u^2+P)A\\u(\rho E+P)A\end{bmatrix}=\begin{bmatrix}\rho_p \dot{r}A\\P\dfrac{\partial A}{\partial z}\\\rho_p \dot{r}_b P_b\left(h_f+\dfrac{1}{2}u_f^2\right)\end{bmatrix} \tag{5.29}$$

With respect to non-conservative form, Yildirim [9] first develops the 1D unsteady model and then, some simplifications to implement the 1D quasi-steady model are assumed. This way, the author states that the flow in the chamber is assumed to be frozen and the variations of the flow parameters with the time may be neglected. Additionally, for simplicity, shear stress and heat loss are not considered and perfect gas law is used. Subsequently, the author after applying the simplifications and rearranging the non-conservative forms of mass conservation, momentum conservation and energy conservation equations, states that the equations describing the 1D quasi-steady model are:

$$\frac{d(\rho u)}{dz}=-\frac{\rho u}{A_p}\frac{dA_p}{dx}+\frac{\dot{r}_b P_b}{A_b}\rho_p \tag{5.30}$$

$$\frac{du}{dz}=\frac{1}{1-M^2}\frac{\rho_p}{\rho}\frac{\dot{r}_b P_b}{A_p}\left((\gamma-1)\frac{h_f}{a^2}+\frac{\gamma+1}{2}M^2-\frac{\rho}{\gamma\rho_p}\right)-\frac{u}{(1-M^2)A_p}\frac{dA_p}{dz} \tag{5.31}$$

$$\frac{dp}{dz}=-\frac{u\rho_p}{A_p}\dot{r}_b P_b\frac{1}{1-M^2}\left(1+(\gamma-1)\frac{h_f}{a^2}+\frac{\gamma-1}{2}M^2-\frac{\rho}{\gamma\rho_p}\right)+M^2\frac{\gamma P}{(1-M^2)A_p}\frac{dA_p}{dz} \tag{5.32}$$

where z is the longitudinal axis of the SRM, ρ is the gas density in kg/m^3, u is the velocity of the gas in m/s, A_p is port area en m^2, $\dot{r}_b$ is the burning rate in m/s, P_b is the burning perimeter in m, ρ_p is the propellant density in kg/m^3, M is the Mach number, a is the sonic speed in m/s equal to $a=\sqrt{\gamma RT}$, h_f is the enthalpy of the combustion products in J/kg and P is the static pressure of the gas in Pa.

In order to solve the flow inside the solid rocket motor the equations (5.30), (5.31) and (5.32) have to be integrated from the head end to the nozzle inlet. However, at the region where there is no side wall injection (out of the grain region), the terms related with the propellant grain disappear. As well, the boundary conditions at the motor head end are $u(z=0)=0$, $\left.\frac{dp}{dz}\right|_{z=0}=0$ and the initial guess pressure $P(z=0)=P_{0\_guess}$. On the other side, the boundary condition at the nozzle inlet is that the Mach number must match that obtained with the 1D nozzle theory (M in equation (5.33)):

$$\frac{A_{inlet}}{A_t} = \frac{1}{M}\left(\frac{1+\frac{\gamma-1}{2}M^2}{\frac{\gamma+1}{2}}\right)^{\frac{\gamma+1}{2(\gamma-1)}} \tag{5.33}$$

It should be pointed out that the pressure at the head end is unknown, so an iteration process shall be done using $P(z=0)$ until the Mach number at the nozzle inlet is approximately the same to that obtained with the 1D nozzle theory (with a certain precision). This iteration procedure is proposed in [16]. Once the Mach number at the nozzle inlet matches the nozzle inlet boundary condition, the nozzle can be calculated using the 1D nozzle relations such as equations (2.18) and (5.33).

In this study, this iteration process has not been implemented, and only the integration of the equations (5.30), (5.31) and (5.32) from the head end to the nozzle inlet has been carried out. The chamber pressure obtained from the 0D unsteady model has been used as the initial guess for the head end pressure. So that, because the iteration with $P(z=0)$ is not done, the Mach number at the nozzle inlet does not match the boundary condition, and hence the results obtained are only qualitative (they show the trend but are not the correct values). Therefore, this study is based on a Quasi-3D grain burnback analysis model coupled with a 0D unsteady flow model. The implementation of the 1D quasi-steady flow model is not complete and so the results of the 1D model are only qualitative.

Finally, in this study, the methods from Stewart et al and Yildirim have been tested. During these tests, it has been noticed that the Stewart et al method has some problems caused by the port area term and its derivative. It should be remarked that in the Quasi-3D burnback analysis approach, the port area term evolves sharply because the grain is modeled with just few cross-sections. So that, the Quasi-3D model provides the flow solver an input which is sharp. Thus, any flow model will experience some problems due to this sharp input. This way, due to the Stewart et al model deals with the terms in conservative form, it is more sensible to sharp inputs. On the other side, this problem is also experienced by the Yildirim method, but it is less sensible to it because its formulation is in non-conservative form. Therefore, the model used in this study is the Yildirim's because it handles better the term of the port area. To sum up, in order to solve the problem of this sharp input and its effect on the flow solver, different interpolation methods could be studied in a future work with the aim of smoothing the port area term.

6. CODE OVERVIEW

This chapter presents an overview of the code implemented in the project *Study of Grain Burnback and Performance of Solid Rocket Motors*. For that, firstly the algorithms of the grain burnback analysis and the SRM internal ballistics are explained. From this process, three different simulation tools have been developed, which are: 2D grain burnback with 0D unsteady flow simulation tool, Quasi-3D grain burnback with 0D unsteady flow simulation tool and Quasi-3D grain burnback with 1D quasi-steady flow simulation tool. Each simulation tool requires its specific inputs as well as provides its particular outputs or results. The three simulation tools have been implemented using Matlab® software.

Following, a brief organization of the chapter is presented. Firstly, the grain burnback and ballistics algorithm is presented, specifying the coupling used in the first two simulation tools aforementioned. Then, the input parameters required by each simulation tool are identified. Later on, the outputs given by the three codes are recognized. Finally, the functions implemented in the three codes to express the mathematical background of each model are shown. Note that the full Matlab® code of the three simulation tools is enclosed in the ANNEX A.

6.1. GRAIN BURNBACK AND BALLISTICS ALGORITHM

In this section the grain burnback and ballistics algorithm is presented. Basically, this section explains how the effects of grain burnback and the different flow models are joined. This way, three different combinations are implemented in the numerical code: 2D grain burnback and 0D unsteady flow, Quasi-3D grain burnback and 0D unsteady flow and Quasi-3D and 1D quasi-steady flow. Note that the 0D steady flow model (lumped model) is not used. This flow model is basic to structure the 0D unsteady flow model but too simple to explain properly the internal ballistics of a SRM. Thereby, the numerical implementation of 0D steady flow model has been ruled out from the beginning and it has gone straight forward to the numerical implementation of 0D quasi-steady flow model. In addition, the numerical employment of 2D grain burnback and 1D quasi-steady flow model has nonsense. The minimum requirement of the numerical application of a 1D quasi-steady flow model is the use of Quasi-3D model to describe the grain burnback. Lastly, before implementing the Quasi-3D grain burnback and 1D quasi-steady flow, the application of the Quasi-3D grain burnback and 0D unsteady flow is a fundamental step. This way, the Quasi-3D model is verified properly.

As it has been aforementioned, the evolution of the grain burning surface has a determining influence on the performance of a solid rocket motor. Actually, the SRM performance is linked to the grain burnback analysis through the internal ballistics simulation (internal flow in the SRM). In fact, the internal ballistics simulation requires inputs from the grain burnback analysis such as the burning area, port area, chamber volume, inter-alia. Yet, this relation is reciprocal because in turn, the propagation speed of the burning surface, namely the burning rate, is determined by the internal flow of the SRM. So, the grain burnback analysis and the internal ballistics simulation are coupled. This way, while the first two simulation tools the coupling is fully considered, in case of Quasi-3D grain burnback and 1D quasi-steady flow simulation tool, the coupling between the grain burnback and the internal ballistics is not completely implemented. That is why the algorithm of this last case is not considered as “coupling”.

6.1.1. 2D GRAIN BURNBACK AND 0D UNSTEADY FLOW COUPLING

In this section the coupling of 2D grain burnback and 0D unsteady flow is presented. As it has been aforementioned, the grain burnback is coupled with the internal ballistic through the burning rate. Meanwhile, the internal ballistics is coupled with the grain burnback analysis through the grain geometrical properties. The algorithm of the 2D grain burnback and 0D unsteady flow simulation tool is shown in Figure 42. The program continues calculating until the simulation time is reached. Consequently, while assigning any value for the simulation time, the burnout can be reached or not. If the burnout is reached but not the simulation time, the program continues calculating. Nevertheless, for instance, the outputted thrust in that case will be equal to zero, because the burnt area will be zero. On the contrary, if the burnout is not reached but the simulation time ends, the program will give the last step of the grain burning surface as well as the last values of the performance parameters. Note that the simulation time required to achieve the burnout depends on the inputted grain geometry and propellant parameters.

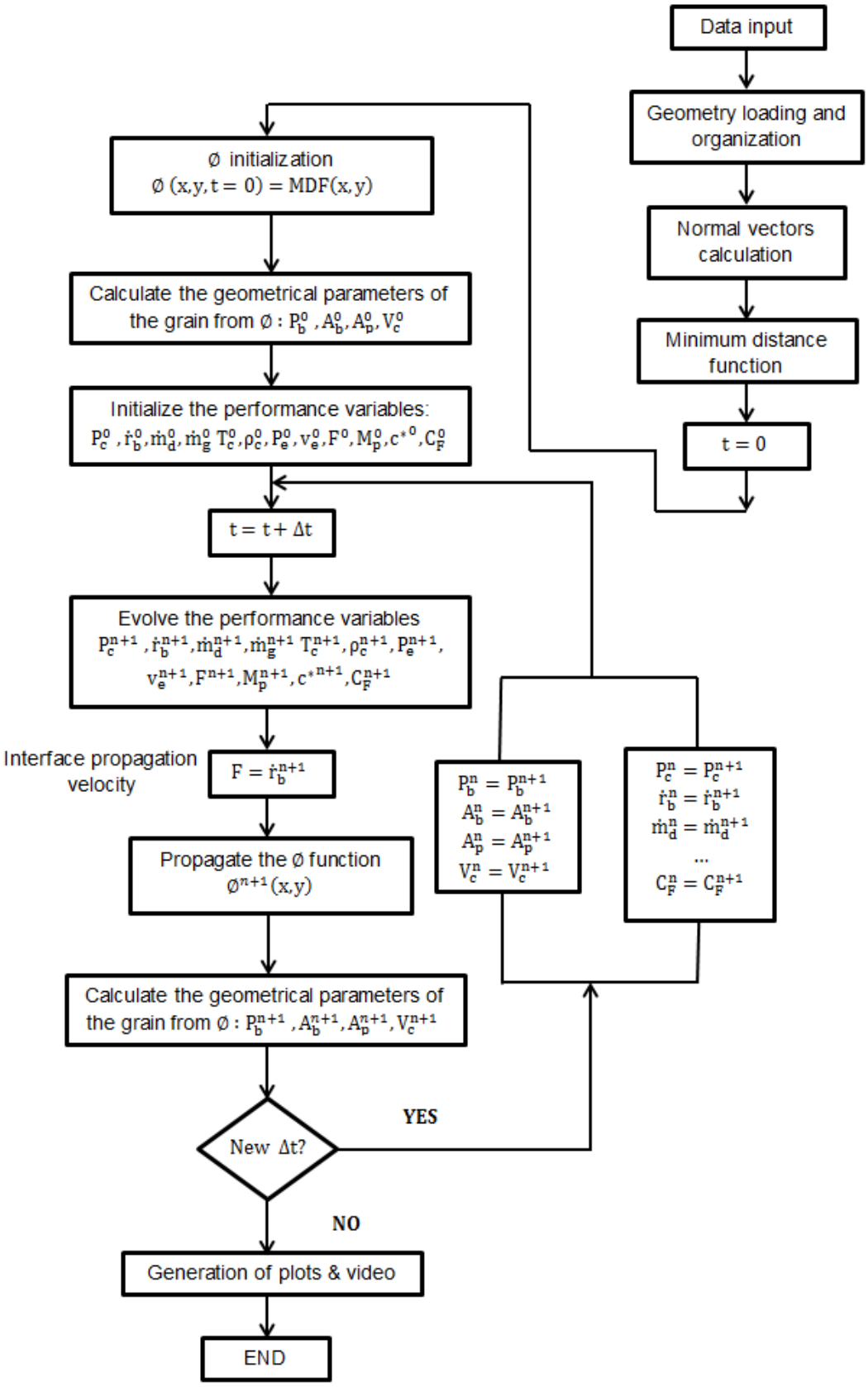

Figure 42. Algorithm of the 2D grain burnback and 0D unsteady flow simulation tool.

6.1.2. QUASI-3D GRAIN BURNBACK AND 0D UNSTEADY FLOW COUPLING

In this section the coupling of Quasi-3D grain burnback and 0D unsteady flow is presented. Analogously that in case of 2D grain burnback and 0D unsteady flow, the grain burnback is coupled with the internal ballistics through the burning rate. Conversely, the internal ballistics is coupled with the grain burnback though the grain geometrical properties. The algorithm of this simulation tool is shown in Figure 43. The program continues calculating until the simulation time is reached. Consequently, while assigning any value for the simulation time, the burnout can be reached or not. The outputted performance parameters will be equal to zero in the remaining time after the burnout has been reached but not the simulation time. Conversely, if the simulation time is not reached but the burnout is, the program will give the last values of the internal ballistics parameters and the last step of the grain burning surface. In this case too, the simulation time where burnout takes place depends on the introduced grain geometry and propellant parameters. As it has been aforementioned, in the Quasi-3D model the ϕ function is propagated for each reference cross-section. For further information, consult the section 4.3.4 of this report.

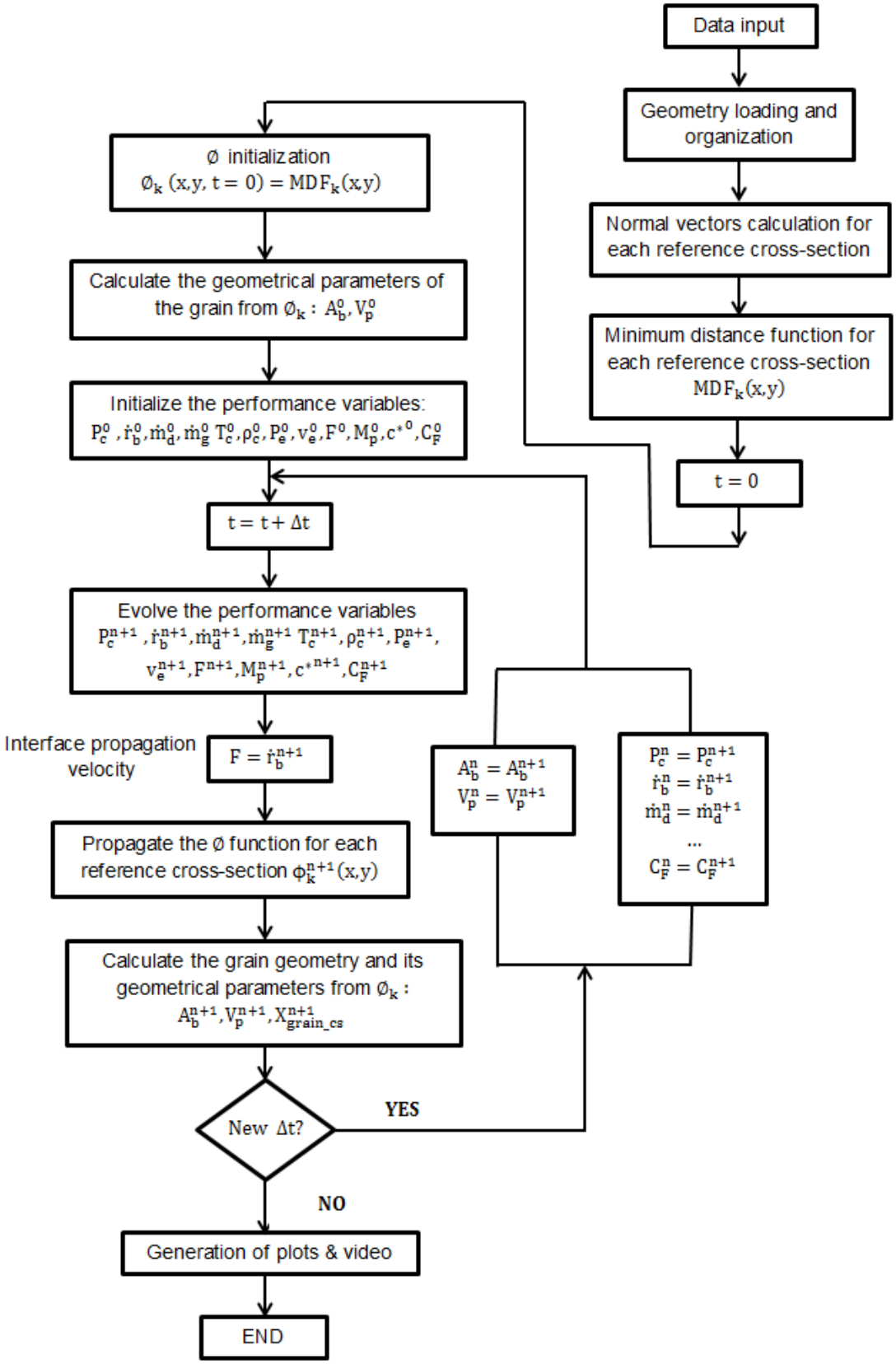

Figure 43. Algorithm of the Quasi-3D grain burnback and 0D unsteady flow simulation tool.

6.1.3. QUASI-3D GRAIN BURNBACK AND 1D QUASI-STEADY FLOW ALGORITHM

In Figure 43 the algorithm of Quasi-3D grain burnback and 1D quasi-steady flow is presented. In contrast with the two previous simulation tools, in this one the coupling of the grain burnback and the 1D flow model is not done. Indeed, for this case, the Quasi-3D grain burnback analysis is coupled with the 0D unsteady flow model, and the 1D model is calculated using the data obtained by this coupling. As it has been mentioned in section 5.3, the quasi-steady 1D flow model has been partially implemented towards showing to the user the trends of how the flow evolves inside the solid rocket motor. Thereby, the results obtained by the 1D model are only qualitative, but illustrative of what happens inside the SRM. So that, the burning rate is spatially constant because the 0D flow model is coupled with the grain burnback analysis.

In this case too, the simulation time is the parameter that controls the ending of the calculation process. This way, in case that the burnout is not reached but the simulation time has, the code gives as result the last burning surface step as well as the parameters of the internal ballistics. On the contrary, if the burnout has been reached but not the simulation time, the output results will go to zero.

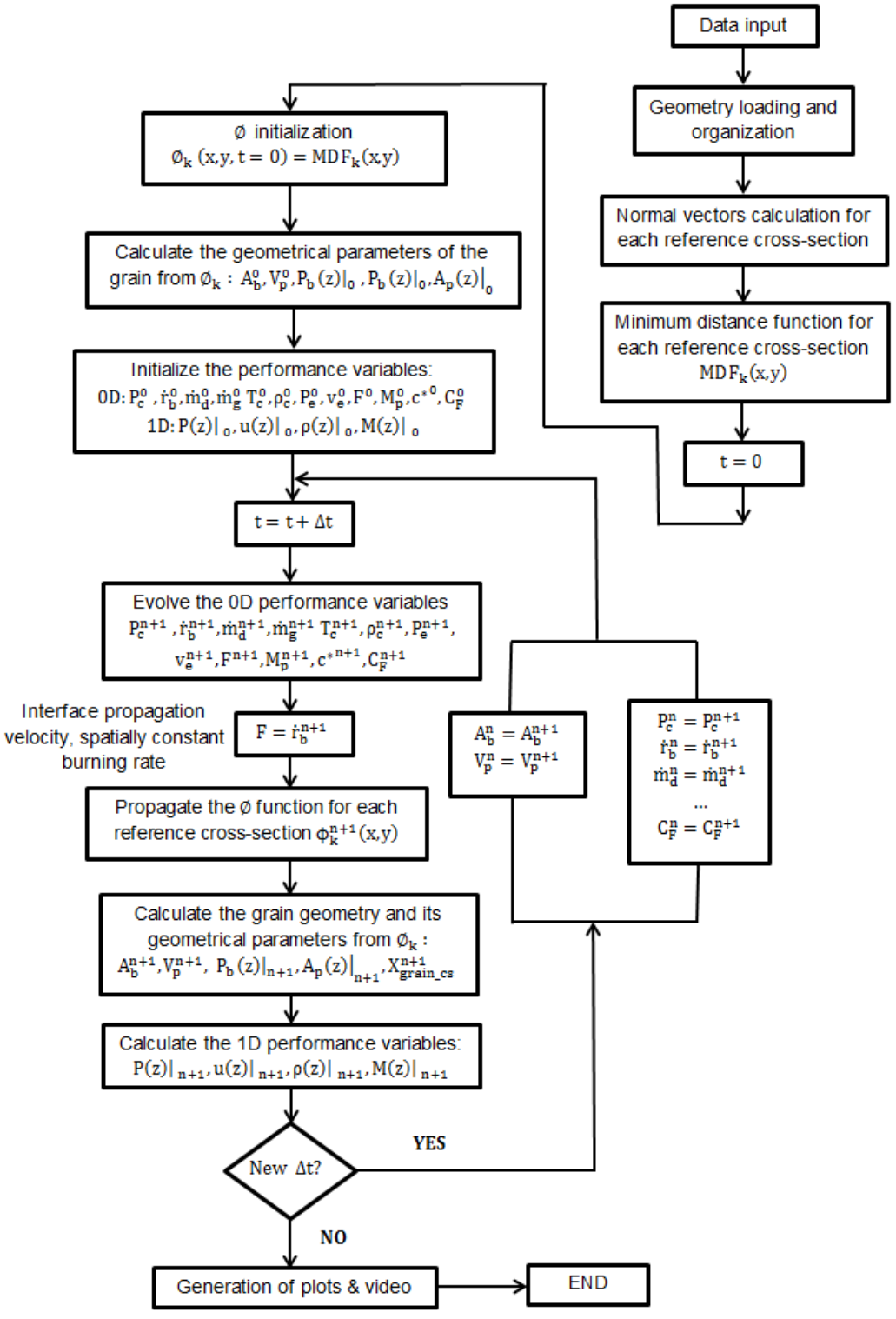

Figure 44. Algorithm of the Quasi-3D grain burnback and 1D quasi-steady flow simulation tool.

6.2. INPUTS

This section explains the inputs required by the three simulation tools. In order to classify the inputs, three categories can be distinguished: the geometry data, the SRM related parameters data and the numerical parameters.

The geometry data is the most important input data. As it has aimed in the requirements of this study, the user is able to introduce arbitrary grain geometry data and obtain the SRM performance from the simulation tool. This grain geometry can be whether 2D or 3D. Nevertheless, it should be pointed out that the data shall be discretized data; so is to say that a cloud of points describing the desired grain geometry. Then this cloud of points, saved in text file format, is uploaded in the Matlab® code. Although the cloud of points can be obtained by the user following different techniques, a recommended manner is the process followed in this study to verify the code and enclose the results given in section 7. This way, in order to obtain the input geometry data, first a CAD (Computer-aided Design) design of the desired grain geometry is realized. For that, CATIA v5R18 software has been used. Once the design is finished, it is saved as IGES (Initial Graphics Exchange Specification) file. This file is uploaded by the GID 10.0.7 software in order to perform the discretization of the geometry. This software, as it has been aforementioned in the acknowledgements, has been freely provided by the International Center for Numerical Methods in Engineering (CIMNE).

Then, after the discretization is done, the generated geometry points are saved in a text file (".txt"). As mentioned before, these text files are uploaded by the simulation tools generated through the Matlab® software. Note that the generated geometry points depend on the discretization values selected in the GID. Therefore, the dimension of the text file is greater when more discretization points are selected in the generation process because the cloud of points studied is bigger. So, this discretization process results highly depend on the user-expertise.

As mentioned before, the arbitrary input grain geometries can be whether 2D or 3D. In case of 2D geometries, the grain and the motor case are designed in CAD, discretized, saved as text files and then, uploaded in Matlab®. It should be pointed out that in the 2D designs the geometry text file contains the points of the grain geometry as well as the motor case. Note that the motor case is a circle of a given radius. Hence, the Matlab® code reads the geometry text file and then generates two different matrixes: a matrix (x-y coordinates) with the geometry of the grain and a matrix with the geometry of the motor case.

```
%Section A
%       X              Y              Z
  -4.34494e+01   -2.10043e+01    4.86260e+00
  -4.36670e+01   -2.05481e+01    4.86260e+00
  -4.32270e+01   -2.14581e+01    4.86260e+00
  -4.38797e+01   -2.00897e+01    4.86260e+00
  -4.30000e+01   -2.19096e+01    4.86260e+00
  -4.40877e+01   -1.96291e+01    4.86260e+00
  -4.27682e+01   -2.23587e+01    4.86260e+00
  -4.42908e+01   -1.91664e+01    4.86260e+00
  -4.25317e+01   -2.28053e+01    4.86260e+00
  -4.44891e+01   -1.87015e+01    4.86260e+00
  -4.22906e+01   -2.32494e+01    4.86260e+00
  -4.46825e+01   -1.82346e+01    4.86260e+00
  -4.20448e+01   -2.36910e+01    4.86260e+00
```

Figure 45. Example of the beginning part of a 3D geometry input data file.

With respect to 3D geometries, the grain design and the motor case profile geometries are given in separated text files. Moreover, the 3D grain is divided in several cross-sections along the motor longitudinal axis. These cross-sections are designed in CAD and then saved as IGES and then, they are discretized in similar way than the 2D grain designs. After the discretization, the clouds of points corresponding to several cross-sections are saved in a unique text file. This way, one text file contains the information of the entire 3D grain design, where coordinate values are separated in columns as it is shown is Figure 45 and different cross-sections are stored consecutively, one after another. So, in order to be able to handle the data by the Matlab® simulation tools, it is required to establish a criterion. In case of this project, at the end of each cross-section data, two rows of zeros have been included as it is shown in Figure 46. In this manner, while loading the text files, if the Matlab code identifies two consecutive rows of zeros, it notices that a new cross-section of the 3D grain design will come next. Finally, with respect to motor case profile, which shall be saved in a separate text file, its axisymmetric properties are exploited. Thereby, in order to describe its geometry only the revolution profile about the longitudinal axis is needed. Thus, it is only necessary to design a 2D profile of the motor case in the CAD software. Then, the process is repeated, by saving this CAD file as IGES file, discretizing it through the GID software, saving the cloud of discretized point in a text file and finally, uploading it in Matlab®.

```
   4.38797e+01    2.00897e+01    4.86260e+00
   4.30000e+01    2.19096e+01    4.86260e+00
   4.36670e+01    2.05481e+01    4.86260e+00
   4.32270e+01    2.14581e+01    4.86260e+00
   4.34494e+01    2.10043e+01    4.86260e+00
   0.00000e+00    0.00000e+00    0.00000e+00
   0.00000e+00    0.00000e+00    0.00000e+00
%Section B
%       X              Y              Z
  -2.14274e+01   -1.03584e+01    3.02260e+01
  -2.15347e+01   -1.01335e+01    3.02260e+01
  -2.13178e+01   -1.05822e+01    3.02260e+01
  -2.16396e+01   -9.90740e+00    3.02260e+01
  -2.12058e+01   -1.08049e+01    3.02260e+01
  -2.17422e+01   -9.68025e+00    3.02260e+01
  -2.10915e+01   -1.10264e+01    3.02260e+01
  -2.18424e+01   -9.45204e+00    3.02260e+01
  -2.09748e+01   -1.12466e+01    3.02260e+01
```

Figure 46. Image of the two-row-zeros criterion, which is established in order to distinguish the different cross-sections in the 3D grain geometry input text files.

The SRM related parameters data and the numerical parameters are inputted in the Matlab® code directly. Precisely, these parameters have to be defined by the user in the first code lines of the main codes of the three simulation tools. Although at first sight this way of inputting the parameters is not very user-friendly, it is the quickest way to do it not increasing the coding lines. Nevertheless, creating for instance a Graphical User Interface (GUI) in Matlab® is out of scope of this project.

In the following sub-points of this section, the nomenclature for the three types of input data (the geometry data, the SRM related parameters data and the numerical parameters data) used in each simulation tool is identified. This way, for each simulation tool the identification symbols used in the main code as well as the meaning and units of each parameter are enclosed.

6.2.1. INPUTS FOR 2D GRAIN BURNBACK AND 0D UNSTEADY FLOW SIMULATION

As it has been aforementioned, there are three kinds of input data: geometry data, SRM related parameter data and numerical parameters data. For this simulation tool, 2D grain burnback is considered. Therefore, the geometry data inputted corresponds with the discretized 2D grain design. Precisely, the clouds of points obtained from the discretization of 2D designs and saved in text files are uploaded by the Matlab® code. The recommended process to create the

discretization points is explained in the introduction of 6.1.3. Although the user can define arbitrary 2D grain designs, in this study only the most significative ones have been analysed. Specifically, the 2D grain geometries used as inputs during this study are: the circular, the star of 10 slotted tips, the tri-slot, the dogbone and the anchor. The text files containing the cloud of points of these 2D geometries are enclosed inside the CD of this project. Precisely, there are located into the folder of 2D grain burnback and 0D unsteady flow simulation tool. They are not copied in paper version because each discretized 2D geometry contains more than 1000 points.

The SRM related parameters and numerical data should be directly defined in the code General_2D_discrete_cart_coord_0D_flow.m by the user. Precisely, this code is the main code of this 2D grain burnback and 0D unsteady flow simulation tool. This way, the input SRM related input parameters are identified in Table 1 and the numerical parameters in the Table 2. In order to see the Matlab® code where they are defined, please consult the section A.1.1. of the ANNEX A.

SRM related parameters		
Symbol	**Description**	**Units**
L_g	Propellant grain length.	m
P_c0	Initial chamber pressure.	Pa
n_exp	n exponent of the propellant burning rate law.	-
a	a parameter of the propellant burning rate law (burning rate in m/s and chamber pressure in Pa).	-
gamma_e	Combustion products gas gamma parameter.	-
R_air	R of the air.	J/kg/K
R_e	R of the combustion products.	J/kg/K
rho_p	Propellant density.	kg/m^3
M_mol	Combustion products molar mass.	kg/mol
De	Nozzle exit diameter.	mm
Dt	Nozzle throat diameter.	mm
L_plenum	Plenum length.	m
P_a	Atmospheric pressure.	Pa
T_c_v	Vector of chamber temperature data relation with chamber pressure.	K
P_c_v	Vector of chamber pressure data.	Pa

Table 1. SRM related input parameters of 2D grain burnback and 0D unsteady flow.

Numerical parameters		
Symbol	**Description**	**Units**
N	Number of nodes per side in the Cartesian grid.	-
N_Delta_web	Number of contour lines shown in the ϕ plot.	-
N_bs_plot	Number of burnt steps shown in the grain evolution plot.	-
Time	Simulation time.	s
n_steps	Number of simulation time steps.	-
t_0	Time duration of the first phase of the simulation (heavy transient).	s
Delta_t0	Time step during t_0 phase of the simulation.	s
units	The distance units used in the geometry.	'm', 'cm', 'mm'
atm_units	Conversion factor from Pascal to atmosphere.	101325

Table 2. Numerical input parameters of 2D grain burnback and 0D unsteady flow.

6.2.2. INPUTS FOR QUASI-3D GRAIN BURNBACK AND 0D UNSTEADY FLOW SIMULATION

In case of this simulation tool there are three kinds of input data too: geometry data, SRM related parameter data and numerical parameters data. For this simulation tool, 3D grain burnback is considered. Therefore, the geometry data inputted corresponds with the discretized 3D grain design as well as with the discretized motor case profile. The clouds of points obtained from the discretization of both designs are saved in text files in order to be uploaded by the Matlab® code. The recommended process to create the discretization points for 3D grain designs is explained in the introduction of 6.1.3 as well as the implications for saving the cross-sections' clouds of points. In this case the user is allowed to define arbitrary 3D grain designs and motor case profiles too. Nevertheless, in this project only the most significative ones have been analysed. This way, the studied SRM 3D configurations are: the NAWC motor nº6 and the NAWC motor nº13. The grain design and motor configuration of these SRM can be found in 6.1.3. Consequently, in order to imitate the real configuration of these motors, a CAD design for each one has been done. From these CAD designs, following the procedure explained in the introduction of 6.1.3, several cross-section have been specified. Then, this cross-section have been discretized using the software 10.0.7 for MacOSX and saved in a unique text file. In addition, the motor case profile has been discretized and saved in another text

file too. Thereby, for each SRM, there are two text files: one containing the cloud of points corresponding to discretized cross-sections and the other containing the cloud of points corresponding to the motor-case. These four text files are included inside the CD of this project; precisely into the folder of Quasi-3D grain burnback and 0D unsteady flow simulation. There are not copied in the paper version because the file that describes the grain geometry contains more than 10,000 points.

The SRM related parameters and numerical data are necessary to be direclty determined in the the main code of this simulation tool by the user. The name of the main code is General_Quasi_3D_discrete_cart_coord_0D_flow.m and its Matlab® code can be found in the section A.2.1. of the ANNEX A. A sucint summary of the SRM related parameters inputted in this code is found in the Table 3. The numerical parameters are defined in the Table 4.

SRM related parameters		
Symbol	**Description**	**Units**
P_c0	Initial chamber pressure.	Pa
n_exp	n exponent of the propellant burning rate law.	-
a	a parameter of the propellant burning rate law (burning rate in m/s and chamber pressure in Pa).	-
gamma_e	Combustion products gas gamma parameter.	-
R_air	R of the air.	J/kg/K
R_e	R of the combustion products.	J/kg/K
rho_p	Propellant density.	kg/m^3
M_mol	Combustion products molar mass.	kg/mol
theta_min_case	Initial angle for the plot of the motor case.	º
theta_max_case	Final angle for the plot of the motor.	º
P_a	Atmospheric pressure.	Pa
T_c_v	Vector of chamber temperature data relation with chamber pressure.	K
P_c_v	Vector of chamber pressure data.	Pa

Table 3. SRM related input parameters of Quasi-3D grain burnback and 0D unsteady flow.

Numerical parameters		
Symbol	**Description**	**Units**
N	Number of nodes per side in the Cartesian grid.	-
N_Delta_web	Number of contour lines shown in the ϕ plot.	-
N_bs_plot	Number of burnt steps shown in the grain evolution plot.	-
N_cs_i	Number of grain cross-sections interpolated between each reference cross-section.	-
Time	Simulation time.	s
n_steps	Number of simulation time steps.	-
t_0	Time duration of the first phase of the simulation (heavy transient).	s
Delta_t0	Time step during t_0 phase of the simulation.	s
K_size	Time step for the size of Xgrain_cs.	-
units	The distance units used in the geometry.	'm', 'cm', 'mm'
atm_units	Conversion factor from Pascal to atmosphere.	101325

Table 4. Numerical input parameters of Quasi-3D grain burnback and 0D unsteady flow.

6.2.3. INPUTS FOR QUASI-3D GRAIN BURNBACK AND 1D QUASI-STEADY FLOW SIMULATION

In this simulation tool there are also three kinds of input data: geometry data, SRM related parameter data and numerical parameters data. Note that in case of this simulation tool, 3D grain burnback is considered. Therefore, the geometry data inputted corresponds with the discretized 3D grain design as well as with the discretized motor case profile. The clouds of points obtained from the discretization of both designs are saved in text files in order to be uploaded by the Matlab® code. The recommended process to create the discretization points for 3D grain designs is explained in the introduction of 6.1.3 as well as the implications for saving the cross-sections' clouds of points. In thi simulation tool the user is allowed to define arbitrary 3D grain designs and motor case profiles too. Nevertheless, in this project only the most significative ones have been analysed. This way, the studied SRM 3D configurations are: the NAWC motor nº6 and the NAWC motor nº13. The grain design and motor configuration of these SRM can be found in 6.1.3. Consequently, in order to imitate the real configuration of these motors, a CAD design for each one has been done. From these CAD designs, following the procedure explained in the introduction of

6.1.3, several cross-section have been specified. Then, this cross-section have been discretized using the software 10.0.7 for MacOSX and saved in a unique text file. In addition, the motor case profile has been discretized and saved in another text file too. Thereby, for each SRM, there are two text files: one containing the cloud of points corresponding to discretized cross-sections and the other containing the cloud of points corresponding to the motor-case. These four text files are included inside the CD of this project; precisely into the folder of Quasi-3D grain burnback and 1D quasi-steady flow simulation. There are not copied in the paper version because the file that describes the grain geometry contains more than 10,000 points.

The SRM related parameters and numerical data are necessary to be direclty determined in the the main code of this simulation tool by the user. The name of the main code is General_Quasi_3D_flow_1D_discrete_cart_coord.m and its Matlab® code can be found in the section A.3.1. of the ANNEX A. A short summary of the SRM related parameters inputted in this code is found in the Table 5. The numerical parameters are defined in the Table 6.

SRM related parameters		
Symbol	**Description**	**Units**
P_c0	Initial chamber pressure.	Pa
n_exp	n exponent of the propellant burning rate law.	-
a	a parameter of the propellant burning rate law (burning rate in m/s and chamber pressure in Pa).	-
gamma_e	Combustion products gas gamma parameter.	-
R_air	R of the air.	J/kg/K
R_e	R of the combustion products.	J/kg/K
rho_p	Propellant density.	kg/m^3
M_mol	Combustion products molar mass.	kg/mol
theta_min_case	Initial angle for the plot of the motor case.	º
theta_max_case	Final angle for the plot of the motor.	º
P_a	Atmospheric pressure.	Pa
T_c_v	Vector of chamber temperature data relation with chamber pressure.	K
P_c_v	Vector of chamber pressure data.	Pa

Table 5. SRM related input parameters of Quasi-3D grain burnback and 1D quasi-steady flow.

Numerical parameters		
Symbol	**Description**	**Units**
N	Number of nodes per side in the Cartesian grid.	-
N_Delta_web	Number of contour lines shown in the ϕ plot.	-
N_bs_plot	Number of burnt steps shown in the grain evolution plot.	-
N_cs_i	Number of grain cross-sections interpolated between each reference cross-section.	-
N_z1	Number of slices for the flow calculation from the head end to the nozzle inlet.	-
N_z2	Number of slices for the flow calculation from the nozzle inlet to the outlet.	-
epsilon	Precision of the calculation for the flow 1D solver.	-
fr	Relaxation factor for the calculation.	-
Time	Simulation time.	s
n_steps	Number of simulation time steps.	-
t_0	Time duration of the first phase of the simulation (heavy transient).	s
Delta_t0	Time step during t_0 phase of the simulation.	s
K_size	Time step for the size of Xgrain_cs.	-
units	The distance units used in the geometry.	‘m’, ‘cm’, ‘mm’
atm_units	Conversion factor from Pascal to atmosphere.	101325

Table 6. Numerical input parameters of Quasi-3D grain burnback and 1D quasi-steady flow.

6.3. OUTPUTS

This section explains the outputs given by the three simulation tools. In order to classify the outputs, two categories can be distinguished: the images that help to understand what happens during the grain burnback and the graphics that show the evolution of SRM performances during its operation. Note that the images obtained can be whether static or dynamics. In this way, in the following sub-sections for each simulation tool the images obtained as well as the plotted parameters are defined.

6.3.1. OUTPUTS FOR 2D GRAIN BURNBACK AND 0D UNSTEADY FLOW SIMULATION

When the main code of this simulation tool, called General_2D_discrete_cart_coord_0D_flow.m (see section A.1.1. of the ANNEX A) is run, the outputs appeared on the computer screen. These outputs can be classified as images or graphics. On the one hand, the outputted images are aimed to clarify to the reader the grain burnback; that is to say, what happens with the grain during the operation of the SRM. These outputted images can be classified as static or dynamic. The static images do not change during the plotting process, they are always the same. Meanwhile, the dynamic images evolve at the same rate that does the grain burnback. Into this latter subgroup, the videos or the evolving 2D images are found. This way, various kinds of output images given by this 2D grain burnback and 0D unsteady flow code are summarized in the Table 7. Note that the videos obtained from this simulation tool are attached in the subfolder "2D Grain Burnback and 0D Unsteady Flow" of the folder "Videos" found inside the CD of this project.

Name	Type	Description
Minimum distance function	Static	It shows the contour of the Minimum Distance Function for a given 2D grain design.
Grain initial geometry	Static	It shows the initial discretized grain geometry in order to corroborate that it is well uploaded in the code.
Grain evolution during the rocket burning	Static	It shows the zero level set (interface) curves at given time steps.
Grain evolution during the rocket burning (video)	Dynamic	It shows the evolution of the level set curve during the grain burnback. The area between the level set curve and the motor case is not filled.
Grain evolution during the rocket burning (filled video)	Dynamic	It shows the evolution of the level set curve during the grain burnback. The area between the level set curve and the motor case is filled.

Table 7. Output images given by the 2D grain burnback and 0D unsteady flow code.

On the other hand, the graphics are intended to show the SRM performance evolution during the operation. That is why they are used to plot different parameters evolution during the grain burnback. Note that the parameters shown in this graphs are the reference to compare different grain configurations. The output graphics given by the 2D grain burnback and 0D unsteady flow simulation tool are explained in Table 8. Finally, it should be pointed out that almost all of

the different parameters plotted in the graphs are expressed according to SI. The only exceptions are the parameters related with the input “units” of Table 6, where the user can define the dimensional units of the geometry.

Name	Description	Units
A_b vs. t	Burning area time evolution.	A_b in m^2 ; t in s
m_{dotg} vs. t	Mass flow rate due to propellant combustion time evolution.	m_{dotg} in kg/s ; t in s
P_c vs. t	Combustion chamber pressure time evolution.	P_c in Pa ; t in s
m_{dotd} vs. t	Mass flow rate exiting the nozzle time evolution.	m_{dotd} in kg/s ; t in s
r_{dot} vs. t	Burning rate time evolution.	r_{dot} in mm/s ; t in s
Perimeter vs. t	Burning perimeter time evolution.	Perimeter in ‘units of Table 6’ ; t in s
A_p vs. t	Port area time evolution.	A_p in ‘units of Table 6^2’ ; t in s
T_c vs. t	Combustion temperature time evolution.	T_c in K ; t in s
F vs. t	Thrust time evolution.	F in N ; t in s
V_{exit} vs. t	Nozzle exit velocity time evolution.	V_{exit} in m/s ; t in s
ρ_c vs. t	Combustion chamber gas density time evolution.	ρ_c in kg/m^3 ; t in s
$M_{propellant}$ vs. t	Propellant mass time evolution.	$M_{propellant}$ in kg; t in s
c* vs. t	Characteristic velocity time evolution.	c* in m/s; t in s
C_f vs. t	Thrust coefficient time evolution.	C_f adimensional; t in s
Perimeter vs. web	Perimeter versus burnt depth evolution.	Perimeter in ‘units of Table 6’ ; web in ‘units of Table 6’
P_{exit} vs. t	Nozzle exit pressure time evolution.	P_{exit} in Pa; t in s

Table 8. Output graphics given by the 2D grain burnback and 0D unsteady flow code.

Note that in case of Perimeter vs. t indicated in Table 8, the units in which is expressed the perimeter in the output corresponds with what it has been inputted as dimensional units of the geometry by the user (input ‘units” of Table 6). That is why for Perimeter in the column of units, it has been written ‘units of Table 6”. Analogously, the units of port area (A_p) also depend on this input. Thereby, for port area in the column of units, it has been indicated ‘units of Table 6^2”. Finally,

in case of Perimeter vs. web of Table 8, their units also depend on the dimensional values inputted by the user. So, this fact has been indicated in the section units of the Table 8 too.

6.3.2. OUTPUTS FOR QUASI-3D GRAIN BURNBACK AND 0D UNSTEADY FLOW SIMULATION

In order to obtain the outputs of this simulation tool, the main code called General_Quasi_3D_discrete_cart_coord_0D_flow.m (see section A.2.1. of the ANNEX A) should be run. Then, the outputs appeared in the computer screen, which can be classified as images or graphics too. One the one hand, in case of images, they can be static or dynamic. While the static images do not change during the plotting process, the dynamic images evolve at the same rate that does the grain burnback. Into this latter subgroup, the videos or the evolving 3D images are found. The Table 9 summarizes all of the output images of this tool. It should be pointed out that all of the videos obtained from this simulation tool are attached in the subfolder “Quasi_3D Grain Burnback and 0D Unsteady Flow” of the folder “Videos” found inside the CD of this project.

Name	Type	Description
3D grain plot	Static	In this image the initial 3D grain geometry, the final 3D grain geometry (at the last time step of the simulation) and the motor case profile are shown.
3D grain burnback video	Dynamic	In this video the 3D grain burnback evolution is shown. The axes are not established in real size, in order to ease the observation of the grain geometry during burnback.
3D grain evolution during rocket burning (real size axes)	Dynamic	In this video the 3D grain burnback evolution is shown. The axes are established in real size.

Table 9. Output images given by the Quasi-3D grain burnback and 0D unsteady flow code.

On the other hand, the graphics show the reference parameters to compare the performance of SRM when different grain configurations are used. This way, the output graphics given by this Quasi-3D grain burnback and 0D unsteady flow simulation tool are summarized in Table 10. Note that almost all of the different parameters plotted in the graphs are expressed according to SI. The only exceptions are the parameters related with the input “units” of Table 6 (e.g. the web), where the user can define the dimensional units of the geometry and the volume of the propellant grain ($V_{propellant}$), which is given in liters.

Name	Description	Units
A_b vs. t	Burning area time evolution.	A_b in m^2 ; t in s
m_{dotg} vs. t	Mass flow rate due to propellant combustion time evolution.	m_{dotg} in kg/s ; t in s
P_c vs. t	Combustion chamber pressure time evolution.	P_c in Pa ; t in s
m_{dotd} vs. t	Mass flow rate exiting the nozzle time evolution.	m_{dotd} in kg/s ; t in s
r_{dot} vs. t	Burning rate time evolution.	r_{dot} in mm/s ; t in s
$V_{propellant}$ vs. t	Propellant volume time evolution.	$V_{propellant}$ in L ; t in s
P_{cm} vs. t	Combustion chamber pressure (0D steady) time evolution.	P_{cm} in Pa; t in s
T_c vs. t	Combustion temperature time evolution.	T_c in K ; t in s
F vs. t	Thrust time evolution.	F in N ; t in s
V_{exit} vs. t	Nozzle exit velocity time evolution.	V_{exit} in m/s ; t in s
ρ_c vs. t	Combustion chamber gas density time evolution.	ρ_c in kg/m^3 ; t in s
$M_{propellant}$ vs. t	Propellant mass time evolution.	$M_{propellant}$ in kg ; t in s
c* vs. t	Characteristic velocity time evolution.	c* in m/s ; t in s
C_f vs. t	Thrust coefficient time evolution.	C_f dimensionless ; t in s
A_b vs. web	Burning area versus burnt depth evolution.	A_b in m^2 ; web in 'units of Table 6'
P_{exit} vs. t	Nozzle exit pressure time evolution.	P_{exit} in Pa ; t in s

Table 10. Output graphics given by the Quasi-3D grain burnback and 0D unsteady flow code.

6.3.3. OUTPUTS FOR QUASI-3D GRAIN BURNBACK AND 1D QUASI-STEADY FLOW SIMULATION

In order to obtain the outputs of this simulation tool, the main code called General_Quasi_3D_flow_1D_discrete_cart_coord.m (see section A.3.1. of the ANNEX A) should be run. In case of this simulation tool, the outputs given can be classified as images and graphs too. One the one hand, the outputted images are the same that the ones given by the Quasi-3D grain burnback and 0D unsteady flow simulation tool. Therefore, in order to consult this information,

please, see Table 9. On the other hand, the graphics given by this simulation tool are summarized in Table 11.

Name	Description	Units
A_b vs. t	Burning area time evolution.	A_b in m^2 ; t in s
m_{dotg} vs. t	Mass flow rate due to propellant combustion time evolution.	m_{dotg} in kg/s ; t in s
P_c vs. t	Combustion chamber pressure time evolution.	P_c in Pa ; t in s
m_{dotd} vs. t	Mass flow rate exiting the nozzle time evolution.	m_{dotd} in kg/s ; t in s
r_{dot} vs. t	Burning rate time evolution.	r_{dot} in mm/s ; t in s
$V_{propellant}$ vs. t	Propellant volume time evolution.	$V_{propellant}$ in L ; t in s
P_{cm} vs. t	Combustion chamber pressure (0D steady) time evolution.	P_{cm} in Pa; t in s
T_c vs. t	Combustion temperature time evolution.	T_c in K ; t in s
F vs. t	Thrust time evolution.	F in N ; t in s
V_{exit} vs. t	Nozzle exit velocity time evolution.	V_{exit} in m/s ; t in s
ρ_c vs. t	Combustion chamber gas density time evolution.	ρ_c in kg/m^3 ; t in s
$M_{propellant}$ vs. t	Propellant mass time evolution.	$M_{propellant}$ in kg ; t in s
c* vs. t	Characteristic velocity time evolution.	c* in m/s ; t in s
C_f vs. t	Thrust coefficient time evolution.	C_f dimensionless; t in s
A_b vs. web	Burning area versus burnt depth evolution.	A_b in m^2 ; web in ‘units of Table 6’
P_{exit} vs. t	Nozzle exit pressure time evolution.	P_{exit} in Pa ; t in s
u vs. z	Gas velocity evolution along the longitudinal axis of the SRM (z axis).	u m/s ; z in m
Mach vs. z	Number of Mach evolution along the longitudinal axis of SRM (z axis).	Mach dimensionless ; z in m
ρ vs. z	Gas density evolution along the longitudinal axis of the SRM (z axis).	ρ in kg/m^3 ; z in m
P vs. z	Static pressure evolution along the longitudinal axis of the SRM (z axis)	P in Pa ; z in m

Table 11. Output graphics given by the Quasi-3D grain burnback and 1D quasi-steady flow code.

Observing this Table 11, it should be pointed out that the first sixteen graphs are the same that for the Quasi-3D grain burnback and 0D unsteady flow simulation tool. This way, in case of these sixteen graphs, almost all of the different parameters plotted in the graphs are expressed according to SI. The only exceptions are the parameters related with the input "units" of Table 6 (e.g. the web), where the user can define the dimensional units of the geometry and the volume of the propellant grain ($V_{propellant}$), which is given in liters. Finally, note that the last four graphic differ considerably from the ones explained in Table 9 and in Table 10. These last four graphs indicate the variation of the parameters (velocity, Mach number, gas density and static pressure) along the longitudinal axis of the SRM (precisely, along the z axis). Note that the variation of these parameters is not shown until the exit nozzle; indeed, it is shown from the head of the SRM until the inlet of the nozzle.

6.4. FUNCTIONS

In this section the different functions developed to obtain the three SRM simulations (2D grain burnback and 0D unsteady flow, Quasi-3D grain burnback and 0D unsteady flow and Quasi-3D grain burnback and 1D quasi-steady flow) are presented. This way, in the following three subsections the specific functions used for each simulation are identified. For further information, it is recommended to consult ANNEX A, where the Matlab® codes of the functions as well as the explanation of each function are attached.

6.4.1. FUNCTIONS FOR 2D GRAIN BURNBACK AND 0D UNSTEADY FLOW SIMULATION

In the following Table 12 the names of the different functions used in the development of the 2D grain burnback and 0D unsteady flow simulation are identified. Note that these 18 files are written in Matlab® format. That is why they end with ".m".

Nº	Type	Name
1	Function	geometry_loading.m
2	Function	initial_geometry_organization.m
3	Function	curve_tangent_normal.m
4	Function	cartesian_grid.m
5	Function	minimum_distance_function.m
6	Function	interface_tracker_per.m
7	Function	port_area.m
8	Function	rocket_performance_0D_transient.m
9	Function	phi_one_time_step_evolution.m
10	Function	interface_tracker_in.m
11	Function	interface_tracker.m
12	Function	interface_tracker2.m
13	Function	mach_calculation.m
14	Function	order_vector_points.m
15	Function	perimeter.m
16	Function	units_conversion.m
17	Function	grain_video_evolution.m
18	Function	grain_video_evolution_filled.m

Table 12. List of names used to identify the functions of the simulation of 2D grain burnback and 0D unsteady flow.

A brief description of the functions of Table 12 is enclosed in the introduction of the section A.1. of the ANNEX A. The code of all those functions of the Table 12 is attached in the section A.1.2. of the same annex.

6.4.2. FUNCTIONS FOR QUASI-3D GRAIN BURNBACK AND 0D UNSTEADY FLOW SIMULATION

In the following Table 13 the names of the different functions used in the development of the Quasi-3D grain burnback and 0D unsteady flow simulation are identified. Note that these 25 files are written in Matlab® format. That is why they end with ".m".

Nº	Type	Name
1	Function	geometry_loading_3D.m
2	Function	chamber_volume.m
3	Function	curve_tangent_normal_3D.m
4	Function	minimum_distance_function_3D.m
5	Function	grain_cross_sections_pmt.m
6	Function	rocket_performance_0D_transient_quasi_3D.m
7	Function	grain_3D_plot_matrix.m
8	Function	motor_case_generation.m
9	Function	add_tri_matrix.m
10	Function	case_radius_int.m
11	Function	grain_cross_sections.m
12	Function	grain_video_evolution_3D.m
13	Function	initial_geometry_organization_3D.m
14	Function	interface_tracker_in_P.m
15	Function	interpolate_cross_section.m
16	Function	tri_generation.m
17	Function	wetted_surface.m
18	Function	wetted_surface2.m
19*	Function	units_conversion.m
20*	Function	mach_calculation.m
21*	Function	phi_one_time_step_evolution.m
22*	Function	cartesian_grid.m
23*	Function	curve_tangent_normal.m
24*	Function	order_vector_points.m
25*	Function	port_area.m

Table 13. List of names used to identify the functions of the simulation of Quasi-3D grain burnback and 0D unsteady flow.

Some of the functions of the Table 13 are repeated in the Table 12; precisely, all of the functions that have a star ("*") in the number are repeated (the set functions located just at the end of the Table 13). So, in order to see the implications of repeated functions, please consult the previous sub-section.

Additionally, a brief explanation of not repeated functions of the Table 13 (not of the main code) is attached in the introduction of the section A.2. of the ANNEX A. Furthermore, the code of all of the not repeated functions of the Table 13 is enclosed in the section of A.2.2. of the same annex.

6.4.3. FUNCTIONS FOR QUASI-3D GRAIN BURNBACK AND 1D QUASI-STEADY FLOW SIMULATION

In the following Table 14 the names of the different functions used in the implementation of the Quasi-3D grain burnback and 1D quasi-steady flow simulation are identified. Note that these 28 files are written in Matlab® format. That is why they end with ".m".

Nº	Type	Name
1	Function	dif_A_p.m
2	Function	grain_cross_sections_pmt_1D.m
3	Function	interpolate_bt_cs.m
4	Function	nozzle_inlet_mach.m
5	Function	p_u_rho_calc.m
6	Function	perimeter_cross_section.m
7	Function	rocket_performance_1D_steady_Q3D.m
8*	Function	add_tri_matrix.m
9*	Function	cartesian_grid.m
10*	Function	case_radius_int.m
11*	Function	chamber_volume.m
12*	Function	curve_tangent_normal.m
13*	Function	curve_tangent_normal_3D.m
14*	Function	geometry_loading_3D.m
15*	Function	grain_3D_plot_matrix.m
16*	Function	grain_video_evolution_3D.m
17*	Function	initial_geometry_organization_3D.m
18*	Function	interface_tracker_in_P.m
19*	Function	interpolate_cross_section.m
20*	Function	mach_calculation.m
21*	Function	minimum_distance_function_3D.m

22*	Function	motor_case_generation.m
23*	Function	order_vector_points.m
24*	Function	phi_one_time_step_evolution.m
25*	Function	port_area.m
26*	Function	tri_generation.m
27*	Function	units_conversion.m
28*	Function	rocket_performance_0D_transient_quasi_3D.m

Table 14. List of names used to identify the functions of the simulation of Quasi-3D grain burnback and 1D quasi-steady flow.

Some of the functions of the Table 14 are repeated in the Table 12 or in the Table 13; precisely, all of the functions that have a star ("*") in the number are repeated (the twenty one functions located just at the end of the Table 14). So that, in order to understand the implications of these repeated functions, please consult the previous two sub-sections. In addition, a brief description of the functions of the not repeated functions (not of the main code) is attached in the introduction of the section A.3. of the ANNEX A. Furthermore, the code of all of the not repeated functions of the Table 14 is enclosed in the section A.3.2. of the same annex.

7. RESULTS

This section presents the main results obtained with the simulation tools developed in this study using Matlab®. Firstly, the 2D grain burnback and 0D unsteady flow code is verified by comparing the simulation results of a circular grain against its analytical solution. Next, the 3D grain burnback and 0D unsteady flow code is tested and compared against the experimental and numerical results for the NAWC motor No. 6 and No. 13. At the end of the section 7.1.2.2, an example of the results obtained with the Quasi-3D grain burnback and 1D quasi-steady flow model tool is shown. Finally, some relevant 2D grain designs studied in this project are presented, though only one is exposed in this document while the rest of the cases are explained in the ANNEX B.

7.1. CODE VERIFICATION

In order to verify the simulation tools developed in this study, it is necessary to compare the results obtained with the code against some reference cases. The data from the reference cases shall be either analytical, experimental, or obtained with a reliable simulation tool. This way, the correctness of the code is given by the matching level of the simulated results with the reference data.

7.1.1. 2D CODE VERIFICATION: CYLINDRICAL GRAIN

The 2D grain burnback and 0D unsteady flow code is verified by comparing the simulation results of a circular grain against its analytical solution. Indeed, the circular grain, for its simplicity and availability of an analytical solution is a good reference to compare with. At any time, the interface perimeter of a circular grain is given by $P_b(t) = 2\pi \cdot r(t)$, where $r(t)$ is the radius of the grain at time t. As well, the propellant burning rate is equal to the time rate of change of the interface radius i.e. $\dot{r_b} = \dot{r}(t)$. Therefore, integrating this expression, the interface radius analytical solution is found as $r(t) = r_0 + \dot{r_b} \cdot t$, where r_0 is the grain initial radius. So, the burning area can be calculated analyticaly for any time as follows: $A_b(t) = \left(2\pi(r_0 + \dot{r_b} \cdot t)\right)\left(L_{grain} - 2 \cdot \dot{r_b} \cdot t\right) + 2\pi\left({R_{case}}^2 - (r_0 + \dot{r_b} \cdot t)^2\right)$.
It has been considered that both grain ends burn. As well, L_{grain} is the grain length and R_{case} is the motor case radius. In Figure 47 the grain initial geometry is shown.

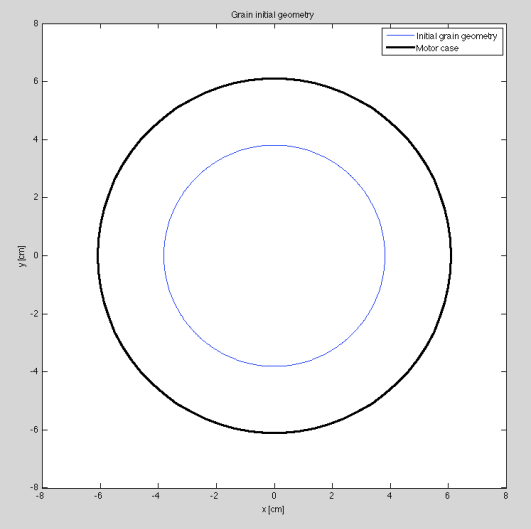

Figure 47. Circular grain initial geometry.

The inputs of the code for this case are shown in Figure 48.

```
%--------------------------------DATA INPUT-------------------------------%
%%
%INPUT PARAMETERS
N=250;                  %Number of nodes per side in the cartesian grid.
N_Delta_web=60;         %Number of contour lines shown in the Phi plot.
N_bs_plot=16;           %Number of burnt steps shown in grain evolution plot.
Time=5.45;               %Simulation time [s]
n_steps=700;              %Number of simulation time steps.
t_0=0.5;                %Time duration of the first phase of the simulation (heavy transient).
Delta_t0=0.003;         %Time step during t_0 phase of the simulation.
Delta_t1=(Time-floor(t_0/Delta_t0)*Delta_t0)/(n_steps-floor(t_0/Delta_t0));    %Time step for the rest of simulation.
%Distance units selection:
units='cm';
%Units conversion:
[K_units] = units_conversion(units);
atm_units=101325;       %Conversion factor from Pa to atm.

%Rocket input parameters:
L_g=0.697992;               %Propellant grain length [m]
P_c0=1.1E6;               %Initial chamber pressure [Pa]
a=3.80617e-6;                %a parameter of the propellant burning rate law: r=a*Pc^n
n_exp=0.461;             %n exponent of the propellant burning rate law: r=a*Pc^n
gamma_e=1.19577;        %Combustion products gas gamma parameter gamma=Cp/Cv
R_air=287;              %R of air [J/KgK]
R_e=322.21;             %R of the combustion products [J/KgK]
rho_p=1800;          %Propellant density [Kg/m^3]
M_mol=0.03875;          %Combustion products molar mass [kg/mol]
De=85;                 %Nozzle exit diameter [mm]
Dt=26.416;                %Nozzle throat diameter [mm]
A_e=(pi*(De*1e-3)^2)/4; %Nozzle exit area [m^2]
A_t=(pi*(Dt*1e-3)^2)/4; %Nozzle trhoat area [m^2]
L_plenum=0.1524;           %Plenum length [m]
[M_e] = mach_calculation(A_e,A_t,gamma_e);     %Nozzle exit Mach number
P_a=1e4;                %Atmospheric pressure [Pa]

T_c_v=[2462.02544,2470.45874,2474.34792,2476.69366,2478.30590,2479.50156];      %Vector of chamber temperature data re
P_c_v=1:1:6;            %Vector of chamber pressure data [MPa].
p_T=polyfit(P_c_v,T_c_v,3);         %Polynomial coeficients of the function adjust of T_c=f(P_c)
```

Figure 48. Inputs of the simulation tool for the circular grain.

In Figure 49, the grain evolution is shown by representing the interface shape at equally spaced time steps. These results are obtained through the 2D grain burnback and 0D unsteady flow code. Note that, despite of the fact that the background grid is Cartesian and with a moderate number of nodes, the round shape of the circular grain is perfectly modeled.

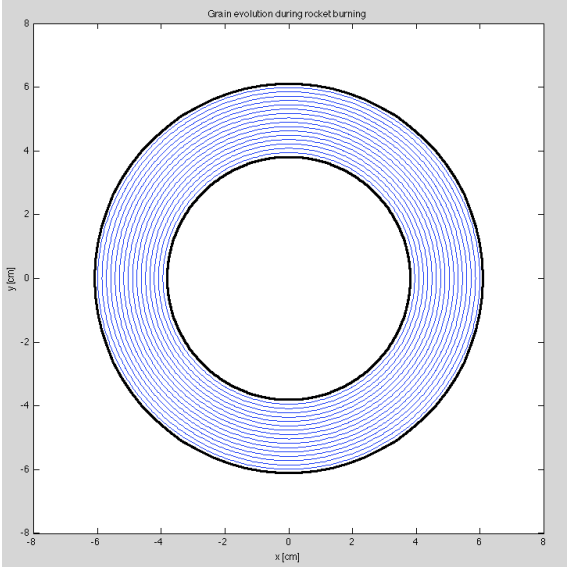

Figure 49. Circular grain evolution during the rocket operation.

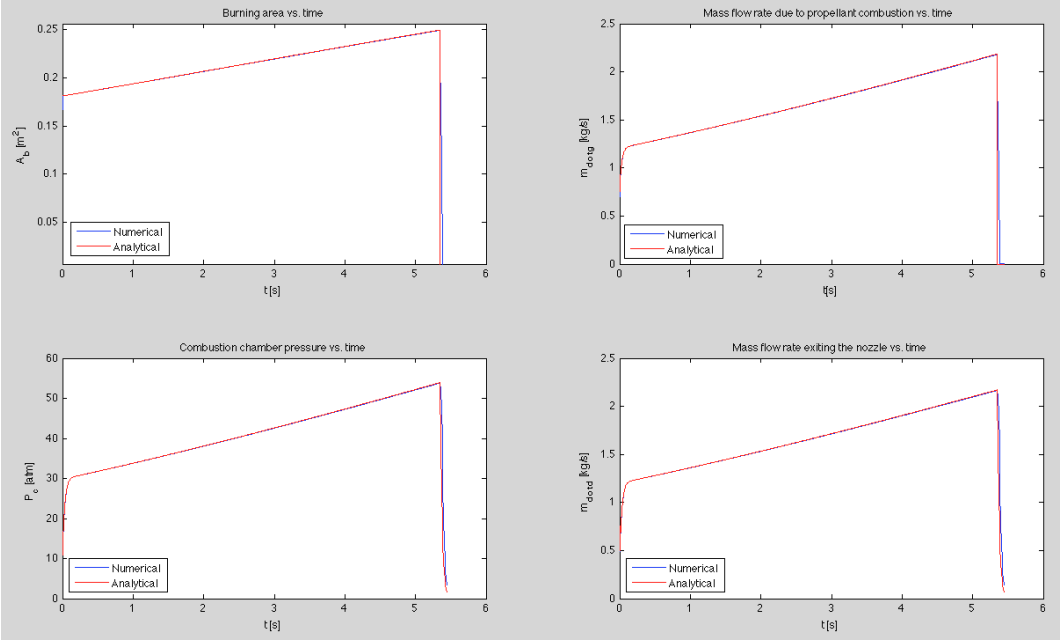

Figure 50. Comparison of the numerical vs. analytical results of burning area, mass flow rate due to propellant combustion, chamber pressure and mass flow rate exiting the nozzle time evolutions.

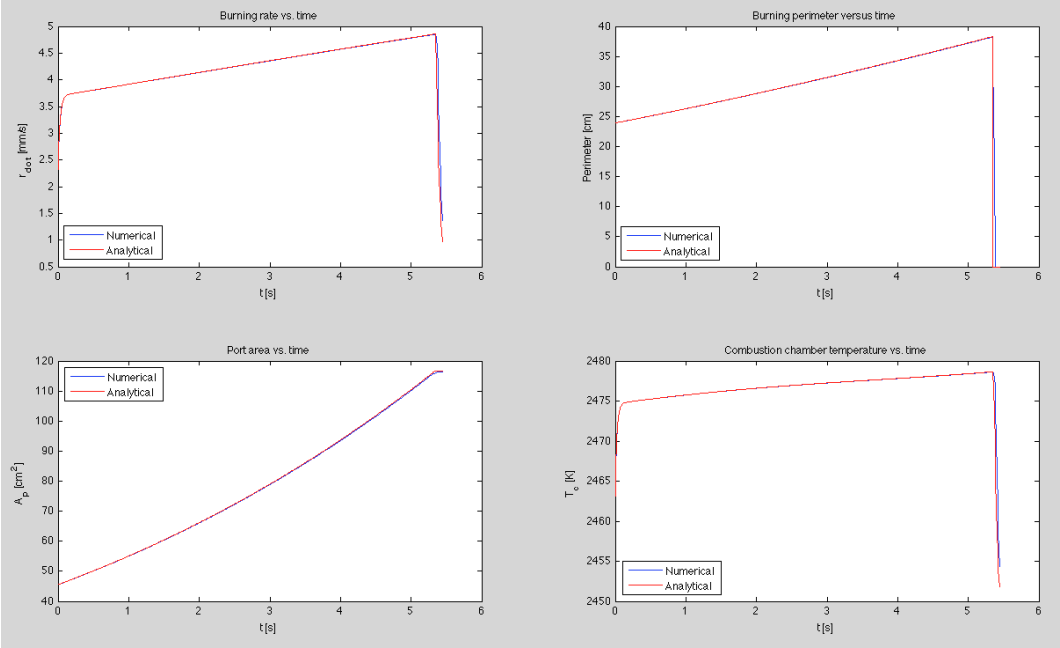

Figure 51. Comparison of the numerical vs. analytical results of burning rate, burning perimeter, port area and chamber temperature time evolutions.

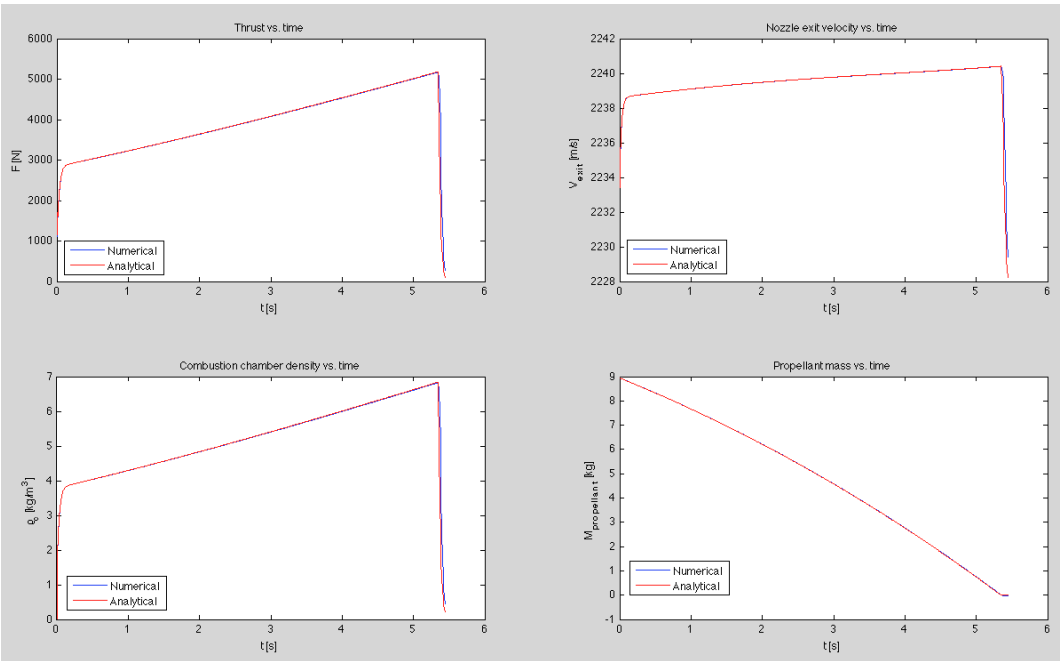

Figure 52. Comparison of the numerical vs. analytical results of thrust, nozzle exit velocity, chamber gas density and propellant mass time evolutions.

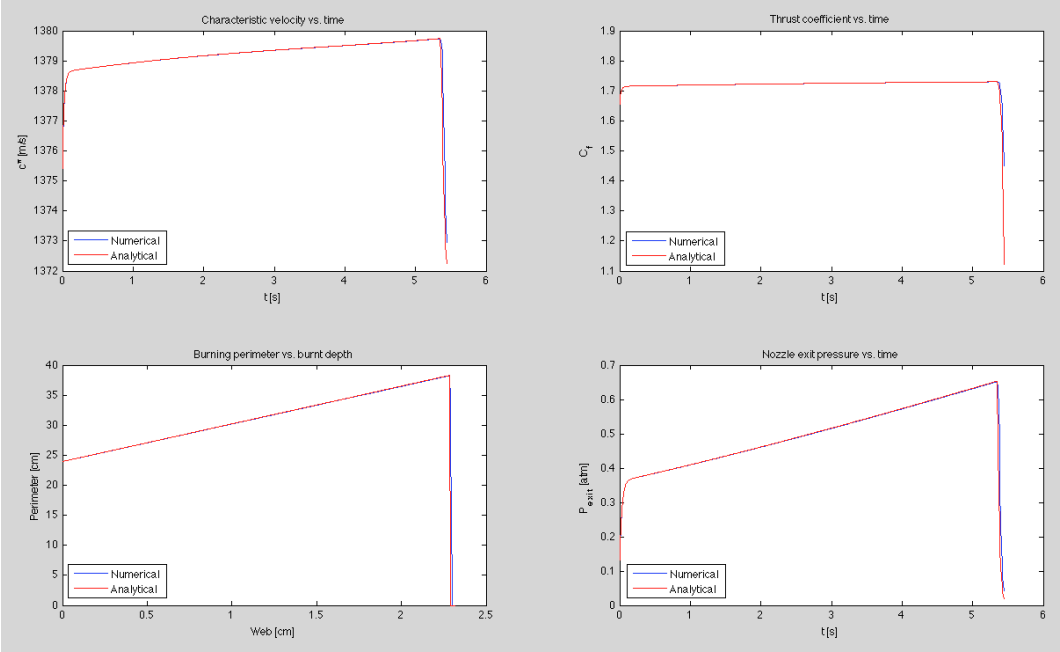

Figure 53. Comparison of the numerical vs. analytical results of characteristic velocity, thrust coefficient and nozzle exit pressure time evolutions and burning perimeter vs. burnt depth evolution.

As it is shown in Figure 50, Figure 51, Figure 52 and Figure 53 the performance results obtained through the 2D grain burnback and 0D unsteady flow code match very well with the circular analytical solution. Indeed the simulation solution falls on the top of the analytical solution.

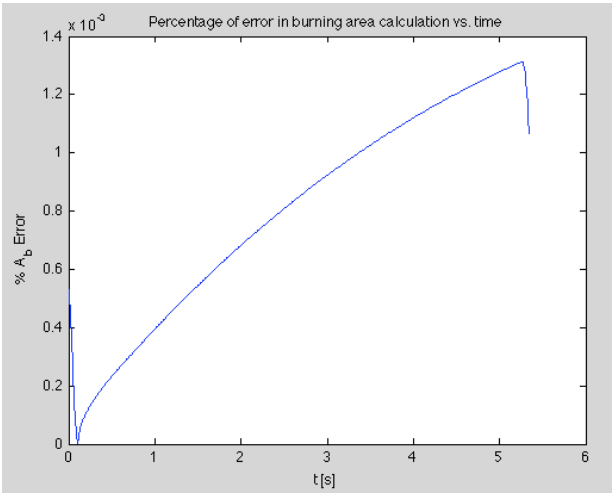

Figure 54. Percentage of error in burning area time evolution.

To sum up, Figure 54 shows the percentage of error in the burning surface area calculation with respect to the time. Note that in the y-axis of this figure, the percentage is multiplied by 10^{-3}, so that the absolute error is of the order of 10^{-5}, which is a very small value. At the same time, it should be pointed out that the

error increases with the time. This is a logical consequence typical from any numerical method, because as the solution is approximated, the error gets bigger with the time. However, this trend could be reduced by re-initializing the ϕ function after some time. This re-initialization aims to take back the ϕ function to a signed distance function. Recall that the ϕ function is initialized with a signed minimum distance function, but as it is propagated with the time it separates from the signed distance function. However, in the code developed in this study, the ϕ function is not re-initialized. Nevertheless, the error is so small for this simulation time that the accuracy obtained by the developed code is good enough.

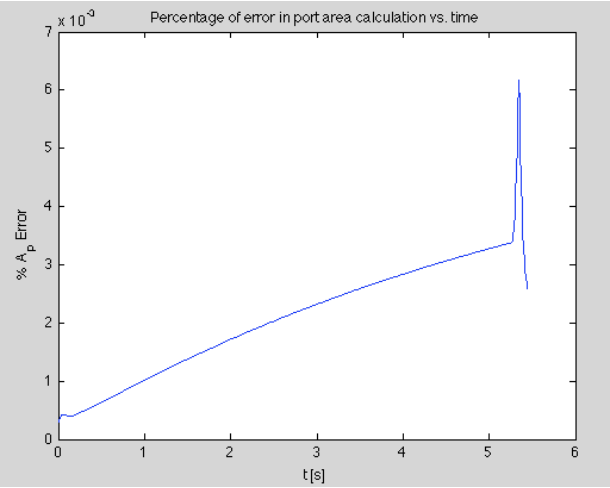

Figure 55. Percentage of error in port area time evolution.

On the other side, the port area calculation experiences the same trend that the burning area. Note that the analytical port area is $A_p(t) = \pi \cdot r(t)^2$, so that the error grows faster than the burning surface area term, because $r(t)$ is squared. As a concluding remark, it can be stated that the 2D grain burnback and 0D unsteady flow code is precise and provides accurate results for 2D grains. Hence, the objective of calculating 2D grain designs has been successfully accomplished.

7.1.2. 3D CODE VERIFICATION

In order to verify the code for 3D grain geometries, the Naval Air Warfare Center (NAWC) rocket motors have been used as reference. There is public available experimental data of motor firings done by the NAWC [12]. This way, the motor No.13 and No.6 have been selected. The grain geometry of these two rocket motors and its ballistic performance have been widely studied by several authors. For instance, in [14] the authors analyzed numerically the temporal evolution of burning surface area, combustion chamber volume and solid propellant volume of the NAWC motor No.6. Furthermore, in [15] the authors studied numerically the

temporal evolution of the chamber pressure of both motors No.13 and No.6. Additionally, due to being widely studied motors, its exact grain geometry and propellant characteristics can be found in [12].

As it has been aforementioned, the procedure followed to obtain these motors 3D performance results and verify the code of Quasi-3D grain burnback consists of:

- First, generate the CAD design of both motors grain geometries and motor case profiles following the dimensions given in [12].
- Decide which cross-sections of the grain provide meaningful information. Save these cross-sections and the motor case profile in IGES format.
- Realize the discretization using the GID software provided by [17]. For that, it is necessary to open the IGES files, do the discretization and then, save the results as text files.
- Upload the text files in the Quasi-3D grain burnback and 0D unsteady flow simulation tool. Introduce the inputs required by the simulation tool (for further information see 6.1.2).
- Run the simulation tool, obtain the different outputs and compare the results with the ones given in [14] and in [15].

In the following sub-section the code verification for the cases of NAWC No.13 and NAWC No.6 is presented separately. Note that in order to verify the code, the Quasi-3D grain burnback and 0D unsteady flow simulation tool has been used. In order to show the results of the 1D quasi-steady simulation tool, a final graph for the NAWC No.6 is presented.

7.1.2.1. NAWC motor No.13

As it is shown in Figure 56, the grain geometry of the NAWC motor No. 13 is an internal burning tube that ends in cone shape. Next, the plots of the interface shape at 6 time steps of the grain burnback simulation are presented.

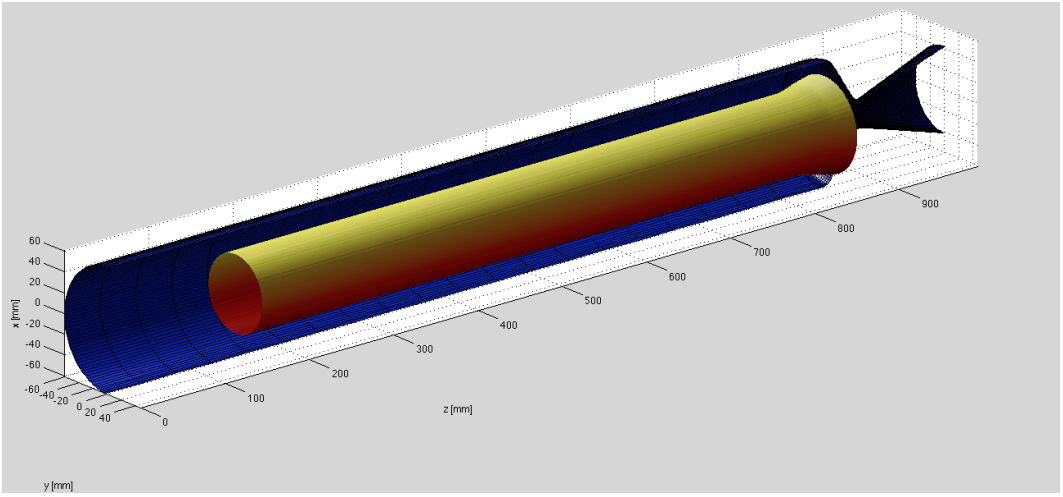

Figure 56. NAWC No.13 grain burning surface (interface) at t=0s.

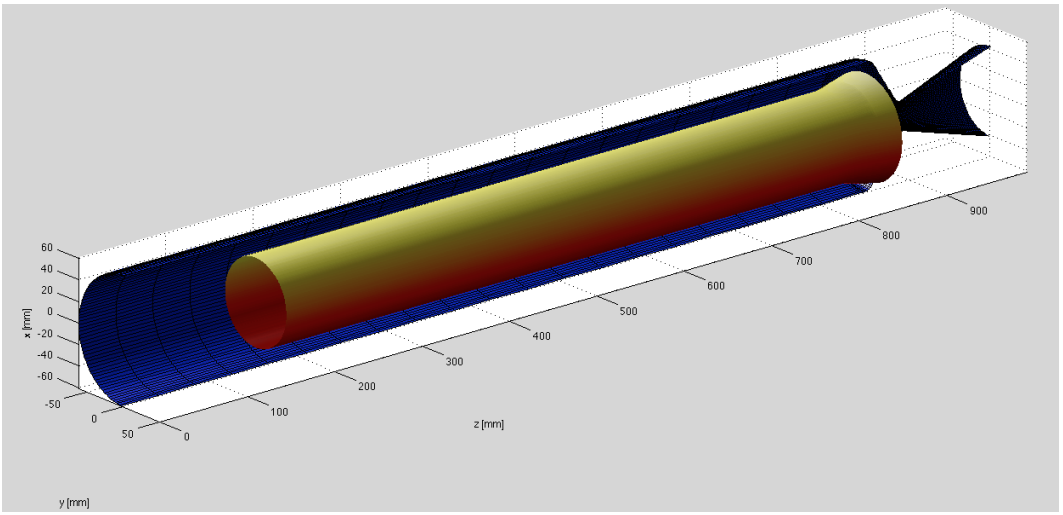

Figure 57. NAWC No.13 grain burning surface (interface) at t=1s.

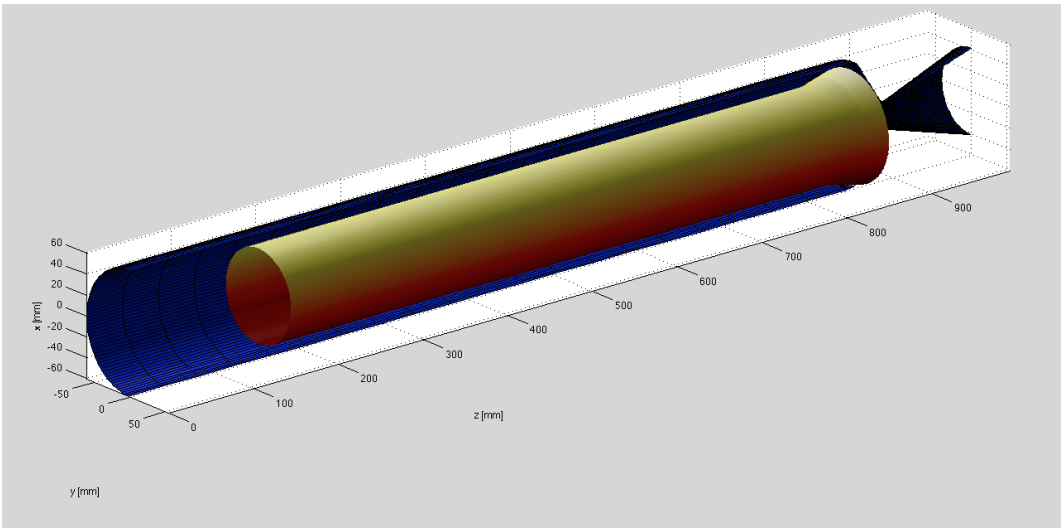

Figure 58. NAWC No.13 grain burning surface (interface) at t=2s.

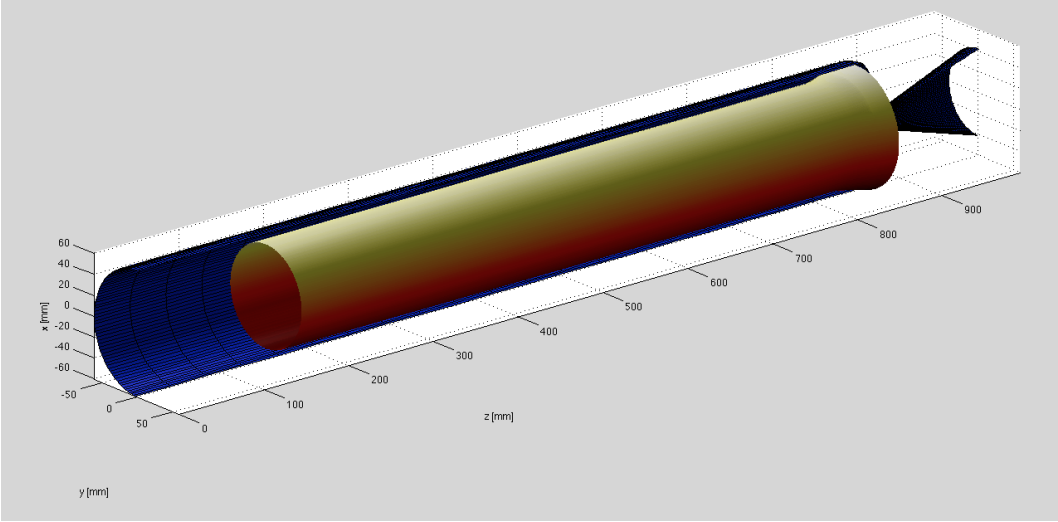

Figure 59. NAWC No.13 grain burning surface (interface) at t=3s.

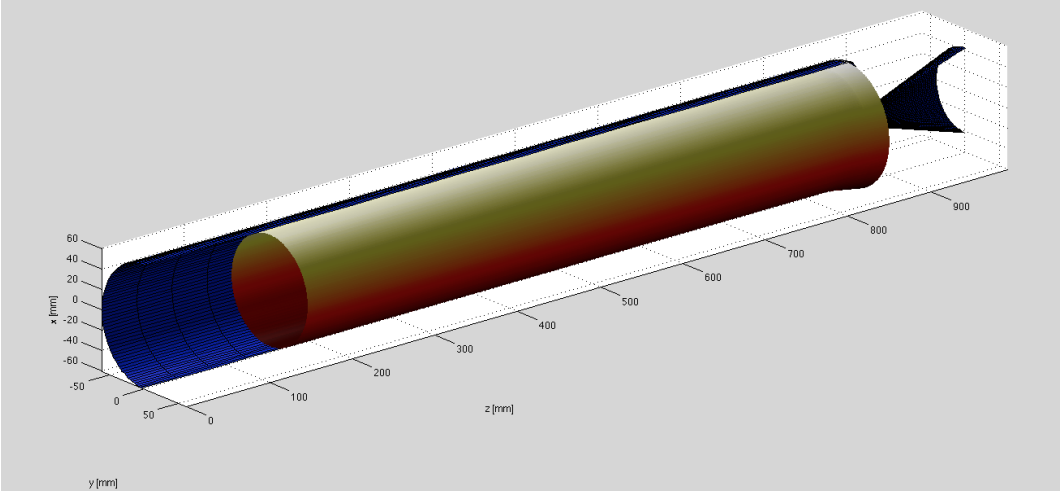

Figure 60. NAWC No.13 grain burning surface (interface) at t=4s.

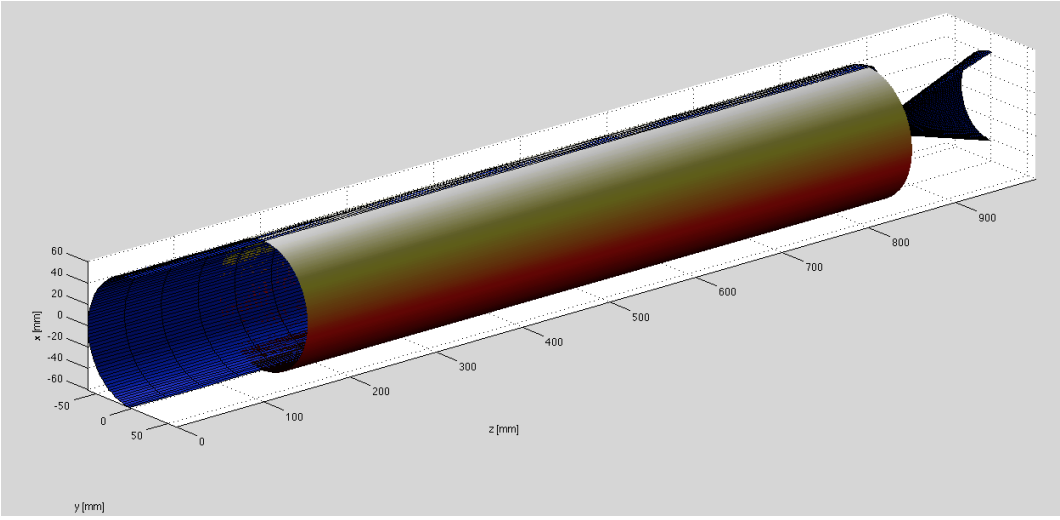

Figure 61. NAWC No.13 grain burning surface (interface) at t=5.3s (burnout).

Through these 6 figures, the sequence of the grain evolution has been shown. Recall that this grain geometry is modeled using only three reference cross-sections, as it is shown in Figure 37. However, an additional interpolated cross-section has been added for each couple of reference cross-sections. So in this case, there are 2 interpolated cross-sections between the three reference cross-sections.

On the other side, Figure 62, Figure 63, Figure 64 and Figure 65 show the performance results obtained with the Quasi-3D grain burnback and 0D unsteady flow simulation tool. These figures show the typical behavior of an internal burning tube grain. For instance, these results are very similar to those obtained for the circular grain validation in the 2D case. This result is logical because practically the grain is an internal tube with a slight cone shape at one of the ends.

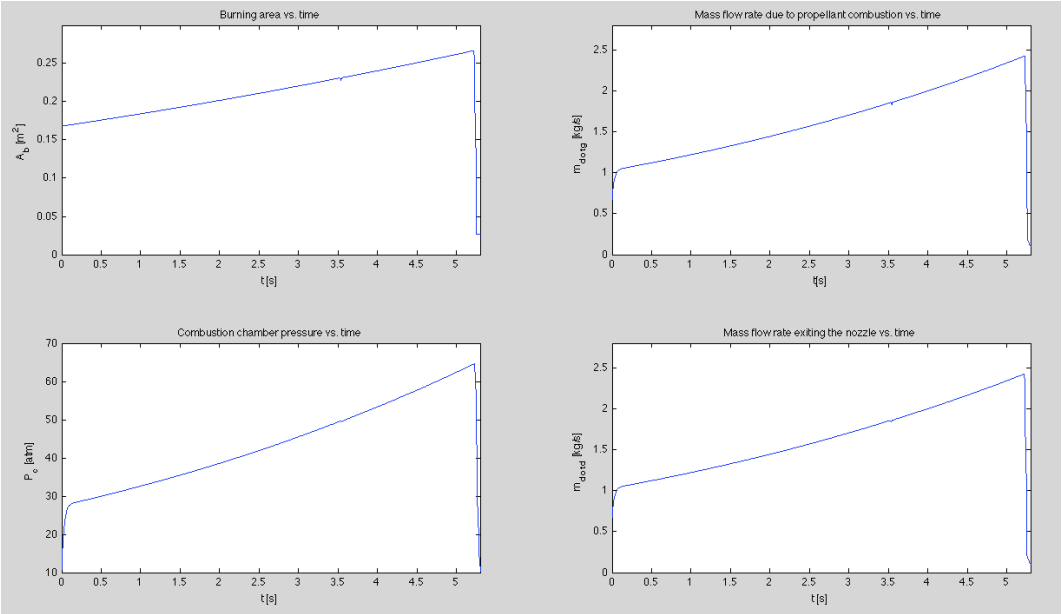

Figure 62. Burning area, mass flow rate due to propellant combustion, chamber pressure and mass flow rate exiting the nozzle time evolutions for the NAWC motor No.13.

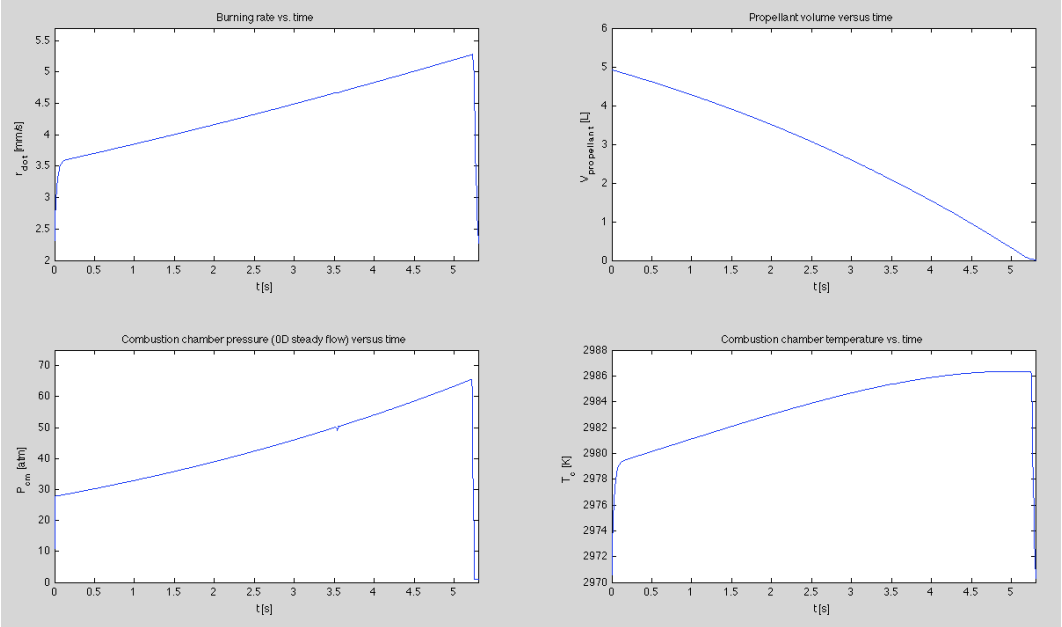

Figure 63. Burning rate, propellant volume, chamber pressure (0D steady flow) and chamber temperature time evolutions for the NAWC motor No.13.

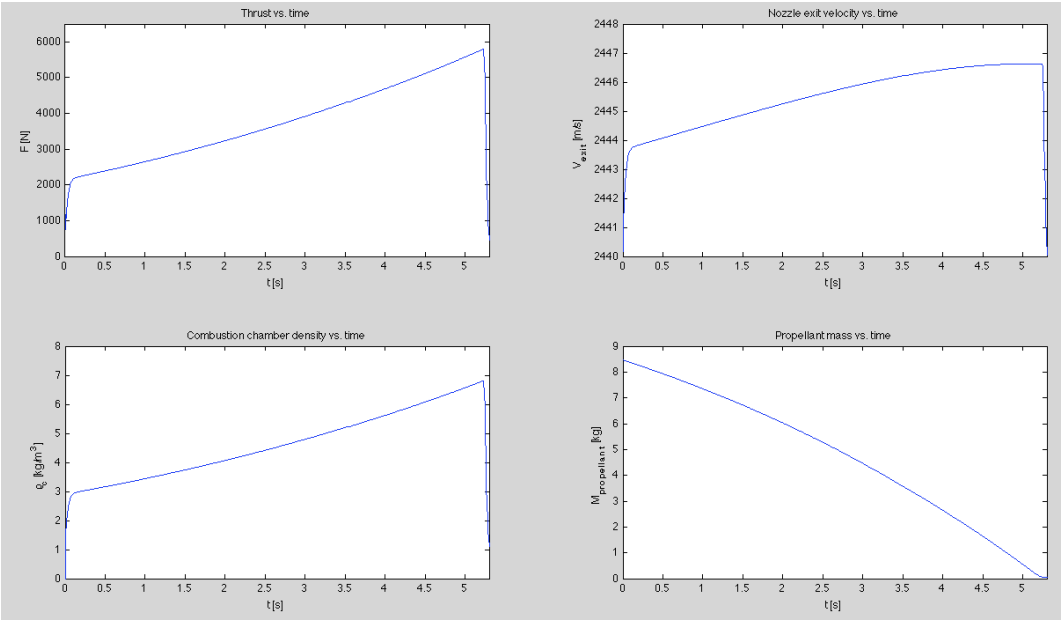

Figure 64. Thrust, nozzle exit velocity, chamber gas density and propellant mass time evolutions for the NAWC motor No.13.

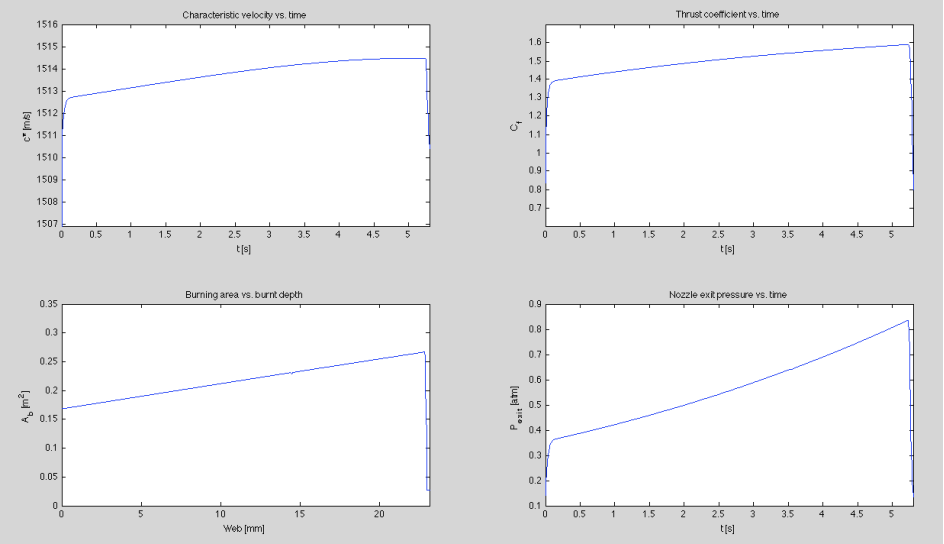

Figure 65. Characteristic velocity, thrust coefficient and nozzle exit pressure time evolution and burning area vs. burnt depth for the NAWC motor No.13.

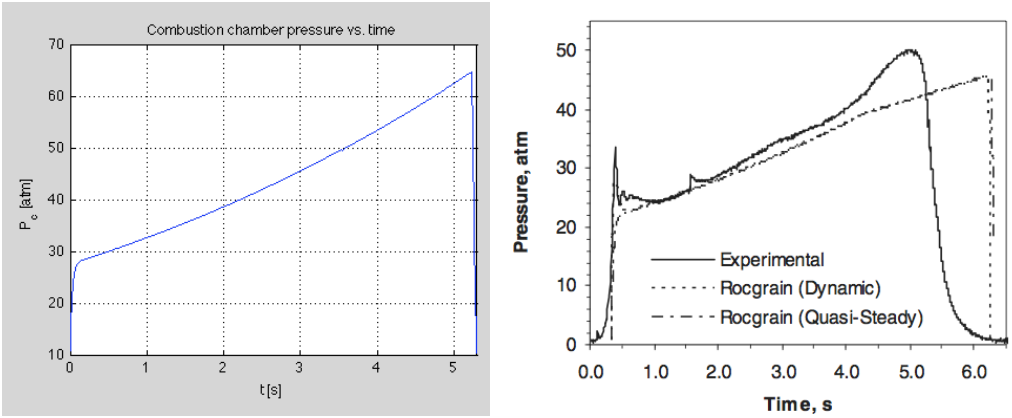

Figure 66. Comparison between chamber pressure time evolution obtained from the simulation tool (L) and the result obtained by Willcox, Brewster, Tang, Stewart and Kuznetsov in source [15] (R) for the NAWC motor No.13.

Finally, in order to verify the results obtained with the Quasi-3D grain burnback and 0D unsteady flow simulation tool, a comparison is done using as reference the experimental results from [12]. Note that the Figure 66 (R) has been extracted from [15], but the experimental results of this motor come from [12]. As it can be stated from the comparison of these two figures, the simulation results match the trend but have a little offset error. This offset error is mainly given by a difference in the propellant input data, because the data used was not exactly the propellant actual properties. So that, the main cause of the error is caused by this difference. This conclusion is inferred because the results obtained with the NAWC motor No. 6, which has a more complicated geometry, match very well and the propellant properties are the actual ones. So, in case of the NAWC motor

No. 13, the difference between the simulation results and the reference ones is mainly caused by the propellant data and not by the Quasi-3D model.

7.1.2.2. NAWC motor No.6

The other studied case, the NAWC motor No.6 has a 3D complex grain geometry. This type of geometry is called star aft grain. Figure 67, Figure 68 and Figure 69 show the geometry of this grain from different points of view. This case is interesting, because it has enough complexity to show the capabilities of the Quasi-3D grain burnback model with 0D unsteady flow.

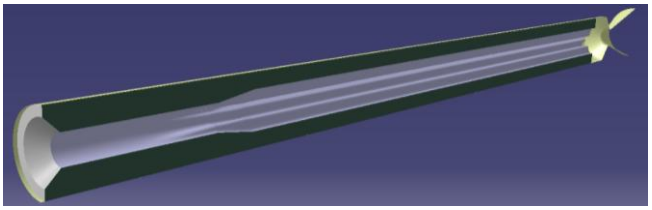

Figure 67. Sectioned front view of the CAD design of the NAWC initial grain geometry.

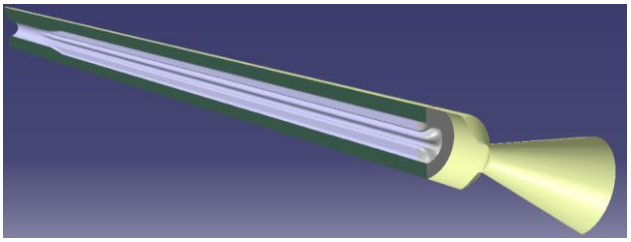

Figure 68. Sectioned rear view of the CAD design of the NAWC initial grain geometry.

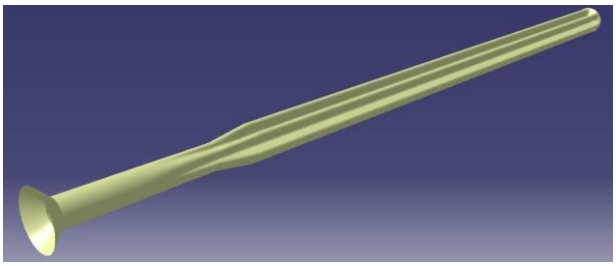

Figure 69. CAD design of the NAWC No.6 inner grain surface.

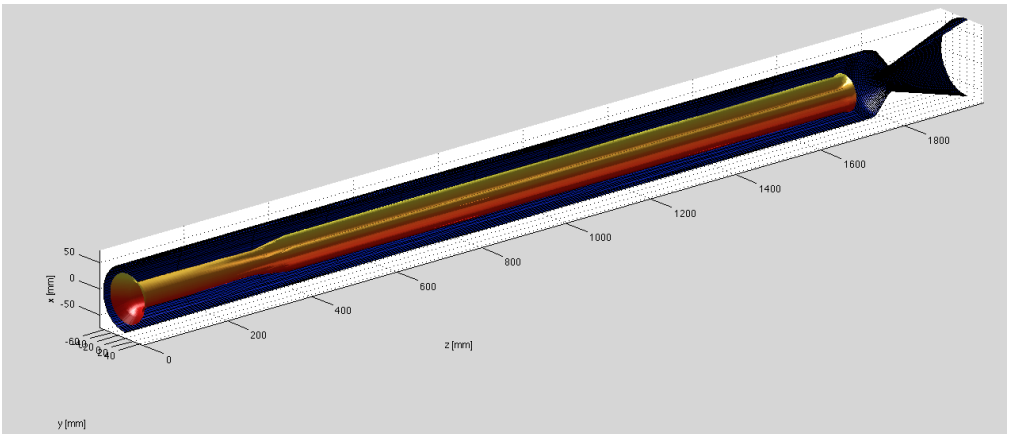

Figure 70. Real size view of NAWC No.6 grain burning surface (interface) at t=0s.

Next, the plots of the interface shape at 6 time steps of the grain burnback simulation are presented.

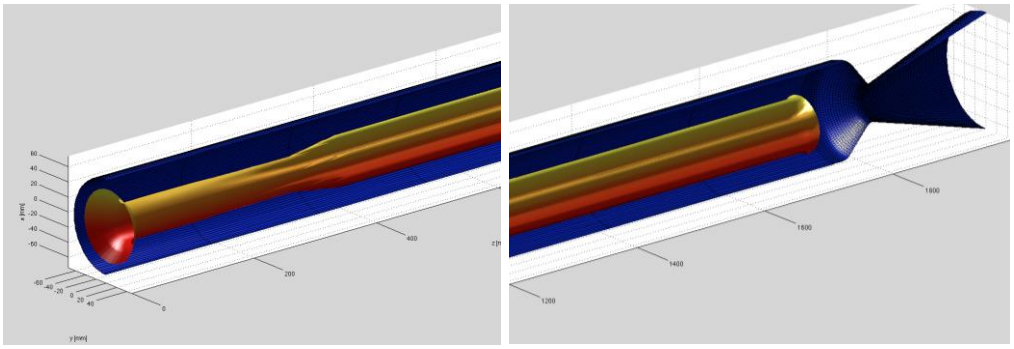

Figure 71. Detail views of NAWC No.6 grain burning surface (interface) at t=0s.

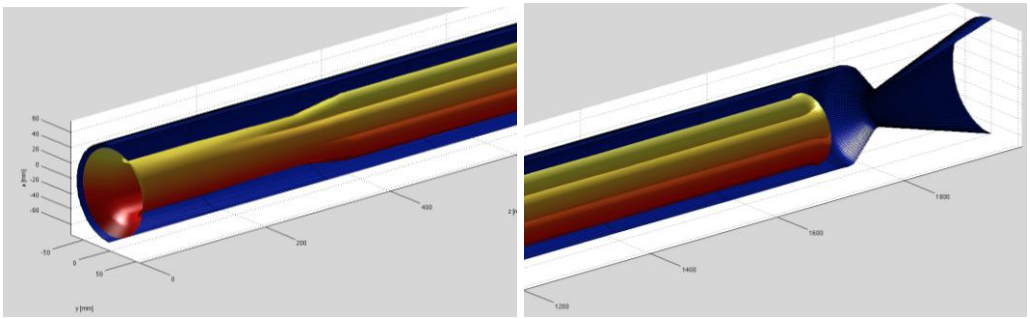

Figure 72. Detail views of NAWC No.6 grain burning surface (interface) at t=1.5s.

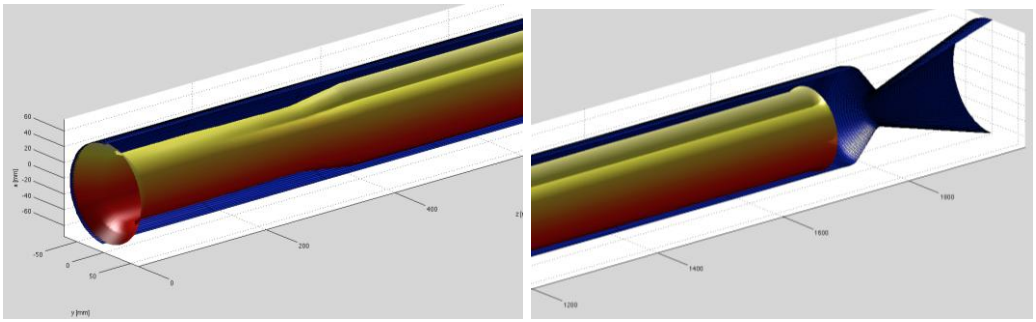

Figure 73. Detail views of NAWC No.6 grain burning surface (interface) at t=3s.

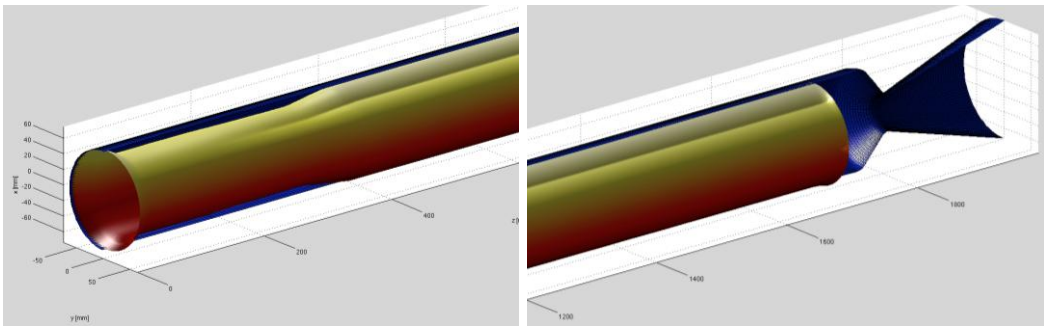

Figure 74. Detail views of NAWC No.6 grain burning surface (interface) at t=4.5s.

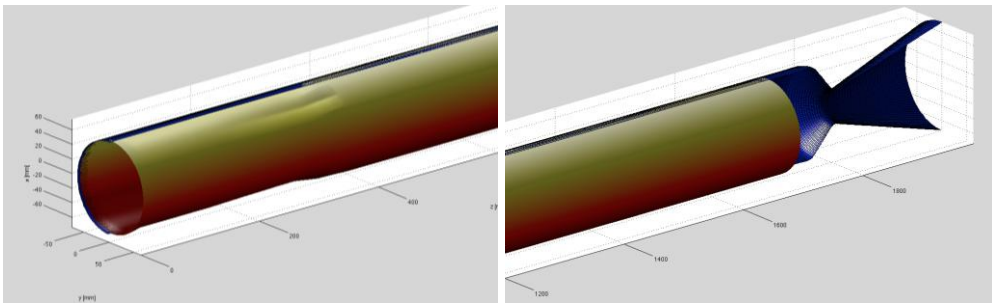

Figure 75. Detail views of NAWC No.6 grain burning surface (interface) at t=6s.

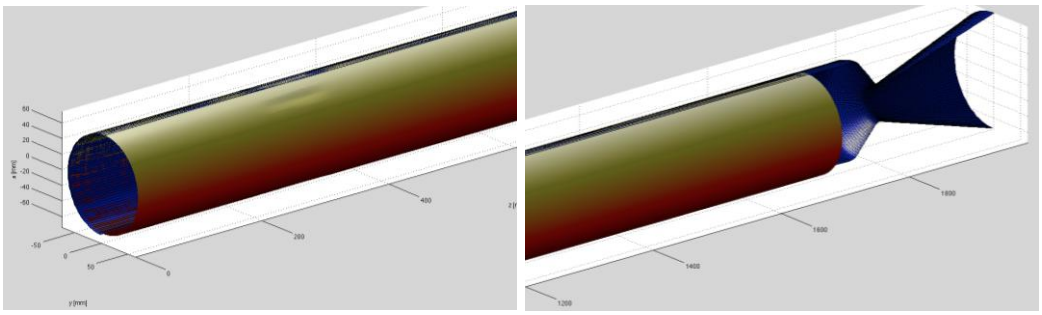

Figure 76. Detail views of NAWC No.6 grain burning surface (interface) at t=7.5s (burnout).

Through these 6 figures, the sequence of the grain evolution has been shown. Recall that this grain geometry is modeled using 17 reference cross-sections, as it is shown in Figure 38. As well, an additional interpolated cross-section has

been added for each couple of reference cross-sections. So in this case, there are 16 interpolated cross-sections.

On the other side, Figure 77, Figure 78, Figure 79 and Figure 80 show the performance results obtained with the Quasi-3D grain burnback and 0D unsteady flow simulation tool for the NAWC motor No. 6.

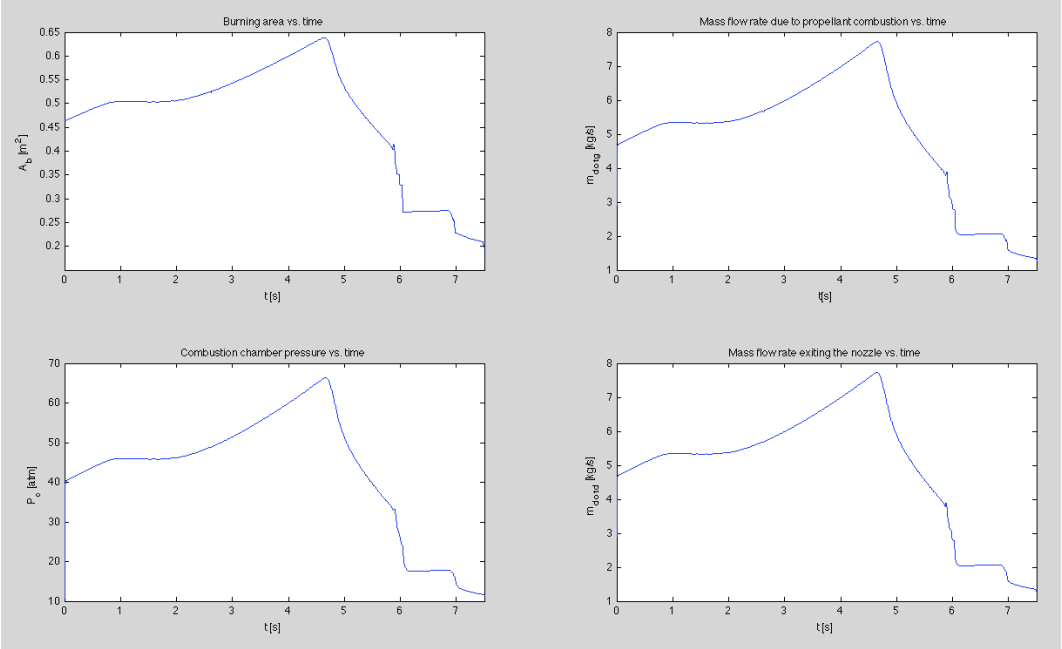

Figure 77. Burning area, mass flow rate due to propellant combustion, chamber pressure and mass flow rate exiting the nozzle time evolutions for the NAWC motor No.6.

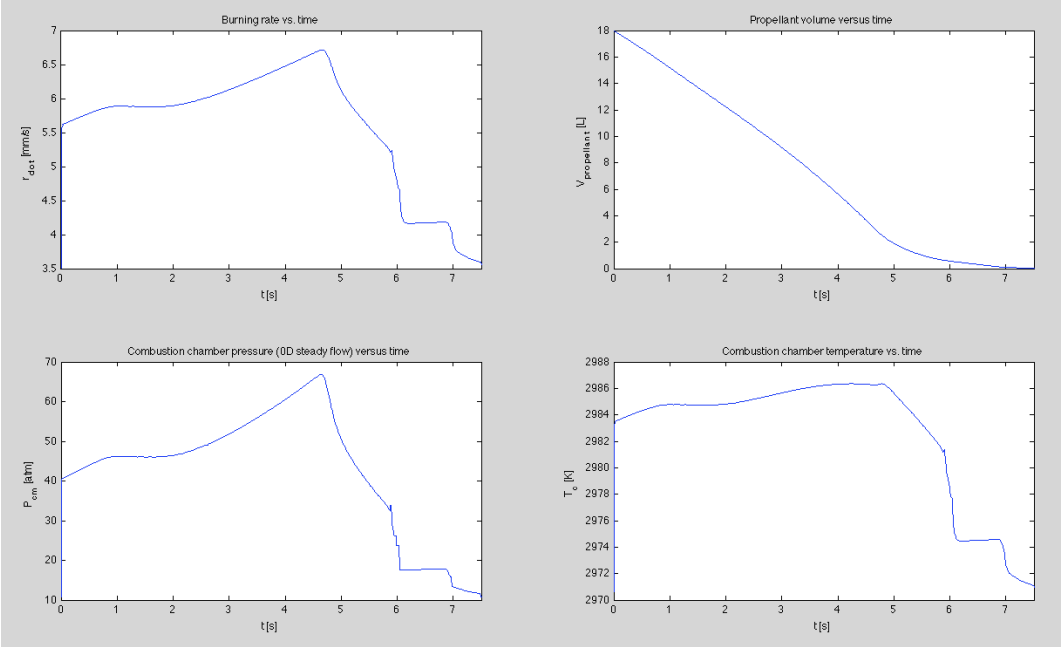

Figure 78. Burning rate, propellant volume, chamber pressure (0D steady flow) and chamber temperature time evolutions for the NAWC motor No.6.

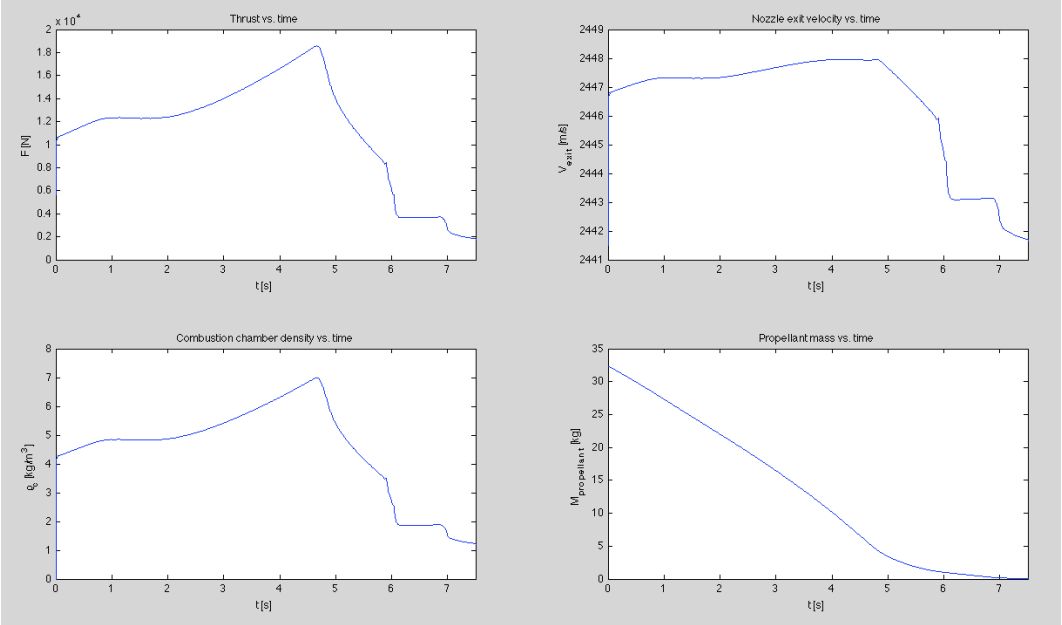

Figure 79. Thrust, nozzle exit velocity, chamber gas density and propellant mass time evolutions for the NAWC motor No.6.

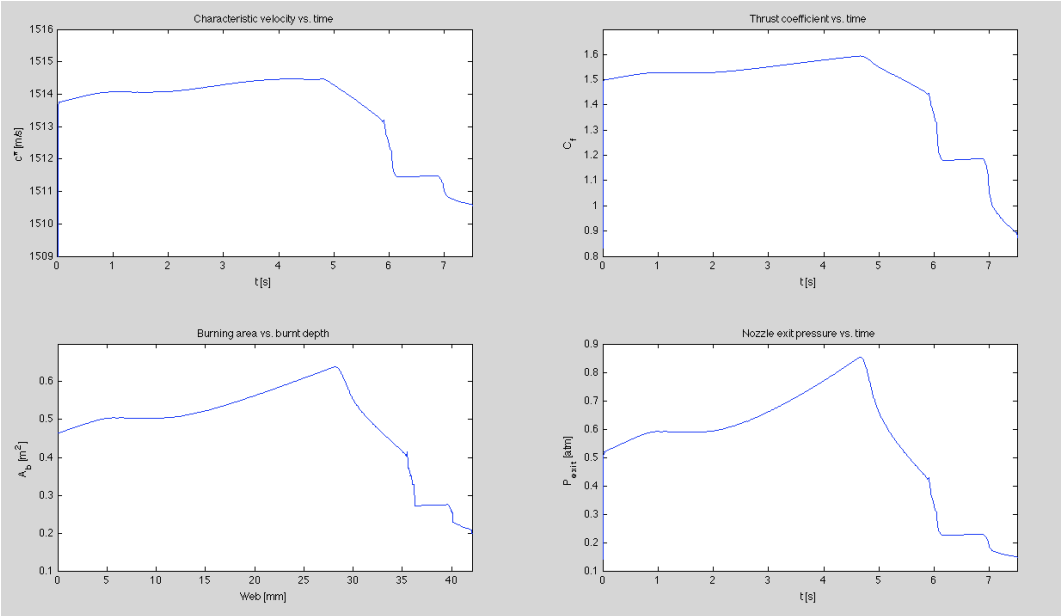

Figure 80. Characteristic velocity, thrust coefficient and nozzle exit pressure time evolution and burning area vs. burnt depth for the NAWC motor No.6.

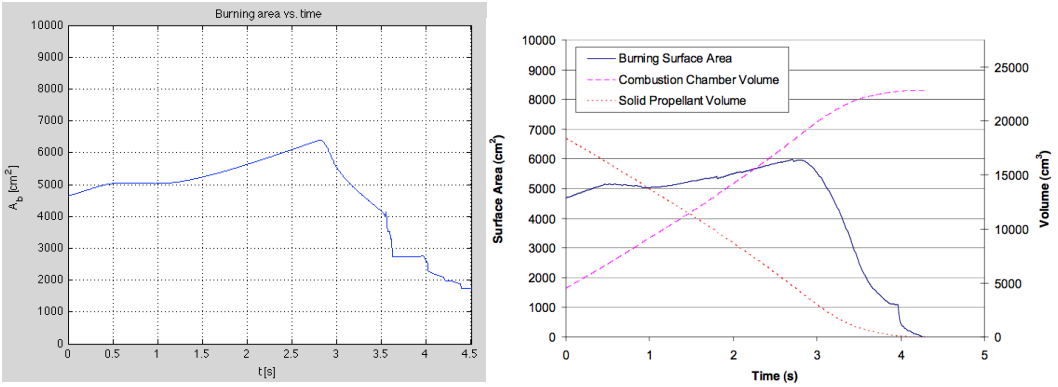

Figure 81. Comparison between the burning area time evolution obtained from the simulation tool (L) and the result obtained by Willcox, Brewster, Tang and Stewart in source [14] (R) for the NAWC motor No.6.

The most important conclusions about the Quasi-3D grain burnback and 0D unsteady flow simulation tool can be extracted from the Figure 81 and Figure 82. Indeed, the NAWC motor No. 6 has a complex 3D grain geometry, so that if the results obtained by the simulation tool match well with the reference ones, it is not casual. Indeed, for this case the propellant data input was the actual one. Moreover, as it is shown in Figure 81, the burning surface area matches very well with the results obtained with the Rocgrain code developed by the CSAR in [14]. Only in the tail off phase, due to the interface intersects the motor case, the code developed in this study has an important error. In the rest of the rocket operation, namely the quasi-steady state phase, the code provides good results. The Rocgrain code uses a minimum distance function to calculate the evolution of the grain geometry. This way, the results from Rocgrain can be used as a reference to check the correctness of the Quasi-3D grain burnback model.

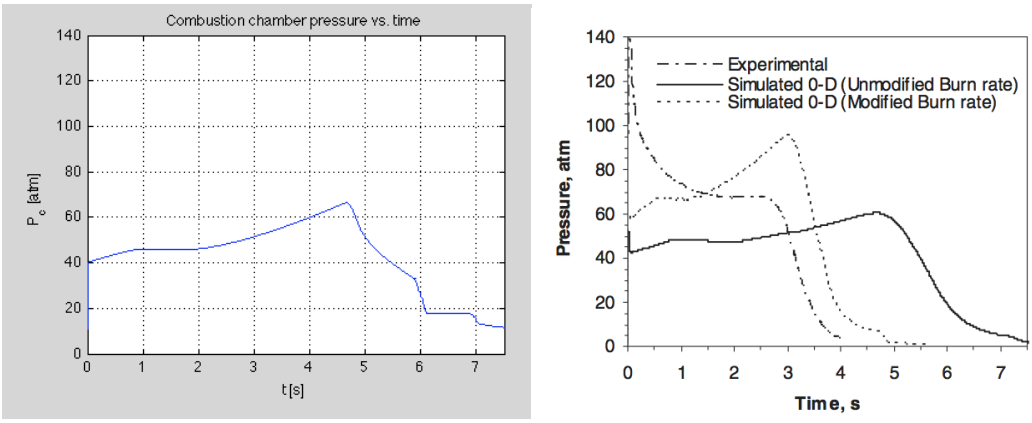

Figure 82. Comparison between chamber pressure time evolution obtained from the simulation tool (L) and the result obtained by Willcox, Brewster, Tang, Stewart and Kuznetsov in source [15] (R) for the NAWC motor No.6.

On the other side, Figure 82 shows the experimental results obtained in [12] for the chamber pressure. As it can be concluded from this figure, the pressure

calculated by the Quasi-3D grain burnback and 0D unsteady flow simulation tool matches well the 0D results obtained by Willcox et al. Only at the end, at the tail off phase the error is significant, in the rest of the phases the results match. The differences between the experimental results, as it is stated in [15], are based on a miss calculation of the burning rate, which in the experiment is so high. But this is not a problem of the burning area calculation but a mistake in the burning rate model. Hence it can be concluded that the Quasi-3D grain burnback and 0D unsteady flow simulation tool provides good results for 3D grain designs. Nevertheless, the tail off calculation should be improved.

Finally, Figure 83 shows the qualitative results of the flow performance variables from the head end to the nozzle inlet for the NAWC motor No. 6. As it has been mentioned in the section 5.3, the 1D model has been partially implemented and so its results are not real but only show the trends of the flow evolution along the motor.

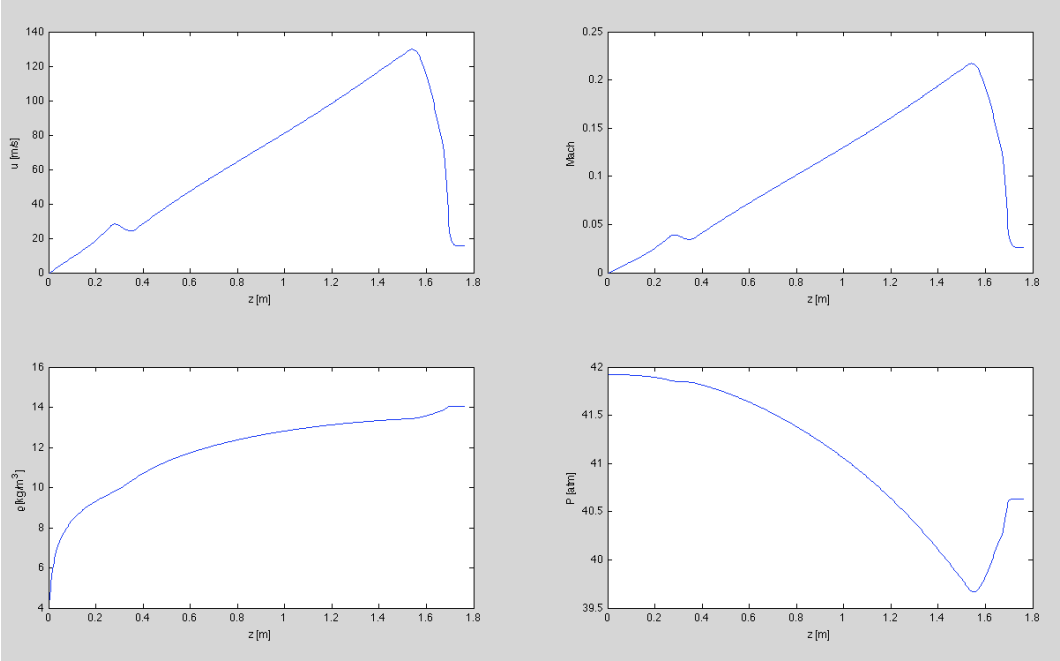

Figure 83. Velocity, Mach number, gas density and static pressure as a function of the longitudinal axis of the rocket motor (z coordinate). Obtained through Quasi-3D and 1D quasi-steady flow simulation tool.

7.2. SIMULATION OF SOME RELEVANT 2D GRAIN DESIGNS

In this section some relevant 2D grain designs simulated by the 2D grain burnback and 0D unsteady flow simulation tool are presented. Although in the this project report only the case of star of 10 slotted tips is enclosed and commented, due to volume of this project report, the rest of the 2D grain designs studied are attached in the ANNEX B of this study. Precisely, in this ANNEX B apart from the 2D grain design of star of 10 slotted tips, the tri-slot, the dogbone and the anchor grain designs and its results are also enclosed. Note that in order to be able to compare the performance results obtained for these four cases, the same input parameters shall be used except for uploaded the text file, that is to say, the SRM related input parameters (see Table 1) as well as numerical parameters (see Table 2) shall be the same for the four cases. The input values uses in this four simulation cases are shown in the Figure 84. Furthermore, in order to prove that the results obtained with the simulation tool are coherent, it is recommended to consult the Figure 3 where the scheme of thrust vs. time curve for different 2D grain geometries is shown.

```
%-------------------------------DATA INPUT-------------------------------%
%%
%INPUT PARAMETERS
N=250;                  %Number of nodes per side in the cartesian grid.
N_Delta_web=60;         %Number of contour lines shown in the Phi plot.
N_bs_plot=16;           %Number of burnt steps shown in grain evolution plot.
Time=17.5;              %Simulation time [s]
n_steps=2000;           %Number of simulation time steps.
t_0=0.5;                %Time duration of the first phase of the simulation (heavy transient).
Delta_t0=0.003;         %Time step during t_0 phase of the simulation.
Delta_t1=(Time-floor(t_0/Delta_t0)*Delta_t0)/(n_steps-floor(t_0/Delta_t0));    %Time step for th
%Distance units selection:
units='cm';
%Units conversion:
[K_units] = units_conversion(units);

%Rocket input parameters:
L_g=1.5;                %Propellant grain length [m]
P_c0=2E6;               %Initial chamber pressure [Pa]
a=8e-5;                 %a parameter of the propellant burning rate law: r=a*Pc^n
n_exp=0.23;             %n exponent of the propellant burning rate law: r=a*Pc^n
gamma_e=1.19577;        %Combustion products gas gamma parameter gamma=Cp/Cv
R_air=287;              %R of air [J/KgK]
R_e=322.21;             %R of the combustion products [J/KgK]
rho_p=1134.75;          %Propellant density [Kg/m^3]
M_mol=0.03875;          %Combustion products molar mass [kg/mol]
De=120;                 %Nozzle exit diameter [mm]
Dt=25;                  %Nozzle throat diameter [mm]
A_e=(pi*(De*1e-3)^2)/4; %Nozzle exit area [m^2]
A_t=(pi*(Dt*1e-3)^2)/4; %Nozzle trhoat area [m^2]
L_plenum=0.1;           %Plenum length [m]
[M_e] = mach_calculation(A_e,A_t,gamma_e);      %Nozzle exit Mach number
P_a=1e4;                %Atmospheric pressure [Pa]

T_c_v=[2462.02544,2470.45874,2474.34792,2476.69366,2478.30590,2479.50156];      %Vector of chamb
P_c_v=1:1:6;            %Vector of chamber pressure data [MPa].
p_T=polyfit(P_c_v,T_c_v,3);         %Polynomial coeficients of the function adjust of T_c=f(P_c)
```

Figure 84. Inputs of the simulation tool for the study some relevant 2D grain designs.

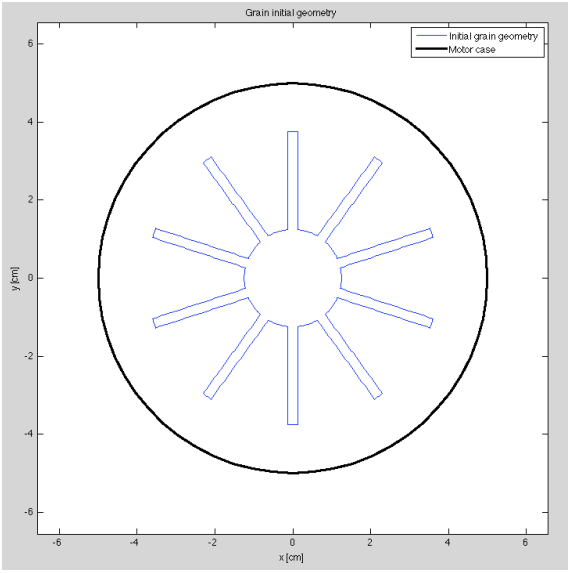

Figure 85. Star of 10 slotted tips grain initial geometry.

In Figure 85 the initial grain geometry of the star of 10 slotted tips is shown. This plot is outputted by the Matlab® in order to verify by the user that the geometry uploaded is correct. This way, the black line represents the motor case and the blue line the initial grain surface.

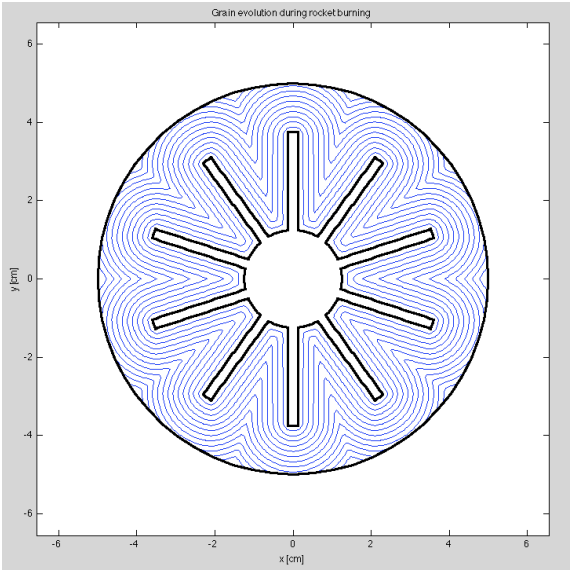

Figure 86. Star of 10 slotted tips grain evolution during the rocket operation.

In Figure 86 the interface evolution at equally spaced time steps is shown. This is a good representation of how grain geometry evolves during the grain burnback. Note that despite of the background grid is Cartesian and with a moderated number of nodes, the front reconstruction is very precise and capable of tracking the sharp cusps formed by this grain geometry. Generally, a polar grid is capable of modeling better geometries with axisymmetric geometries. Nonetheless, the

Cartesian grid here used is enough to deal with the grain burnback of the star of 10 slotted tips. In order to consider the enclosed performance curves, note that one of the advantages of this grain geometry is its high volumetric fraction.

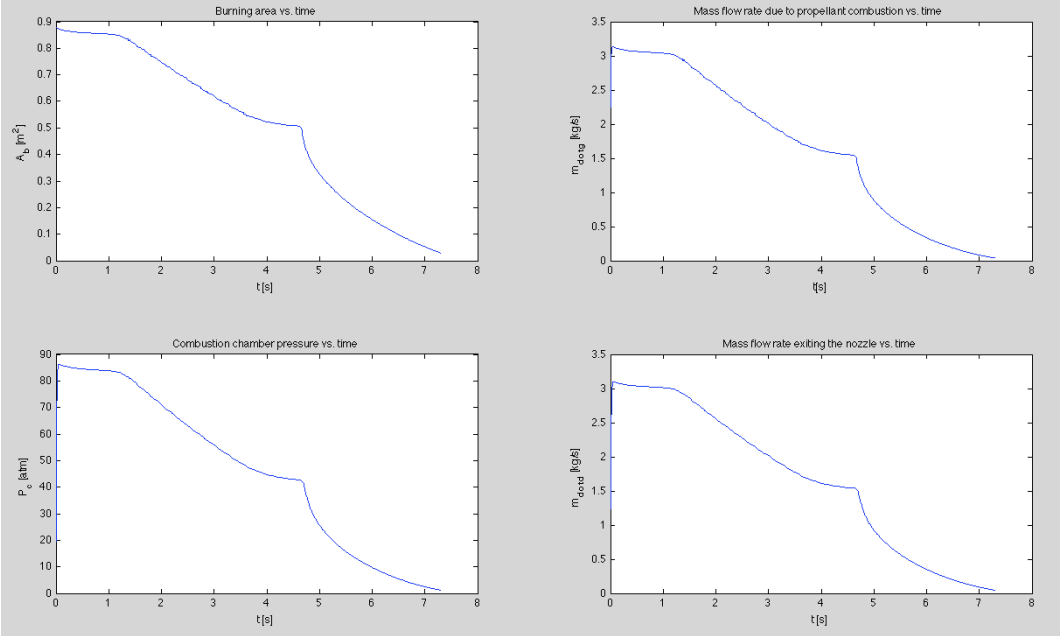

Figure 87. Burning area, mass flow rate due to propellant combustion, chamber pressure and mass flow rate exiting the nozzle time evolutions during the burning process of the star of 10 slotted tips grain.

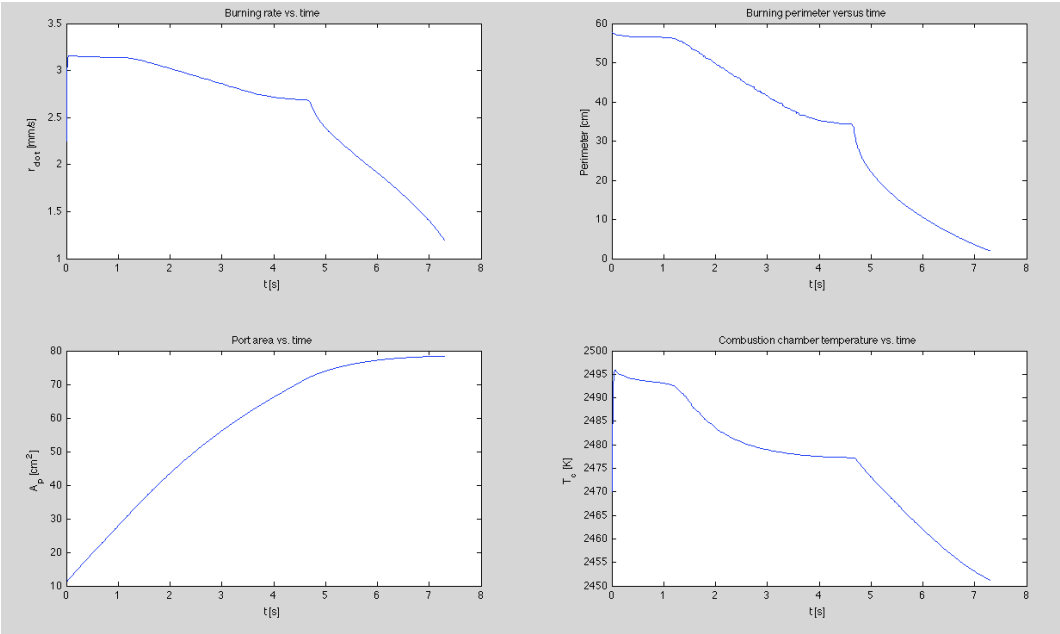

Figure 88. Burning rate, burning perimeter, port area and chamber temperature time evolutions during the burning process of the star of 10 slotted tips grain.

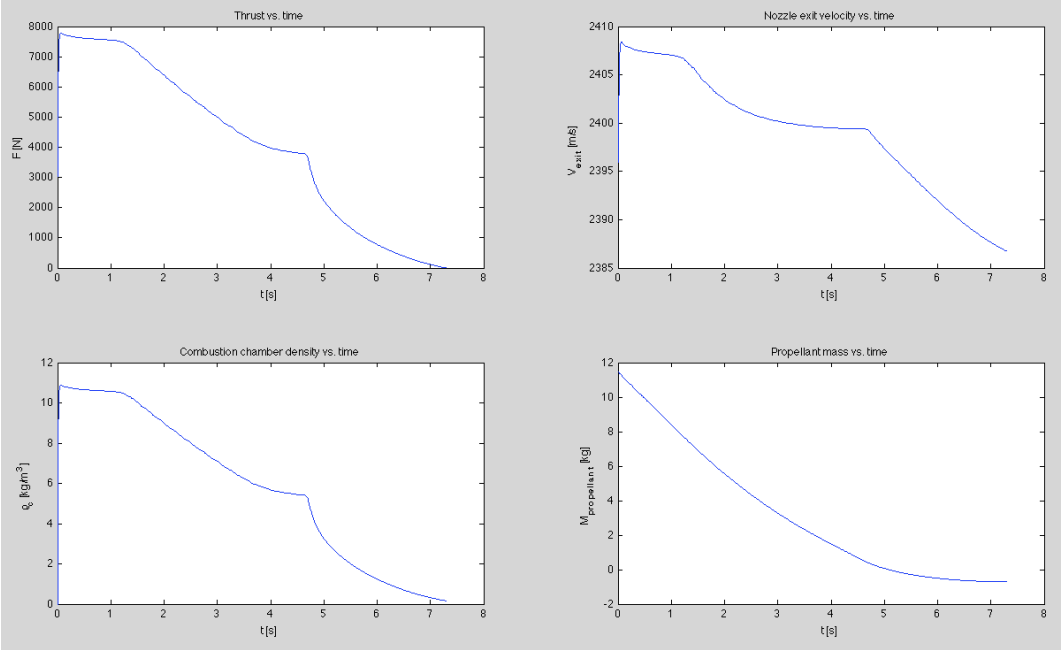

Figure 89. Thrust, nozzle exit velocity, chamber gas density and propellant mass time evolutions during the burning process of the star of 10 slotted tips grain.

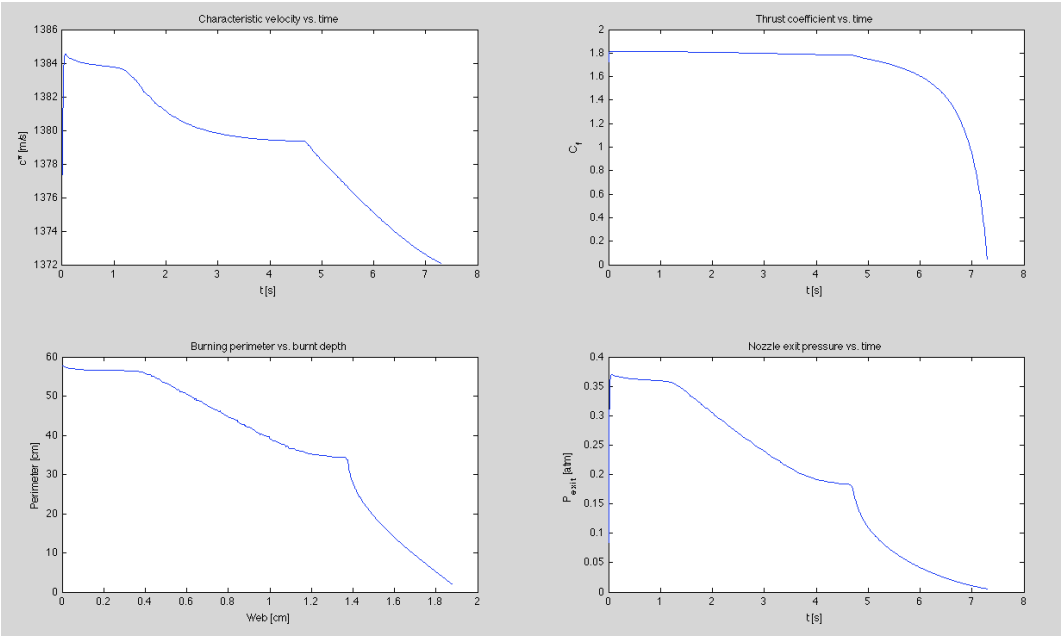

Figure 90. Characteristic velocity, thrust coefficient and nozzle exit pressure time evolutions and burning perimeter vs. burnt depth evolution during the burning process of the star of 10 slotted tips grain.

Figure 87, Figure 88, Figure 89 and Figure 90 show the typical performance curves obtained for the star of 10 slotted tips, which is coherent with Figure 3. The most characteristic feature of this grain geometry is its regressive thrust curve which is very suitable to provide a moderated acceleration to the payload.

8. CONCLUSIONS

In this chapter the main conclusions of the *Study of Grain Burnback and Performance of Solid Rocket Motors* are enclosed. Additionally, soma remarks of future work are also given.

8.1. CONCLUSIONS

First and foremost, it should be pointed out that the goals of the project have been reached as well as the requirements have been fulfilled. Precisely, the final outcome of this study is a Matlab® based numerical code able to perform the grain burnback analysis and simulate the internal ballistics of a SRM for 2D and 3D grain geometries. This code requires the user to introduce a 2D or 3D grain design geometry and some additional SRM related parameters. Furthermore, the calculation is done quickly, not requiring excessive computational effort. This way, the numerical code can be run whether on a small laptop or on a powerful desktop computer lasting at most few minutes to output the performance results for a complicated 3D geometry. Therefore, due to its raid calculation capability, the numerical code is especially useful for preliminary design phase of SRMs. Note that during the preliminary design, several configurations shall be tested. Thereby, preferring efficiency rather than high accuracy of the first SRM's configuration outline, the lasted time calculating becomes a decisive factor. In addition, the academic potential of code generated is undeniable. Due to having attached several videos, the students could be perfectly illustrated with the concerning of grain burnback.

One of the key points in order to obtain a handy numerical code for the user is based on the versatility that offers uploading the geometry of the grain through text files into the Matlab® code. Each file contains a cloud of points that describes specific 2D grain geometry and it is obtained from a discretization process. This way, the code enclosed does not limit the software use by the user to perform the discretization or the CAD design. Although the results here presented have been obtained using the student version of CATIA v5R18 to design and the GID 10.0.7 to discretize, the final user is allowed to use for instance, SolidWorks to design or Ansys to discretize the 2D geometries.

Furthermore, the most important contribution of this study is the Quasi-3D approach to model 3D grain geometries. Indeed, this approach models the 3D grain surface through few 2D reference cross-sections, which are then used to calculate the grain-burnback. In addition, this model allows to calculate quickly and without excessive computational cost the grain burnback of 3D grain

geometries. Although this modelization adds a little error compared with other models such as 3D Level Set Method (see the code verification for the motors NAWC No.13 and No.6), these divergences mostly appear at the end of the thrust vs. time curve, precisely at the tail off phase. Indeed, they are due to how the code computes the area when the interface intersects with the motor case. This computation adds the little error aforementioned. Nonetheless, bearing in mind that the end of the curve is the less important, it can be ascertained that the Quasi-3D approach adjusts reasonably the real thrust vs. time for the NAWC motor No.13 and No.6. Thereby, this fact has to be considered by the user during the interpretation of other SRM tail off results outputted by the numerical code here presented.

Moreover, it is remarkable that the Quasi-3D grain burnback does not consider the effect of the burning edges; that is to say, the Quasi-3D grain burnback propagation is only in the axial direction not considering the longitudinal propagation. The effect of burning edge increases when the ratio grain length/grain diameter (L/D) decreases. So if the L/D is very large, the effect of the burning end is very small. In case of NAWC motor No.13 and NAWC motor No.6. the ratio L/D is large. Consequently, the results of both motors are not affected by the effects of burning ends. Nonetheless, if very short and width SRMs are tested with the Qausi-3D grain burnback code, it is recommended to take into account this effect during the interpretation of the results obtained through the code.

The success of Quasi-3D model could not be possible without the implementation of the 2D grain burnback. Indeed, the 2D grain burnback has been the test-bench of the LSM and then, it has been used to simulate the Quasi-3D grain burnback. As it has been proved, the LSM is a powerful mathematical tool in order to simulate the grain burnback. Although, in case of 2D grain burnback some divergences comparing with experimental results appeared, these are mainly due to the no consideration of erosive burning. This way, once the LSM in 2D was implemented in the Quasi-3D grain burnback, the results obtained considering steady-state matched perfectly with those obtained with LSM in 3D or with the Rockgrain software. This last software, developed by Willcox et al ([14] and [15]) in the Center for Simulation of Advanced Rockets (CSAR) at the University of Illinois, is based on a signed minimum distance function. Precisely, the methodology presented by the software Rocgrain is very similar to the LSM because the LSM function is very close to a signed minimum distance function. Therefore, bearing in mind that the three methodologies possess the same mathematical background with little differences, the robustness presented by Quasi-3D grain burnback with 2D LSM application is again proved.

One of the noteworthy effects perceived during the implementation and subsequent verification of the code is the transient burning. The transient burning takes place just after the ignition of the SRM begins. This way, it lasts a short period of the thrust vs. time curve (indeed, less than a second) but it can compromise the convergence of the full simulation. Hence, in order to simulate this transient burning, it is necessary to apply very small time step. If the time steps are not small enough, the simulation process begins to oscillate not converging to a proper result. Nonetheless, once the transient burning ends, the time step can be incremented because burning process uses to be more stable. In this second part the time step is controlled by the LSM. Note that with bigger time step, lower computational cost but less accurate results can be obtained. This way, the final user of the code has deal with this trade-off assuming the best solution. In order to give the opportunity to apply his/her own criteria to the user, in the three numerical codes here presented two time steps are considered: the former is applied in the transient burning; the latter is applied in the rest of the simulation, which is under the domain of the LSM.

To sum up, the issue of flow models shall be discussed. As it has been aforementioned, the 0D steady model (lumped model) is too simple to obtain accurate results. Therefore, the 0D unsteady flow model with 2D grain burnback has been directly implemented. Once the validity of LSM to simulate for the 2D grain burnback as has been verified, the 0D unsteady flow model has been implemented for the case of Quasi-3D grain burnback. Then, after the validation of this code, the Quasi-3D grain burnback model has been implemented with a 1D quasi-steady flow model. Note that the use of 2D grain burnback and 1D quasi-steady flow model is nonsense. Indeed, the 1D quasi-steady flow model must be used at least with a Quasi-3D grain burnback. Furthermore, in this study the 1D quasi-steady flow models from Stewart et al and Yildirim have been tested. During these tests, it has been noticed that the Stewart et al method has some problems caused by the port area term and its derivative. It should be remarked that in the Quasi-3D burnback analysis approach, the port area term evolves sharply because the grain is modeled with just few cross-sections. So that, the Quasi-3D model provides to the flow solver an input which is sharp. Thus, any flow model will experience some problems due to this sharp input. This way, due to the Stewart et al model deals with the terms in conservative form, it is more sensible to sharp inputs. On the other side, this problem is also experienced by the Yildirim method, but it is less sensible to it because its formulation is in non-conservative form. Therefore, the model used in this study is the Yildirim's because it handles better the term of the port area.

After the previous discussion, it should be pointed out that the results reached with Quasi-3D grain burnback and 1D quasi-steady simulation tool are qualitative.

With the purpose of obtaining better performance results, an own 1D flow model must be developed. Nonetheless, the development of 1D flow model alone has the enough entity to do a study only focused on this point, such as some master thesis. Bearing in mind that the generation of an own 1D flow model is out of scope of this study, the qualitative results obtained from the third simulation tool are satisfactory for this study. So, the implementation of an own 1D flow model with Quasi-3D grain burnback is considered as future work.

8.2. FUTURE WORK

After the realization of this study, many future steps can be done in order to broad the simulation capabilities of the code here presented. Some of these future works are listed as next:

- Consideration of the erosive burning effects in the simulation code. This phenomenon becomes important in SRMs were the ratio L/D is large. This way, based on the work here presented, the effects of erosive burning can be first implemented in the model of 2D grain burnback and then, added to the Quasi-3D grain configuration.
- Consideration of other phenomena such as the effects of the movement of the rocket, the nozzle ablation or the heat transfer in the grain burnback as well as in the internal ballistics of the SRMs. Due to the complexity of these aspects (see [8]), they have the enough entity to do a study only focused on these points, such as some master thesis, being the starting point the study here presented.
- Generation of an own 1D flow model in order to obtain better results for the Quasi-3D grain burnback. As it has been aforementioned, this is also a complex matter and so that, it possess the entity to do a study only focused on this point, such as some master thesis.
- Generation of a Graphical User Interface (GUI) in order to ease the input of parameter to the user. This GUI would not be so necessary to input the text files containing the clouds of points generated through the discretization process. It would be more focused on inputs related with SRM parameters (see Table 1,Table 3 and Table 5) and with numerical parameters (see Table 2, Table 4 and Table 6). Additionally, due to the fact that each simulation tool has each own inputs, the GUI shall be adapted to each one.

9. BUDGET

This section covers the budget for the realization of the project *Study of Grain Burnback and Performance of Solid Rocket Motors.* Note that the cost of this work is mainly estimated from the author's engineering work, the hardware (HW) infrastructure and software (SW) licenses, inter-alia. The study lasted for 5 months; so, the budget here presented is considering approximately 100 working days. In addition, it is important to remark that the currency used to show the final values is the Euro [€]. All of the intermediate costs are expressed with two decimals and finally, in the total budget, the total cost is rounded-off. The consideration of taxes is out of scope of this budget.

9.1. ENGINEERING WORK

In this section the cost corresponding to engineering work is presented. An average cost of 14€ per hour is considered. As it is observed in Table 15, in the budget of engineering work three main contributors are taken into account: the generation of the code for 2D grain burnback and 0D unsteady flow simulation; the generation of the code for Quasi-3D grain burnback and 0D unsteady flow simulation and the generation of the Quasi-3D grain burnback and 1D quasi-steady flow simulation. The previous research and learnt process from the state of the art as well as the writing of the documents are considered independently.

For three main contributors aforementioned, the required hours and the cost of developing are shown in Table 15. This way, the necessary hours to develop the simulation tool of 2D grain burnback and 0D unsteady flow are 120 hours. So, if a cost of 14€/h is considered, the final cost of this code is around 1680€. In case of Quasi-3D grain burnback and 0D unsteady flow, the required hours are 103, which means an estimated cost of 1442€. Finally, for the generation of Quasi-3D grain burnback and 1D quasi-steady flow code 87hours are stipulated, which means a final cost of 1218€. Note that the code development process for each case is sequential; that is to say that it is unfeasible generate the code of the Quasi-3D grain burnback before having generated the code for 2D grain burnback. Analogously, it is unfeasible to develop the code for 1D quasi-steady flow without having done the case of 0D unsteady flow. The knowledge and code robustness acquired during the progress of the study have to be considered in the final budget. Therefore, if it is aimed to study a SRM considering Quasi-3D grain burnback and 1D quasi-steady flown, the final cost of the engineering work shall be almost the same as the total engineering cost shown in Table 15 (5740€).

Concept		Quantity [h]	€/h	Cost
Stat of the art research and learning		30	14	420
Generation of the code for 2D grain burnback and 0D unsteady flow simulation	Development of geometry_loading.m	4	14	56
	Development of initial_geometry_organization.m	6	14	84
	Development of curve_tangent_normal.m	5	14	70
	Development of cartersian_grid.m	2	14	28
	Development of minimum_distance_function.m	8	14	112
	Development of interface_tracker.m	16	14	224
	Development of order_vector_points.m	10	14	140
	Development of rocket_performance_0D_transient.m	4	14	56
	Development of phi_one_time_step_evolution.m	5	14	70
	Development of other functions	25	14	350
	Development of the main code and integration of all of the functions	10	14	140
	Code verification and solution of errors	20	14	280
	Generation of videos, final results, etc.	5	14	70
	Subtotal of 2D grain burnback and 0D flow	120	14	1680
Generation of the code for Quasi-3D grain burnback and 0D unsteady flow simulation	Development of geometry_loading_3D.m	2	14	28
	Development of chamber_volume.m	2	14	28
	Development of curve_tangent_normal_3D.m	2	14	28
	Development of minimum_distance_function_3D.m	2	14	28
	Development of grain_cross_sections_pmt.m	14	14	196
	Development of rocket_performance_0D_transient_quasi_3D.m	3	14	42
	Development of interpolate_cross_section.m	14	14	196
	Development of initial_geometry_organization_3D.m	4	14	56
	Development of other functions	14	14	196
	Adaptation of 2D grain burnback and 0D flow functions to Quasi-3D and 0D flow	8	14	112
	Development of the main code and integration of all of the functions	12	14	168
	Code verification and solution of errors	20	14	280
	Generation of videos, final results, etc.	6	14	84
	Subtotal of Quasi-3D grain burnback and 0D flow	103	14	1442

Generation of the code for Quasi-3D grain burnback and 1D quasi-steady flow simulation	Development of dif_A_p.m	2	14	28
	Development of grain_cross_sections_pmt_1D.m	5	14	70
	Development of interpolate_bt_cs.m	4	14	56
	Development of nozzle_inlet_mach.m	2	14	28
	Development of perimeter_cross_section.m	2	14	28
	Development of rocket_performance_1D_steady_Q3D.m	24	14	336
	Adaptation of previous functions to Quasi-3D and 1D flow	10	14	140
	Development of the main code and integration of all of the functions	12	14	168
	Code verification and solution of errors	20	14	280
	Generation of videos, final results, etc.	6	14	84
	Subtotal of Quasi-3D grain burnback and 1D flow	87	14	1218
Writing of the document		70	14	980
Subtotal		410	14	5740

Table 15. Table of Engineering work cost.

9.2. HARDWARE AND WORKING INFRASTRUCTURE

In this section the hardware and the additional working infrastructure utilized during the study is considered. The main hardware component used to develop the study is a MacBook Pro of 15 inches. This laptop was not specifically bought to perform the study. In addition, it is intended to use for future work too. Consequently, it is estimated an amortization period of 3 years (1095 days). As it has been aforementioned, 100 working days are considered.

Concept	**Quantity**	**Magnitude**	**€/ Magnitude**	**Amortization period [day]**	**Duration of study [days]**	**Cost [€]**
MacBook Pro 15 inches	1	computer	1927	1095	100	175.98
Electricity	5	month	40	-	-	200
Water & installations	5	month	60	-	-	300
Subtotal						**675.98**

Table 16. Table of Hardware and working infrastructure cost.

In case of electricity and water and installations, 5 working months have been considered. For each concept an average value for monthly cost are given (40€/month for electricity and 60€/month for water & installations). This way, the total cost for hardware and working installations is around 676€.

9.3. SOFTWARE LICENSES

In this section the cost of difference software licenses used during the study is attached. To corroborate the value of *Microsoft Office Home and Student 2010* license, consult [20]. In case of the license of Matlab®, it is necessary to register in the official web page of MathWorks® [21] to consult the price of the software license and buy it. In case of the Computer-aided Design software, CATIA v5R18 has been used. Due to the fact that this software is student version, the additional cost of the license is equal to 0. Nonetheless, if CATIA v5R18 professional version is used, the cost of this study would be increased in more than 200€ depending on which module is bought [22]. Additionally, the fourth remarkable software used is the GID 10.0.7 for MacOSX. As it has been aforementioned in the acknowledgements, this software has been freely provided by International Center for Numerical Methods in Engineering (CIMNE) [17]. Therefore, the additional cost of using this software is null. To sum up, it should be pointed out that to convert the value from USD to Euros an ex-change rate of 1.3 is applied. This way, Table 17 shows software licenses cost. As it can be observed, the final Software licenses cost is 592.30€.

Concept	Quantity	$/License	€/License	Cost [€]
MATLAB R2012b	1	-	500	500
CATIA v5R18 Student version	1	-	0	0
GID 10.0.7 for MacOSX	1	-	0	0
Microsoft Office Home and Student 2010	1	119.99	92.30	92.30
VLC Media Player	1	-	0	0
Subtotal				**592.30**

Table 17. Table of Software licenses cost.

9.4. TOTAL

The total cost of the project *Study of Grain Burnback and Performance of Solid Rocket Motors* is given in Table 18. For that, subtotal costs given in Table 15, Table 16 and Table 17 are taken into account.

Concept	Cost [€]
Engineering Work	5740
Hardware and Infrastructure	675.98
Software Licenses	592.30
Total	7008.28
Total cost (rounded off)	**7000**

Table 18. Table of Total cost.

In Table 18 the total cost of the study is rounded off in order to obtain more realistic final cost. This way, the final cost project *Study of Grain Burnback and Performance of Solid Rocket Motors* is 7000€. As it has been aforementioned in the section 9.1, the final cost of a study of a SRM only considering Quasi-3D grain burnback and 1D quasi-steady flow shall be similar to the final cost given in Table 18. In case of simpler modeling (e.g. Quasi-3D grain burnback and 0D unsteady flow) the final cost of the study shall be exhaustively calculated. That is to say that although this kind of study shall be cheaper, costs such the time required to achieve the knowledge or the time lasted writing the documents should be also computed in the final value of the simpler study.

10. ENVIRONMENTAL EFFECTS

The *Study of Grain Burnback and Performance of Solid Rocket Motors* is entirely computer-based. Therefore, no wastes were generated during this work, except for the electrical energy consumption and the printed copy. The electrical energy was used to feed the computer as well as the room light. Nevertheless, it is assured that the electrical energy consumption has been performed in a rational way. In case of the printed copy, it is believed that I would follow the protocols to recycle paper correctly.

Nonetheless, it should be pointed out that the *Study of Grain Burnback and Performance of Solid Rocket Motors* intends to contribute to the reduction of testing in SRM which itself means a reduction of environmental amiss effects provoked by the testing and manufacturing of SRM. For instance, as it claimed in [17], most of the perchlorate contamination in the US is attributed to the use of ammonium perchlorate as an oxidizer and primary ingredient in SRM fuel. The majority of sites where perchlorate was detected as a contaminant in groundwater are associated with anthropological (man-made) sources, namely activities associated with the manufacturing or testing of solid rocket fuel by the Department of Defense (DOD) of the US and by the National Aeronautics and Space Administration (NASA). Indeed, perchlorate is an inorganic anion, consisting of four atoms of oxygen and one atom of chlorine, which is released in the environment when the highly soluble salts (e.g. ammonium perchlorate) are dissolved in water. It is not easily degraded and it does not bind well to other matter such as mineral surfaces or activated carbon; so that, perchlorate can persists for decades and move freely within the bodies of water. Bearing in mind that ammonium perchlorate is one of the most widely used oxidizers of SRM, it is comprehensible that many locations have been contaminated testing, manufacture and maintenance of SRM. Therefore, if the number of required tests is reduced during the design to fly phase of SRM, fewer ammonium perchlorate will be burnt. Thus, lowering the quantity of tests, the emissions of perchlorate are reduced. For that, it is fundamental to develop computer-based SRM burnback and simulation tools, such as the one presented through this study.

Consequently, it has been proven that the potential environmental benefits of this study are greater than the wastes produced during its generation. Undeniably, the environmental damage caused by this study is insignificant and far below the legal requirements. In fact, the simulation tool developed is an indispensible step for making eco-friendlier SRM design processes.

11. REFERENCES

11.1. BOOKS, LECTURES AND ARTICLES

[1] M.J.L. Turner. *Rocket and Spacecraft Propulsion.* Springer (Praxis Publishing), third edition, (2009).

[2] R. Carreras. *Disseny de la Geometria del Gra: optimització de la llei d'empenta. Klemmung. Sistemes d'ignició.* Lecture notes of the curse rocket motors held at ETSEIAT-UPC, (fall 2010).

[3] S. Osher and J. Sethian. *Fronts Propagating with Curvature-Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations.* Journal of Computational Physics, volume 79, pages 12-49 (1988).

[4] J.A. Sethian. *Level Set Methods and Fast Marching Methods.* Cambridge University Press, second edition, (1999).

[5] J.A. Sethian. *Numerical Algorithms for Propagating Interfaces: Hamilton-Jacobi Equations and Conservation Laws.* Journal of Differential Geometry, volume 31, pages 131-161 (1990).

[6] J.A. Sethian, *Curvature and the Evolution of Fronts.* Communication of Mathematical Physics, 101, (1985).

[7] P. Persson. *The Level Set Method*. Lecture Notes, MIT 16.920J / 2.097J / 6.339J Numerical Methods for Partial Differential Equations (March 2005).

[8] E. Cavallini. *Modelling and Numerical Simulation of Solid Rocket Motors Internal Ballistics.* Ph.D. Thesis, Sapienza Università di Roma (Academic year 2008-2009).

[9] C. Yildirim. *Analysis of Grain Burnback and Internal Flow in Solid Propellant Rocket Motors in 3-Dimensions.* Ph.D. Thesis, Middle East Technical University (March 2007).

[10] *Solid Propellant Grain Design and Internal Ballistics.* NASA SP-8076. NASA (March 1972).

[11] G.P. Sutton and Oscar Biblarz. *Rocket Propulsion Elements.* John Wiley & Sons, seventh edition (2001).

[12] F.S. Blomshield. *Pulsed Motor Firings*. NAWCWD TP 8444, Naval Air Warfare Center Weapons Division, China Lake, CA (August 2000).

[13] D.S. Stewart, K.C. Tang, S. Yoo, M.Q. Brewster and I.R. Kuznetsov. *Multi-scale modeling of solid rocket motors: Time integration methods from computational aerodynamics applied to stable quasi-steady motor burning.* TAM Report No. 1054. Theoretical and Applied Mechanics. University of Illinois at Urbana-Champaign, (October 2004).

[14] M.A. Willcox, M.Q. Brewster, K.C. Tang and D.S. Stewart. *Solid Propellant Grain Design and Burnback Simulation using a Minimum Distance Function.* AIAA 2005-4350 (2005).

[15] M.A. Willcox, M.Q. Brewster, K.C. Tang, D.S. Stewart and I.R. Kuznetsov. *Solid Rocket Motor Internal Ballistics Simulation Using Three-Dimensional Grain Burnback.* Journal of Propulsion and Power, Vol.23, No.3 (May-June 2007)

[16] *Solid Rocket Motor Performance Analysis and Prediction.* NASA SP-8039. NASA (May 1971).

11.2. WEBSITES

[17] International Center for Numerical Methods in Engineering (CIMNE). *GID, The Personal Pre and Post Processor.* http://www.gidhome.com/home (last access 2012/12/31).

[18] Aerospaceweb.org. *Solid Rockets & Aluminum*. http://www.aerospaceweb.org/question/propulsion/q0246.shtml (last access 15/12/2012).

[19] NASA. *Engineering Innovations*. http://er.jsc.nasa.gov/seh/536823main_Wings-ch4.pdf (last access 2012/12/26).

[20] Microsoft. *Buy Microsoft Office 2010.* http://office.microsoft.com/en-us/buy/buy-office-2010-FX101843016.aspx?WT.mc_id=ODC_enus_ O15PLO_Home (last access 2012/12/26).

[21] MathWorks®. http://www.mathworks.com/products/?s_cid=global_nav (last access 2012/12/26).

[22] Kingdom Classifieds (KC). *CATIA v5r18 CAD on DVD CATIA v5.18+documentation/help for sale.* http://www.interking.com/ads/catia18.htm (last access 2012/12/26).

[23] California Department of Toxic Substances Control. *Section 3. Sources of perchlorate in the environment.* http://www.dtsc.ca.gov/LawsRegsPolicies/ Regs /upload/HWMP_WS_dPerch-Sec3.pdf

[This page is intentionally left blank.]