Towards a computational framework for coupling contact and fragmentation in impact modelling

Riccardo Brambilla

Delft University of Technology

TU Delft

Towards a computational framework for coupling contact and fragmentation in impact modelling

by

Riccardo Brambilla

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Friday March 8, 2024 at 11:00 AM.

Student number:	5457505	
Project duration:	April, 2023 – March, 2024	
Thesis committee:	Dr. B. Giovanardi,	TU Delft, supervisor
	Dr. B. Chen,	TU Delft, chair
	Dr. M. Gerritsma,	TU Delft, external member

Cover:	Rendering of a satellite fragmentation process, courtesy of [1] (Modified)
Style:	TU Delft Report Style, with modifications by Daan Zwaneveld

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Acknowledgements

I would like to thank my supervisor Bianca Giovanardi for the continued supervision, tutoring and for introducing me to the field of numerical modelling.
Furthermore, my journey would not have been possible without my family continuous support and patience.
Finally, I would like to thank my friends, which have all contributed to make the person I am today. Special thanks to the people who worked with me in these years at TU Delft, both in university classes and in extra-curricular activities, who made this journey much more interesting and exciting.

Riccardo Brambilla
Delft, February 2024

Executive Summary

The exponential increase of LEO payloads created by the New Space economy increases greatly the risk of orbital collision. One of the strategies to mitigate this risk is the development of robust spacecraft shielding structures. This not only safeguards the mission's success but also prevents the spacecraft from fragmenting into additional debris in the case of an impact. Most shielding structures are designed to withstand impacts with objects smaller than 1cm, since this is the threshold size for ground based debris tracking. The impact of much smaller objects colliding at orbital velocities can still result in significant consequences, since most of orbital flybys occur at relative speeds between 2 and 10 km/s. When an impact happens, these encounters are called hypervelocity impacts (HVIs).
Gaining more insight in HVI is vital for the design of better spacecraft shielding structures. Direct observation of the impacts is almost impossible, and thus designers have traditionally relied heavily on experimental campaigns and semi-analytical methods. With the increase of computational power experienced in the last few decades, numerical simulations have gained traction as the go-to methodology to study HVIs. No simulation technique has been found in the literature that is able to model the entire velocity range of interest and that can be scaled to full-scale satellite impact simulations. The energies usually involved in HVI lead to a total projectile fragmentation, full hydrodynamic behaviour of the materials involved and a heavy coupling of strength and thermal effects. Therefore, simulating these events proves extremely difficult, and many different simulation techniques have been proposed.
The total projectile fragmentation means that fracture and contact modelling are fundamental for the simulation outcome and are heavily coupled between themselves and with stress wave propagation. Modelling fragmentation without hindering stress wave propagation has been judged as the first necessary step to develop the desired HVI methodology. Therefore, this thesis will focus on coupling a suitable fracture simulation with a contact algorithm. The former is identified as the Discontinuous Galerkin/Cohesive Zone Method (DG/CZM) Finite Element (FE) framework, which has shown good wave propagation and fracture modelling properties. The chosen contact formulation is the Decomposition Contact Response (DCR) algorithm, a constraint-based formulation which revolves around momentum decomposition. The algorithm is available in the SFC legacy library, which is coupled with the DG/CZM formulation implemented in the Summit FE software.
The DCR methodology is firstly adapted to a discontinuous FE formulation, without including fracture. Then, fracture is introduced by activating the full capabilities of the DG/CZM formulation. Results have been numerically verified with simple analytical cases and for energy conservation. While the coupling of DG and DCR is promising, the introduction of fracture did not lead to results displaying acceptable error margins. The results have been deeply investigated and the issues identified, and a solution concept has been proposed.

Contents

Nomenclature

Abbreviations

Abbreviation	Definition
1D	One dimensional
2D	Two dimensional
3D	Three dimensional
ALE	Arbitrary Lagrangian-Eulerian
ASAT	Anti-SAtellite Test
BLE	Ballistic Limit Equations
CFRP	Carbon Fiber Reinforced Composites
CEM	ContactEnergy Momentum SFC class
CG	Continuous Galerkin
CS	Contact Surface
DCR	Decomposition Contact Response
DEM	Discrete Element Method
DG	Discontinuous Galerkin
DG/CZM	Discontinuous Galerkin / Cohesive Zone Method
DOF	Degrees Of Freedom
EOS	Equation Of State
FE	Finite Element
FS	Function Space
FER	Finite Element Reconstruction
GEO	Geosynchronous Earth Orbit
HiVI	High Velocity Impacts
HVI	HyperVelocity Impacts
ID	IDentifier
JC	Johnson-Cook
LEO	Low Earth Orbit
NSFE	Node Separation Finite Elements
ORQ	Orthogonal Range Query
SG	Steinberg-Guinan
SPH	Smoothed Particle Hydrodynamics
TSL	Traction Separation Law
WS	Whipple Shield

Symbols

Symbol	Definition	Unit
$\mathbf{B}$	body forces	[Nm^3/kg]
C_I	Computational cost	[-]
C_i	number of contact interactions for time step i	[-]
c_E	Elastic wave speed	[m/s]
c_p	Plastic wave speed	[m/s]
c_s	Shock wave speed	[m/s]
c_v	Specific heat at constant volume	[J/kgK]
E	Young's modulus	[MPa]

Symbol	Definition	Unit
E_{kin}	Kinetic energy	[J]
$\mathcal{E}$	Internal energy	[J]
e	DCR restitution coefficient	[-]
$\rceil$	Specific internal energy	[J/kg]
f_{DG}	number of faces in the DG mesh	[-]
$\mathbf{f}$	force vector	[N]
G_c	fracture energy	[J]
$g(\mathbf{X})$	DCR costraint function	[m]
$\mathbf{g}$	DCR gap vector	[m]
h_s	mesh size	[m]
K	Bulk modulus	[MPa]
M	Mass matrix	[kg]
$\mathbf{N}^-$	outward unit surface normal vector of a finite element	[-]
$\mathbf{n}$	outward unit surface normal vector of a contact mesh triangle	[-]
n	number of elements in the FE mesh	[-]
$n_{interface}$	number of DG/CZM interfaces in the FE mesh	[-]
n_t	number of time steps in a simulation	[-]
$\mathbf{P}$	First Piola-Kirchhoff stress	[MPa]
p	Hydrostatic pressure	[MPa]
$\mathbf{p}$	Linear momentum	[kg m/s]
T	temperature	[K]
$\mathbf{T}$	surface traction	[MPa]
$\mathcal{T}$	traction per unit area	[MPa]
t	time	[s]
$\mathbf{u}_i$	displacement of node i	[m]
$\dot{\mathbf{u}}_i$	velocity of node i	[m/s]
$\ddot{\mathbf{u}}_i$	acceleration of node i	[m/s^2]
u	displacement of node i	[m]
V	Volume	[m^3]
v	Velocity	[m/s]
W_{in}	internal elastic energy [J]	
W_{pl}	plastic work	[W]
W_{proj}	DCR projection work	[W]
W_{sep}	Work of separation for cohesive elements	[W]
$\mathbf{X}_i$	position vector of node i	[m]
α	DG/CZM interface status parameter	[-]
β_{NM}	Newmark parameter	[-]
β_s	DG/CZM penalty parameter	[-]
β_{tq}	Taylor-Quinney coefficient	[-]
γ	DG/CZM tangential to normal weight ratio	[-]
γ_{NM}	Newmark parameter	[-]
δ	TSL separation	[m]
δ_{ij}	Kronecker's delta	[-]
$\delta\boldsymbol{\varphi}$	DG trial function	[m]
ε	Lagrangian strain	[-]
$\dot{\varepsilon}$	Lagrangian strain rate	[1/s]
μ	Couloumb's friction coefficient	[-]
μ_i	Artificial viscosity	[-]
ν	Poisson's ratio	[-]
ρ	Density	[kg/m^3]
σ	Cauchy stress	[Mpa]
σ_c	Critical stress threshold for fracture initiation	[Mpa]
σ_{hel}	Elastic limit	[Mpa]

Symbol	Definition	Unit
σ_y	Yield stress	[MPa]
$\boldsymbol{\varphi}$	displacement field	[m]
Ω	domain	[-]

1

Introduction

Space exploration is currently experiencing an increasing growth, thanks to the latest technological advancements. A New Space economy is blooming, which promises easier and extended access to Space for scientific, commercial and military operations. The New Space concept pivots around reducing costs and increasing standardization, aiming mostly at small size satellites in Low Earth Orbit (LEO). LEO offers many advantages over higher orbits, the most important being the lower energy needed for satellite placements. This, along with low communication latency, makes it a very convenient location also for megaconstellations like OneWeb [2] or Starlink [3]. The focus on smaller, cheaper satellites means an increase in launches and LEO payload population. In figure 1.1 the increase in launches up to 2020 is shown.

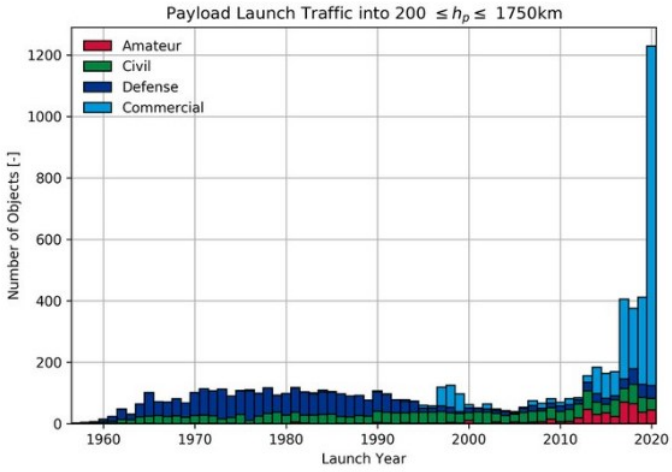


Figure 1.1: LEO launches between 1960 and 2020, courtesy of [4]. The New Space population increase is clearly visible as the spike in the right of the plot. Notable is also the decrease in defense-related launches after the end of the Cold War.

This increased rate of launch is surpassing the rate of decay for LEO objects. Indeed, thanks to the combined effect of non-spherical gravity, atmospheric drag and solar pressure, objects in LEO lose altitude and burn in the atmosphere [5]. Nonetheless, the increase in number of LEO objects happens an order of magnitude faster than their decrease [6].
A higher amount of LEO objects also means that the debris number is projected to increase, raising the risk of collisions. Debris is defined as an artificial object which no longer serves its purpose [6]. Thus, this broad categorization includes different objects with different sizes, from micrometers (e.g., paint fragments) to meters (e.g., discarded rocket stages). The main source of debris are collisions and fragmentation, which can be caused by deliberate acts (e.g., AntiSatellite (ASAT) tests) or accidental occurrences (e.g., the Iridium-Cosmos collision in 2009). Additionally, some are also created by older satellites and launchers which were not designed for a re-entry burn at the end of their service life.

All these objects travel at extremely high speeds (for LEO, up to 7.8 km/s), which means that for an average size spacecraft the impact with a 1 cm size object can already be catastrophic. As an example, in figure 1.2 the effect of a 1.2 cm diameter sphere on a 18 cm thick aluminium block is shown. The danger is even higher for smaller size spacecrafts like micro- and cubesats. Indeed, smaller size means less inertia, and thus the spacecrafts could be more easily tumbled by smaller impactors. Additionally, these smaller systems also have smaller control systems which make it harder to perform evasive maneuvers and makes more likely to be hit and report serious damage.

Figure 1.2: Impact of a 1.2 cm diameter aluminium sphere into an 18 cm thick aluminium block, courtesy of [7]. On the top of the target the impact crater is clearly visible, along with the ejecta baffles protruding out of its sides. On the bottom, spall fracture can be observed clearly.

Small sized debris can therefore still lead to a catastrophic failure in the case of impact. Additionally, the smaller sized debris are hard to track, since Earth-based radars suffer from low resolution caused from atmospheric disturbances. Therefore, even with operating controllable spacecrafts it is hard to avoid collisions with millimeter sized objects. Thus, most spacecraft are shielded for resisting impacts with objects up to 1 cm in size. The same holds for bigger orbital structures like the International Space Station (ISS), which even if performing routinely evasive maneuvers to avoid trackable debris, is still protected from Whipple shields in the vital areas.
Moreover, impacts between debris can also happen and are almost impossible to avoid. If projected into the future, considering an increased LEO debris density, it could lead in the worst case to a Kessler's syndrome situation, where the debris density gets so high that any object passing through LEO would experience collisions and most likely get destroyed, leading to humanity being effectively unable to leave Earth.
Even without assuming this worst case, available data clearly shows the potential threat of any collision and of its related increment in LEO debris population. Figure 1.3 shows the increments corresponding to ASAT tests (2007, 2021) or accidental collisions (2009).

LEOs are the main subject of interest for the increase in debris density, but recent developments have highlighted the risks related to impacts in Geo-Synchronous Orbits (GEO) [7] as well. Indeed, previous works assumed that relative velocities in GEO would not surpass 500 m/s, but it has now been proven that flybys with velocities up to $v = 3$ km/s are not rare [8]. Satellites in GEO also have traditionally a bigger size compared to those in LEO and this makes the survival to impacts more likely. Additionally, it has to be considered that not only man-made debris are present in orbit: during interplanetary transit and planetary missions (e.g., Moon roving), micrometeoroids pose a threat to man-made vehicles and habitats. In interplanetary space non-trackable micro-meteoroids can get to relative velocities of 72 km/s [9], while for Moon habitats the maximum velocity is around 10km/s [10].
Considering all the mentioned trends and the plans for human expansion in the Solar system, like the on-going Artemis program [11], it is reasonable to expect that future spacecrafts will experience a higher amount of impacts from small sized fragments. Thus, the space industry is showing a renovated interest in design and development of effective shielding strategies meeting the strict requirements for spaceflight.
The most common shielding strategy used in spacecrafts is the Whipple shield [12]. It consists in two spaced plates, of which the outer is called "bumper" and the inner is the satellite wall. When the bumper is impacted, it fragments and absorbs most of the impactor's kinetic energy. Thus, the risk of

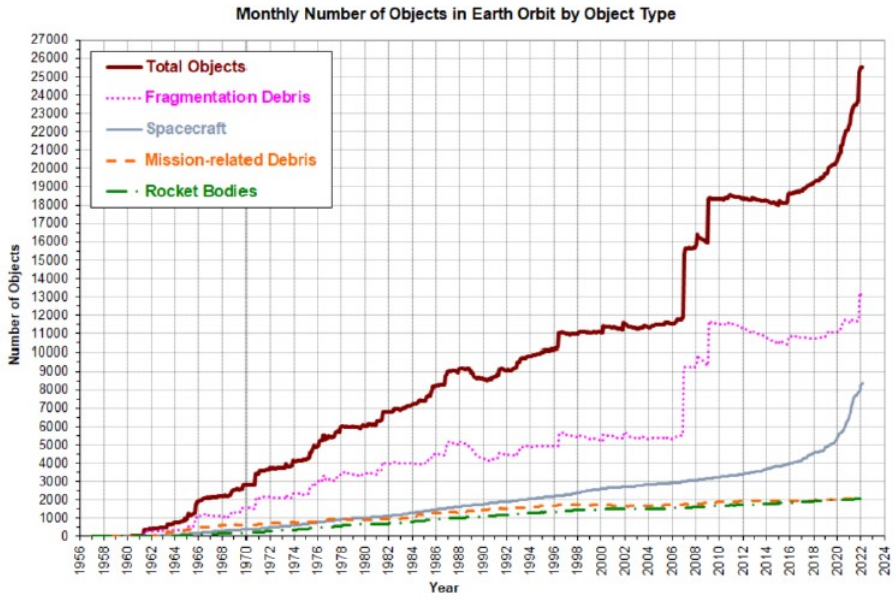


Figure 1.3: Combined number of objects in LEO, courtesy of [7]. The increase in orbiting payloads is clearly visible, as are the spikes in fragmentation debris created by ASAT tests in 2007 and 2021 and by accidental collisions in 2009.

total perforation is highly reduced with a relatively lightweight solution. Most of the development for these structures is currently made heuristically, based on extensive testing or semi-empirical equations [6]. Most of the current available data on hypervelocity impacts are obtained through gas light-gun testing campaigns, which are extremely expensive, suffer from limited availability (because of the small number of existing facilities) and usually give results related to a precise material and geometry of impactor and target. Thus, the design process would greatly benefit from having a simulation technique giving accurate results without requiring excessive computational time.
Developing such simulation technique has been the focus of many research efforts in the last 50 years since multiple factors make it very complicated. Indeed, during HVIs the energies considered create shockwaves in the material that surpass most metals' bulk moduli, which then behave like compressible fluids. This means that simply considering balance equations is not sufficient, and instead they have to be solved together with an equation of state (EOS) prescribing a pressure-volume constitutive relation. Additionally, phenomena like large geometrical deformations, fracture, contact and thermal effects all play an important part in the accuracy of the simulation. For example, with $v \sim 1\ km/s$ plastic heating, plastic flow and fracture are the dominant phenomena to consider, while for $v \geq\ 10 km/s$ phase changes, like melting and vaporization, become important in determining the outcome of the simulation.

1.1. Motivation

Multiple simulation techniques have been tested to account for these complexities, like eulerian hydrocodes, lagrangian finite element codes and meshless particle methods. None of these methods can accurately capture the coupling between stress wave propagation, fracture and contact for the velocity range of highest interest, which is $1\ km/s \leq v \leq 8\ km/s$. Indeed, a typical HVI consists in a first impact on the spacecraft outer wall, which then generates a debris cloud with lower velocities and larger spread. These secondary impacts are usually much more dangerous for the satellite survival, since they can damage electronic components and wires. Similarly, the waves created by the primary impact can be the source of heavy damage of many delicate satellite components.
These aspects are usually not well capture by traditional tests and simulation set-ups, which focus on the target penetration and on the formation of the debris cloud. Therefore, the increase of computational power experienced in the last years pushed researchers to consider the opportunity of full-scale simulation of satellite impacts for better assessing the expected damage. Thus, the scalability of simulation methods is also becoming an important feature. Eulerian hydrocodes capture well the coupling and the above mentioned features only for velocities $v \geq 6\ km/s$. If the secondary impacts

have lower velocities, their results might not be reliable. Traditional lagrangian finite elements, on the other hand, allow for meshing of complex satellite architectures but have issues modelling the damage and large deformations created by impacts with $v \geq 3\ km/s$. Moreover, they cannot model well the debris cloud and the most widespread damage modelling techniques create issues with stress waves propagation. Lastly, meshless methods are great in modelling primary impacts and the debris cloud morphology, but make extremely complicated to mesh large, complex architecture. Additionally, they do not offer directly information about size, orientation and velocity of solid fragments in the debris cloud, which are fundamental to assess the damage created by the secondary impacts.
The aim of this thesis is to propose and lay the foundations for an approach able to model the coupling between stress wave propagation, contact and fracture. For fracture and wave propagation we chose the Discontinuous Galerkin/ Cohesive Zone Method (DG/CZM) Finite Element (FE) formulation, which is a lagrangian FE technique developed for massive parallel calculation and has proven the ability to model accurately dynamic fracture and wave propagation in terminal ballistics application [13]. However, it currently lacks a contact formulation. Thus, we chose the Decomposition Contact Response (DCR) [14] for the contact formulation. This algorithm features a parallel implementation and has shown good energy conservation properties.

1.2. Document outline

This thesis document follows the following structure: the physics involved in hypervelocity impacts and the available simulation methods are first discussed in chapter 2. Subsequently the DG/CZM framework, the DCR algorithm and the their implementations' software architectures are explained in depth in chapter 3. Afterwards, in chapter 4 are discussed the results of the analysis performed on the DCR algorithm analytical formulation and on its implementation. Then, in chapter 5 the steps taken to implement the DCR algorithm in an impact simulation using the Discontinuous Galerkin (DG) FE formulation are explained. In this chapter no damage and fracture are allowed in the mesh, meaning that the efforts are directed mostly to converting the mesh architecture from Continuous Galerkin (CG) to DG discretization and to removing some of the limitations previously highlighted in chapter 4. Afterwards, in chapter 6 fracture onset and progression are activated and the modifications implemented to account for the activation of DG/CZM's cohesive elements are explained. Lastly, in chapter 7 the work done is summarized and recommendations on how to continue developing the coupling between DCR and DG/CZM are given.

1.3. Research question

Based on the literature review performed in chapter 2, it has been inferred that spacecraft shielding designers could make their design efforts quicker and cheaper if a hypervelocity impact (HVI) modelling technique able to perform full-scale satellite fragmentation simulations for impact in the $1km/s \leq v \leq 8km/s$ velocity range within one physics-based framework was available. Thus, the main research question is:

Research question (RQ)

In what manner could a numerical approach be created to simulate the fragmentation of satellites under impact conditions with velocities in the range of $1km/s \leq v \leq 8km/s$?

The development of such a tool is an ambitious, time consuming and complex effort. From the literature review the main features for such technique have been identified and their couplings defined. Indeed, the technique would need to include models for fragmentation, stress wave propagation and thermo-mechanic coupling, together with an appropriate material model, a hydrodynamic stress formulation and an appropriate equation of state. In this context, fragmentation is intended as the coupling of fracture and contact. The techniques used to model these two features usually have an heavy influence on stress wave propagation. Thus, these three aspects can be considered coupled. Similarly, the equation of state and hydrodynamic stress models are influencing each other heavily, and so do the material model used and the thermo-mechanical coupling.
Each of these three subgroups can be isolated into a separated research sub-question:

First research sub-question (RSQ1)

Which techniques can be used for modelling fracture, contact and stress wave propagation and how can they be coupled?

Second research sub-question (RSQ2)

Which technique can be used for modelling thermo-mechanical coupling in HVI and how can it be coupled with an appropriate material model?

Third research sub-question (RSQ3)

How could a hydrodynamic stress formulation be included in the HVI framework and could it be coupled with an appropriate equation of state?

The triple coupling in RSQ1 is considered the most challenging of the three, and thus we chose it as the focus of this thesis work.

2

Literature review

This chapter will offer an overview of hypervelocity impacts, of their main typical features and of the physics behind them. Then, the main numerical simulation methods utilized in literature will be discussed and their respective strengths and weaknesses will be discussed.

2.1. Hypervelocity impacts' features and physics

The more traditional metric utilized for categorization of impact events is the relative velocity of impactor and target. Impacts up to $v \sim 1km/s$ are usually considered High Velocity Impacts (HiVI), while the "pure" Hypervelocity Impact (HVI) regime is considered when $v \geq 10km/s$. All velocities in between are usually labelled as hypervelocity regime, but the material response changes noticeably between the two extremes [15]. Indeed, towards "pure" HiVI regime strength effects and thermal softening are the main features to consider, while at HVI melting, vaporization and other high energy effects become dominant and fully hydrodynamic formulations are necessary. In Signetti's work [15] a rather complete review of recent literature entries for HVI modelling in the $1km/s \leq v \leq 10km/s$ is made. The analysis focuses on thick target simulations, but some interesting inferences can still be made about thin targets. Thick and thin targets have different t/D ratio between the wall thickness t and the spherical impactor diameter D. Satellite structures usually belong in the thin targets category, as the use of thick shielding armor is not possible due to launch weight constraints. For thin targets the target deformation, shock wave propagation, wall perforation and debris cloud formation are the most relevant aspects, while for thick targets perforation does not usually happen and phenomena like spall and material flow are more relevant.
The number of publications is plotted as a function of velocity ranges and figure 2.1 makes clear that the most common range analyzed is the lower end of the hypervelocity transition regime, with $v \sim 2km/s$.

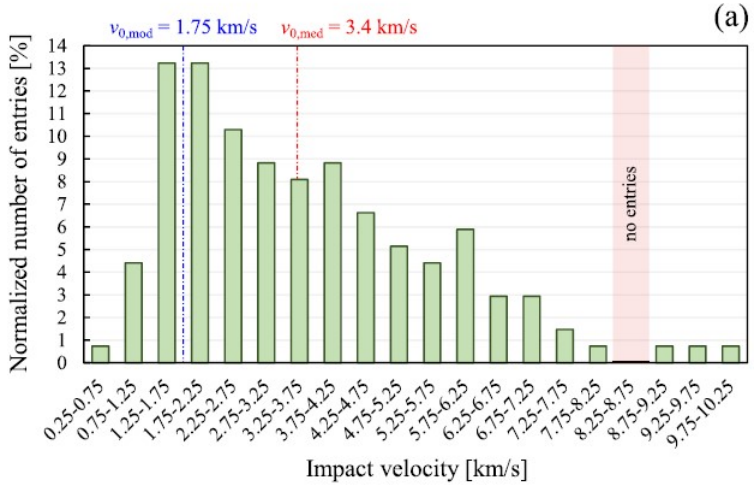


Figure 2.1: Literature entries as a function of impact relative velocity, courtesy of [15]. More than 50% of HyVI literature entries has $v \leq 5\ km/s$.This velocity range is more easily reproducible with currently available experimental setups and thus more material data is available

This traditional classification, based merely on velocity, does not account for the materials involved and their different responses. In literature multiple examples of new criteria accounting for the material characteristics can be found, like the strain rate, the "Bulk Mach number" [16], the pressure criterion [17]. All this criteria are either using quantities complex to measure in an empirical set-up (like the strain rate) or require a categorization "a posteriori", and thus have not found a wide spread application.
Given the extremely high energies involved and almost ubiquitous fragmentation of target and/or impactor [18], directly measuring outputs is extremely complicated. The most common approach for experimental characterization of HVI is thus the use of high speed cameras, witness plates and "post-mortem" analyses. Nonetheless, given that impacts in orbit are almost impossible to be directly observed, extensive experimental campaigns are the main tools to study the characteristics of HVI. The most famous, complete and extensive study available is the work from Piekutowski [18].

2.1.1. Debris cloud morphology

Piekutowski's work is intended to study the fragmentation of projectile and of a thin target. The most common set-up utilized was a spherical aluminium impactor hitting a thin aluminium plate at a normal angle. Three different alluminium alloys and 5 sheet thicknesses have been used for the targets and two different alloys and 3 diameters for the projectile. A limited amount of tests have been performed on a multiple bumper setup at a normal angle, with up to 5 bumpers in a row, and on a single bumper set-up with an oblique impact angle.
A "witness" plate is positioned at 38 cm from the target, in order to have enough time to take multiple photos of the expanding debris cloud.

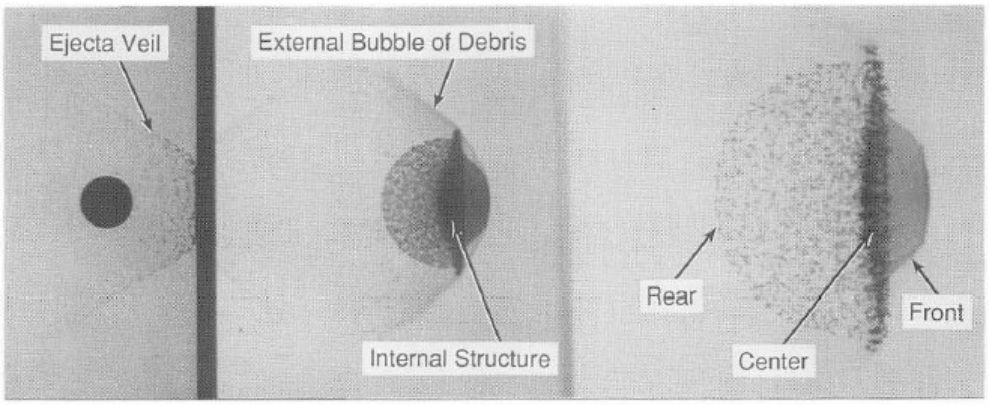


Figure 2.2: Example of debris cloud for an aluminium on aluminium impact with $v = 6.7km/s$, courtesy of [18]. Three frames are superposed in the image, showing the evolution of the debris cloud in time. On the left, the spherical impactor is shown before the impact. The, the target plate just after the impact is shown, together with the fragmented impactor and the forming debris cloud. Lastly, on the right, the completely developed debris cloud is shown with its different regions clearly distinguishable

The effect of impact velocity, scale, material and of the plate thickness to impactor diameter ratio t/D was explored in depth. Most of the tests where limited to small t/D, as this is the case for most lightweight satellite structures. This allowed to extensively analyze the debris cloud morphology and evolution, along with its velocity and the fragments composing it.
The first important feature encountered in analyzing the debris cloud is the "ejecta veil". This a stream of particles directed in the opposite direction w.r.t. the impactor velocity, created during the early stages of the impact. It is mainly composed of plate fragments and is ejected from the bumper front. In the meantime, the projectile perforates the plate. For aluminium sphere on thin aluminium sheet impacts, this happens within the first $15\mu s$ after the impact.
Then, as the projectile pierces the plate, an expanding bubble of plate debris forms on the plate's rear side. Finally, in front of the bubble an "internal structure" of projectile debris can be seen. The author separates this structure into front, center and rear region.
The internal structure changes its shape and composition depending on impactor shape, material and impact velocity. For very thin bumper plates (thickness to impactor diameter ratio $t/D < 0.2$) the central region of the impactor is composed of a solid impactor fragment. This main fragment is surrounded by many solid slivers with different shapes.
The rear region in most cases is composed of a hemispherical cloud of spalled fragments from the impactor. In ballistics, spall happens when reflected pressure waves and relaxation waves interact and create fracture at a certain distance from the impact surface. Spalled fragments are observed to decrease in size for higher velocities and t/D [18].
The front region is composed of thin, mixed fragments of both bumper and impactor. For higher impact velocities, this region is mostly made of molten or vaporized fragments. Piekutowski estimated that for the considered velocities ($v \sim 7km/s$), around 30% of the projectile initial volume should be in liquid state and around 40% in a mixed liquid-solid state.
The front and central regions represent the most dangerous features for rear wall perforation. It is notable that for $t/D > 0.424$, the front and central areas of the cloud do not exist anymore, and the distribution is more uniformly sparse in space.
For increasing impact velocity, the debris cloud increases in length and diameter, and the projectile gets increasingly fragmented. Precise description of the perforation hole morphology and dimensions as a function of multiple parameters is also given in this work. Additionally, it is observed that bumper strength influences the perforation diameter, probably mostly during the later stages' deformation when shock-induced stresses become smaller.
The main findings of Piekutowski's work are that the features of the debris cloud are heavily dependent on the material, the shape and the size of the impactor. Relatively limited emphasis was posed on the effect of non-spherical impactors and their orientation. Indeed, spherical projectiles easily allow to compare different experiments and to reduce the parameters, since variables like impactor orientation and Angle of Attack (AoA) do not have to be considered.
Nonetheless, Piekutowski performed some experiments using aluminium short rods, cylinders, disks

and a sabot insert as projectiles on zinc plates. Zinc is used because it was expected to fragment more than aluminium, allowing for better photographs of the debris cloud. The shape of the cloud and type of fragments contained in it were found to be heavily dependent on both projectile shape and orientation. No systematic study on the relation between non-spherical impactors' orientation and debris size and morphology have been performed, but on average a higher number of solid fragments is observed with non-spherical impactors [18]. A similar effect is also observed when using a damaged spherical impactor. These findings are particularly relevant, since full-scale satellite fragmentation tests like NASA's SOCIT or DebriSat have shown that most of the fragmentation debris are not shaped as spheres [19].
Directly observing and cataloguing the fragmentation debris created by ASAT tests or orbital impacts is not directly feasible, and thus impact experiments on representative satellite architectures have been performed and all the resulting debris collected and categorized. In figure 2.3 the results from DebriSat are collected as a function of the "reference length", which is an average of the three dimensions of the debris.

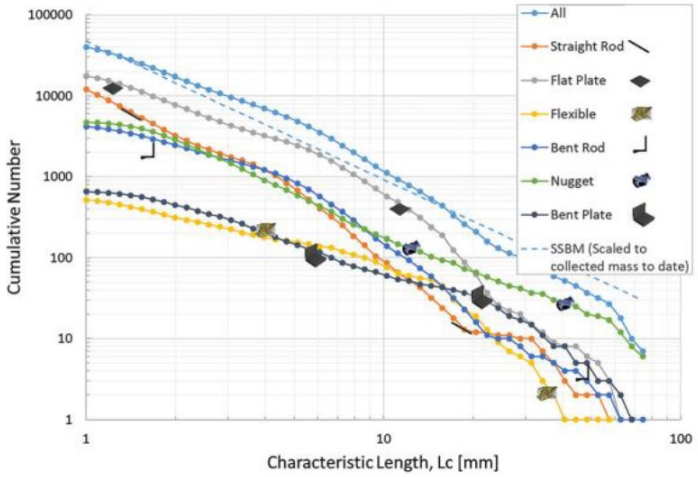


Figure 2.3: Characteristic length - number of debris relation for the DebriSat test [19]. From this plot is clearly visible that spherical shaped fragments are very unlikely to be created from satellite fragmentation. Since the most common orbital debris source in orbit is satellite fragmentation, as shown in figure 1.3, it means that is important to investigate the effect of different impactor shapes. Straight rods and small plates are the most common shapes for small debris, while nuggets are prevalent for bigger characteristic lengths

In the DebriSat case, which represents the most recent and extensive test campaign of this kind, a prevalence of small "plate-like" fragments (flakes) and elongated cylindrical fragments (rods) was observed. Additionally, the most common materials have been observed to be plastic, phenolic resin and aluminium.
Additionally, numerous works have explored the effect of different projectile shapes by numerical simulation. In [20] Cour-Palais reviews earlier studies and consider flat plate-like impactors. It was found that non-spherical impactor pose a higher penetration risk to WS because of a lower projectile fragmentation.
More systematic studies about projectile impact angle, orientation and material are found in more recent works. In [21] the authors use Smooth Particle Hydrodynamics (SPH) to model a polycarbonate cube impacting a thin aluminium plate at different angles and used experiments to validate the model. In [22] the coupled FE-SPH simulation method is used to assess the effect of sideslip angle and angle of attack on the impact of short aluminium cylinders on aluminium thin plates. Liu et Al [23], which used an SPH formulation in Autodyn to investigate the effect of aluminium conical impactors on aluminium thin targets. Overall, the orientation and material influence heavily vaporization, target perforation and overall fragmentation. The smaller the impacting area (e.g., a cube edge impacting instead of a surface) the higher the amount of solid fragments in the debris cloud.
Another relevant debris shape mentioned in [19] is the "nugget" or ellipsoidal shape. It has been previously addressed by Carrasquilla [24] who used the CTH hydrocode and a test campaign from the Ernst Mach institute to study the effect of both oblate and prolate elliptical impactors. This study showed that for $4km/s \leq v \leq 8km/s$ the BLE for traditional aluminium WS does not hold and needs to

be modified.
Lastly, an extensive analysis on the effect of various projectile shapes on honeycomb sandwich targets is performed in [25]. The effect of aluminium sphere, disc and ring shaped projectile is investigated, along with the effect of impact location w.r.t. the honeycomb cell. The paper used SPH, FE and coupled FE-SPH formulations for the simulation. Ring-shaped projectile seem to be more dangerous compared to both spheres and discs the case of "edge-on" impacts.

2.1.2. Hydrodynamic behaviour and wave physics

During impacts with $1km/s \leq v \leq 10km/s$ solid materials can behave like compressible fluids under the impulsive impact pressure loading [26]. Theoretical prediction of the behaviour of materials under HVI's extreme loading based on their atomic structure is not currently feasible, and thus most materials are characterized experimentally. Multiple curves are produced to express the relation between two or more material variables. The two most commonly utilized are between wave velocity and particle velocity $c_s - u$ or between Cauchy stress and Lagrange strain $\sigma - \varepsilon$. The latter is shown in figure 2.4. In literature, the former is most often referred to as "the Hugoniot", but this term should refer more precisely to the conservation laws expressed across the shock front. The typical expression obtained by measuring the free surface wave speed is [27]:

$$c_s = a + s_1 u.$$

Where c_s is the shock speed, a, s_1 are experimental parameters and u is the particle velocity.
This expression can then successfully be used to relate two or more variables between hydrostatic pressure p, volume V (or density), shock speed c_s and specific internal energy ⌉ using an Equation Of State (EOS) [15].
The hydrostatic pressure is one of the two components in which the stress tensor is usually divided when treating hypervelocity impacts, with the other being the deviatoric stress:

$$\sigma = -p\delta_{ij} + \sigma_{dev} = -\begin{bmatrix} p & 0 & 0 \\ 0 & p & 0 \\ 0 & 0 & p \end{bmatrix} + \begin{bmatrix} \sigma_{11}+p & \sigma_{12} & \sigma_{13} \\ \sigma_{21} & \sigma_{22}+p & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_{33}+p \end{bmatrix}.$$

Where δ_{ij} is Kronecker's delta and the hydrostatic pressure p is taken positive when compressive.

Pressure and shock wave morphology

Each pressure pulse or wave created by an impact is composed of front, middle and rear regions. The front region is often called "shock front" and is the location of the discontinuity in material properties. The middle region has slowly varying properties from the values just after the shock, while the rear region, often called "rarefaction region", is where the properties return to values close to the unperturbed state. In some cases the middle region is not present [28]. Additionally, depending on the magnitude of the shock, increases in entropy and irreversible processes in the material like solid-solid phase changes, dislocations and plastic flow can be observed [29].
Immediately after an HVI event, the material experiences a first elastic wave with speed c_E. When the instantaneous stress exceeds the elastic limit (or Hugoniot yield stress $\sigma_{hel} \sim \frac{2}{3}\sigma_y$) of the material a plastic wave arises. Its speed is $c_p = c_p(\rho, \frac{d\sigma}{d\varepsilon})$, meaning that plastic wave speed is defined by the slope of figure 2.4 for a given strain value. Thus, each impact can originate multiple plastic waves with different speeds as the strain increases.

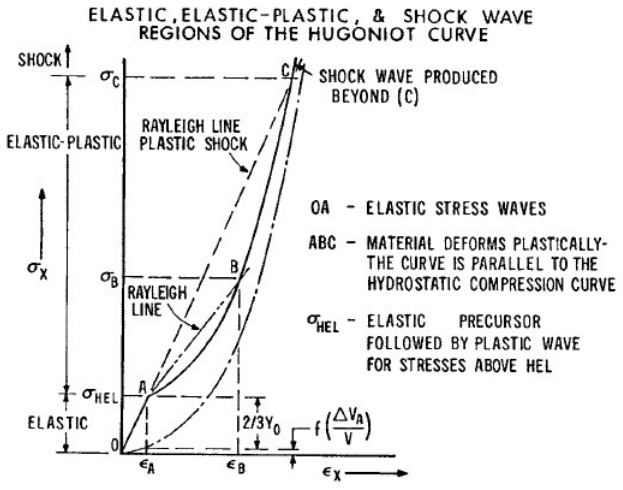
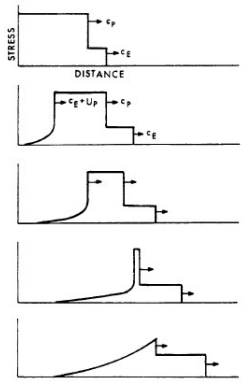
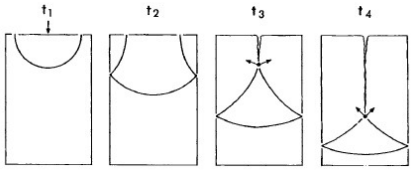


Figure 2.4: Stress-strain curve for impact events, courtesy of [26]. After an impact event, a first elastic wave is created while the stress reaches σ_{hel}. If the energy is high enough for the stress to increase this threshold, multiple plastic waves are created with increasing velocities. Indeed, plastic wave velocity is a function of the slope of the AC segment, which is clearly increasing until reaching the shock threshold σ_c. Passed this value and if the geometry allows it, plastic waves overcome elastic one and the two degenerate into a shockwave

If the stress pulse has a big enough magnitude, the plastic wave velocity increases until it overcomes the elastic waves and one steep-fronted wave is created, which moves with speed c_s (being the free surface shockwave speed used in the Hugoniot). Once that the pressure waves has a nearly vertical front, it is called a "shock wave" and creates a discontinuity in the properties between shocked and unshocked material. This makes necessary to use a hydrodynamic formulation to describe the behaviour of the material. At the end of the impulse loading created by the impact a last, elastic unloading wave appears called "rarefaction wave". Often, this wave travels faster than the plastic ones and thus can catch up from behind and attenuate their magnitude, as shown in figure 2.5a.
The attenuation phenomena is observed if no defined boundaries are present in the target/projectile geometry. Indeed, pressure waves can be reflected and refracted, which can lead to destructive or constructive interactions. When waves encounter a normal boundary w.r.t. to their speed (e.g., the end of the impacted plate) they can be reflected back in the opposite direction. When they encounter another pressure wave, the material founds itself under an instantaneous tensile loading, which can lead to "spall" fracture at a certain distance from the impact. In figure 2.5b, a slightly more complex case with three boundaries is shown as an example, with the material experiencing fracture parallel to the wave's initial propagation velocity.



(a) Attenuation of a shock wave by a rarefaction wave, courtesy of [26]. As the stress pulse loses intensity, a final elastic wave is created which travels faster than the shockwaves. If the geometry allows it, this elastic wave could overcome the shockwave and attenuate their magnitude



(b) Refraction of pressure waves and central fracture in a brittle material, courtesy of [26]. The pressure wave starts with a spherical front. As it progresses through the plate, it's reflected on the sides of the plate and starts traveling normal to its initial direction. When the two reflected front meet in the middle of the plate, the instantaneous tensile loading opens a crack

Equations of state

When the materials enter a fully hydrodynamic regime and shock waves appear, discontinuities in the system properties have to be considered. Thus, it is no more sufficient to consider only conservation of momentum, mass and energy, but also some form of the above mentioned Hugoniot curve and an equation of state (EOS). Most of the EOS are derived from semi-empirical considerations and thus have a very specific applicability range and underlying assumptions. The work from Signetti [15] is used to consider which EOS are most common in literature considering the transition hypervelocity regime.

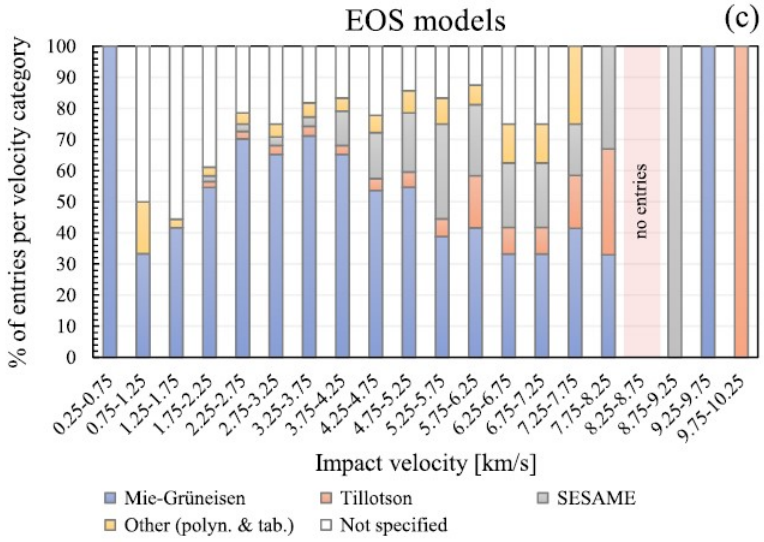


Figure 2.6: Literature entries as a function of impact relative velocity and EOS, courtesy of [15]. A clear prevalence of the Mie-Grüneisen equation is visible, explained by its ability to capture thermal effects combined with a relative simplicity. A remarkable amount of works do not specify which EOS formulation has been used. Lastly, it can also be observed how Tillotson EOS and the SESAME EOS are more used for higher velocity ranges, which is consistent with their ability to account for phase changes

The use of EOS sees a strong prevalence of the Mie-Grüneisen equation [30], which is consistent with the range of velocities analyzed by the authors. The EOS does not account for phase changes and for $v \leq 10\ km/s$ vaporization and melting are not prevalent, so the limitations of the EOS do not hinder the simulation outcomes. Polynomial, tabulated equations and the SESAME EOS are derived for specific materials and are obtained by direct measurements, so are the most precise option, especially for specific applications and alloys with an extended heritage. Their limited use is most likely related to the limited accessibility of the underlying databases. The Tillotson EOS [31] allows for increasing

complexity, compared to Mie-Grüneisen, as it accounts for phase transitions, different material states and mixed transition phases. It also requires more material data, which could prove harder to be retrieved. Combining this with the fact that with $v \leq 8\ km/s$ melted and vaporized debris account for only $\sim 10/20\%$ of the total debris mass [18], makes the use of Tillotson's EOS not too common. Signetti's works focused on the thermal effects influencing HVI simulations, and thus the exclusion of the linear EOS is probably due to its neglecting the influence of internal energy on pressure. There is proof of good results obtained with this simpler model [32].

2.1.3. Thermo-mechanical coupling and volumetric heating

Lastly, the presence of shockwaves, extremely large deformation and high strain rates leads to a strong coupling between mechanical and thermal effects, which becomes more relevant as impact velocity increases.
Two thermo-mechanical heating mechanisms: plastic heating and shock heating. The former is prevalent in the lower ranges of impact speeds of HVI, while the latter gets much more important towards the "pure" hypervelocity regime.

Plastic heating

The commonly utilized method to compute the increment in temperature from plastic heating is the so-called Taylor-Quinney approach [15], which is expressed as:

$$\Delta T = \beta_{tq} \frac{\Delta W_{pl}}{\rho c_v}.$$

Where ΔW_{pl} is the plastic work and β_{tq} the Taylor-Quinney coefficient. In HVI, is often assumed $0.9 \leq \beta_{tq} \leq 1$.

Coupled material models

Plastic heating affects material properties with phenomena like thermal softening. Moreover, it is also coupled with other large deformation non-linearities like plasticity, plastic damage and strain rate effects. In figure 2.7, an overview of the constitutive models used in literature is given.

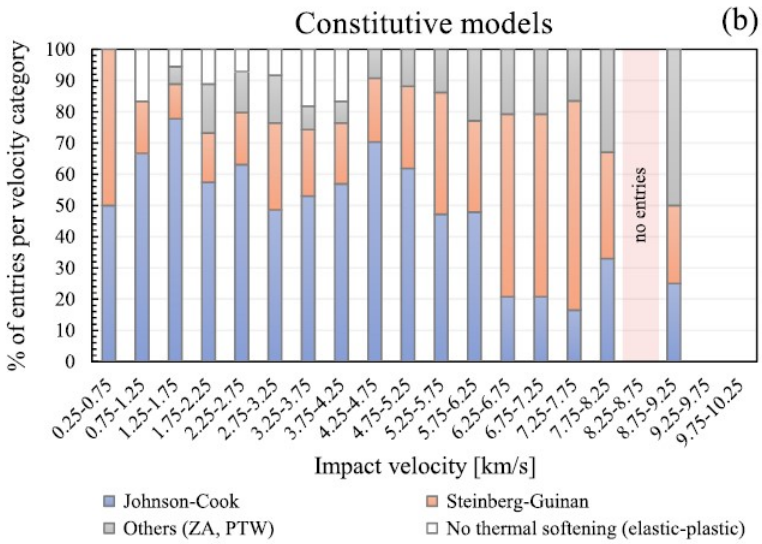


Figure 2.7: Literature entries as a function of impact relative velocity and constitutive model, courtesy of [15]. JC is clearly the prevalent option, most likely due to its easier implementation and to the inclusion of a damage model in its formulation. For lower velocity range a simple elasto-plastic material has also been used, neglecting completely the thermal effects. Finally, for higher velocity ranges SG is used more often, which is consistent with its strain rate saturation assumption

The constitutive models utilized are divided into 4 categories: Johnson-Cook (JC) [33], Steinberg-Guinan (SG) [34], elasto-plastic and all other models.

The elasto-plastic model allows for a much easier treatment of the problem, but ignores thermal softening, which could be a source of big scatter w.r.t. experimental results for higher velocities.
JC is by far the most utilized constitutive model, which also means that material data are easier to retrieve. It includes strain rate effects, while SG assumes the strain rate influence to be saturated. Steinberg places the saturation threshold for strain rate to be $\dot{\varepsilon} \sim 10^5\ 1/s$. Therefore, it is reasonable to see a more widespread use of SG for $6.25\ km/s \leq v \leq 8.25\ km/s$. The strain rate is also a function of the material properties and of the strength model, thus JC might be also preferable because it does not require to assume a strain rate value prior to the simulation. Additionally, many of the entries using SG are employing it within the CTH hydrocode, which assumes full hydrodynamic behaviour of the materials. In such cases, the assumption of saturated strain rate is not problematic.

Shock heating
Shock heating arises from a sudden pressure increase, which leads to an increase of internal energy and entropy. Given the very small time steps necessary to analyze HVI (in the order of μs) allow to assume the phenomenon adiabatic, this increase is directly related to an increase in temperature.
The common approach to compute the increase in temperature due to the superposition of plastic and shock heating is numerical [29]. For a generic step from increment i to $i+1$:

$$T_{i+1} = T_i + \frac{\mathcal{E}_{i+1}}{c_v \rho_i}.$$

Where T_i, ρ_i are respectively the temperature and the density at the start of the increment and $\mathcal{E}_{i+1}$ the internal energy at the end of the increment. The internal energy can be evaluated numerically as:

$$\mathcal{E}_{i+1} = \mathcal{E}_i + \int_V \int_{t_i}^{t_{i+1}} (\sigma_i - \mu_i)\, \dot{\varepsilon}_i dt\, dV.$$

Where μ_i is the artificial viscosity for the considered time step [26], used to avoid numerical instabilities. For $1\ km/s \leq v \leq 10\ km/s$ phase changes are not dominant: for example, in [35] impact testing on a variety of metals shows that when $v \sim 20 km/s$ phase changes are absorbing up to 20% of the system's energy, meaning that at lower velocities their effect could be disregarded.

2.2. Modelling approaches

Most physical problems can be solved either analytically or numerically, if not using experimental or semi-empirical techniques. However, for HVI obtaining a closed form analytical solution is possible only for very simple geometries and under strict assumptions, which are often not met in real applications. Thus, such engineering problems are usually solved numerically. Methods like finite elements, finite differences or finite volumes are used to discretize the problems. The following discussion will focus on the simulation techniques used to simulate HVI and their strengths and limitations.

2.2.1. Finite elements

In the lagrangian formulation, the equations of motion are formulated in the undeformed configuration. This requires a complex implementation, but trivially satisfies mass conservation of elements and allows to more easily track the deformation history, which helps when plastic deformation and damage are important towards the simulation outcome. At the same time, more issues are recurrent for highly dynamic events as HVI, like element distortion and mesh entanglement. This often leads to a rather poor representation of the debris cloud created by HVI [36].
The main advantages of FE are that numerous commercial codes are available, that they are the main simulation technique used in structural analysis and their application to ballistics is widespread. The most common utilized codes are LSDyna and Abaqus.

Element erosion
FE are rarely used in HVI simulations due the known mesh distortion issues. An example is [37], where Abaqus Explicit is utilized to model the impact of a cylindrical aluminium projectile on a thin aluminium target with $v = 0.97 km/s$ with both FE and SPH. The FE model shows poor performance compared to the meshless SPH technique. The paper showcases the traditional "element erosion" technique

implemented to account for damage progression and to avoid excessive mesh distortion. When an element is considered fully damaged or when plastic strain surpasses a certain threshold, it is simply deleted from the simulation. The authors point at this feature as one of the most probable causes for the poor performance of the FE simulation, since its non-physical deletion of mass and energy leads to a wrong deformation pattern for the projectile.
The effect of element erosion on wave propagation might also partially explain the errors of the FE result. Indeed, when using element erosion no wave can travel trough "fractured" interfaces, since the elements are simply deleted and the interfaces are no longer in contact. Additionally, no mention is made about the implementation of self-contact for the target elements, which might have partially fixed this issue. Element erosion is also heavily dependent on mesh size [36]. Indeed, the smallest fragment size is linked to the element size, as the single FE cannot be further fragmented. Similarly, cracks created through element erosion have the same size as the eroded elements.

Node separation

In an effort to obtain a debris cloud and overcome the issues of element erosion, a different method for damage progression is implemented in [38]. The author applies the node separation finite element (NSFE) technique. In traditional FE neighbouring elements share the same nodes, while in this formulation each element possesses a copy of the node. These copies are superposed and constrained together until fracture onsets, when the constraint is released and a crack is opened. This technique is implemented into the LS-Dyna commercial code and tested on the impact for an aluminium sphere on a thin aluminium plate at $v = 6km/s$ with good results.
The simulation also includes the Johnson-Cook (JC) material model and a volumetric heating model. Moreover, elements which are completely detached are not deleted but instead kept in the simulation. Additionally, the nodes of the element which melt according to JC are kept in the simulation with a constant linear motion, unable to interact with the surroundings. Element erosion is still necessary to deal with possible excessive mesh distortion and elements' self-piercing. These measures allow to retain a much higher fraction of the system energy when compared to using element erosion for damage progression.
The debris cloud shape in [38] is satisfactory when considering fragments and "melted" elements' nodes, but a heavy dependency on the mesh size is still observed. In [39] the same technique is applied to a composite Kevlar/epoxy and to a CFRP plate, subjected to the impact of a spherical aluminium projectile at respectively $v = 3km/s$ and $v = 6km/s$. When considering the eroded elements, a good representation of the debris cloud is obtained. Internal contact for opened interfaces needs to be added to account for wave propagation happening after crack closure.
In [36] NSFE is compared with various particle methods and coupled particle methods/FE formulations. In contrast with the previous two papers, NSFE seems to perform much worse than the alternatives. The two most probable cause are the mesh element size used for NSFE, which is much bigger than the that used for particle methods, and not considering melted elements as done by [38]. Lastly, the author compares different methods and different discretization sizes based on the absolute number of debris created, which is a debatable metric as particle methods are expected to create more debris, especially when smaller particle size is used.

2.2.2. Lagrangian meshless methods

The hydrodynamic behaviour of materials in HVI and difficulties encountered in modelling the debris clouds following HVI have pushed many authors to utilize meshless methods. In these formulations particles are used instead of finite elements or finite volume cells. Therefore, meshless methods are also often called particle methods. The interactions between the particles are modelled with lagrangian continuum mechanics equations' approximations and define the response of the material. Different particle methods model are distinguished by how the interaction between the particles is modelled and on how the field variables are estimated.

Smoothed particle hydrodynamics

The most common meshless method utilized to model hypervelocity impacts is Smoothed Particle Hydrodynamics (SPH) [6, 36]. This technique has been initially developed to model planetary formation

phenomena and uses probability to predict the interactions between particles. The SPH methodology is based on the use of a set of Newtonian particles, which are modelled following a continuum mechanics lagrangian formulation. Around each particle a volume is defined based on the smoothing length, a user-defined parameter. Within this area the particle interactions are modelled using statistical smoothing (from which the methodology derives its name), which allows for complex simulations to be run with acceptable computational times compared to traditional finite difference codes.
This method currently offers the best results in modelling the debris cloud shape created by the HVI [6], but still requires 5 times the simulation time when compared to similar scale simulations using FE [36].
The definition of the domain boundaries in SPH represents one of the method's main issues. The most common technique is to use "ghost nodes", which are fixed massless particles which act as a boundary for those involved in the simulation. Usually, this makes the size of the simulation grow considerably, up to the ghost particles' mass equalling that of particles actually involved in the simulation [37]. Therefore, when the total surface area is big compared to the impacted area, the use of FE is still required [6].
Another issue often encountered in SPH is tensile instability [40], which occurs when the material undergoes high impulsive tensile loading. This kind of loading is usually what causes spall fractures, due to the interaction of two compressive waves opening a crack at a finite distance from the impacted area. Tensile instability has been traditionally fixed with the use of artificial viscosity [40]. The use of numerical stabilization is also needed because some of the energy equation formulations used in SPH can hold unphysical solutions, like negative internal energy. For example, in Abaqus' formulation a predictor-corrector scheme is used to avoid these numerical issues, which adds to the computational load and further lowers the method's efficiency [37].
Lastly, with SPH only an estimate of the mass and velocity of the debris cloud can be retrieved, with no information on shape and size of the individual debris fragments. This could heavily influence an estimation of the damage caused by an impact to internal components of a spacecraft. It is possible to use methodologies like the Finite Element Reconstruction (FER) [41] to statistically estimate size and shape of the debris based on SPH outputs, but this increases complexity and again hinders the applicability of this method to full scale simulations.

Discrete element method
DEM is another particle-based method which considers the material as composed of independent, rigid spheres which interact through repulsive or cohesive potentials. It has been mostly used for granular material, but in [42] good success has been reached in modelling HVI with $v \sim 5km/s$.
This paper's formulation ignores thermal and friction effects, which then lead to an overestimation of the debris cloud dimensions and velocity. Additionally, the parameter defining the interaction potentials are not directly measured from standard material tests, but instead require a complicated, multi step convergence process. Nonetheless, this method reaches a good agreement with experimental results, requires the tuning of a limited number of parameters and uses a consistent formulation throughout the entire simulation.
Moreover, size and shape of fragments in the debris cloud can be obtained directly and interactions between fragments in the debris cloud are considered without needing additional contact routines. Lastly, tangential (frictional) and shear potential can also be added to the simulation to better simulate different kind of impacts (e.g., conical or edge-on disc-shaped impactors).
The method is available in the commercial code LSDyna and can be coupled with other formulations, like FEM [32].

2.2.3. Eulerian hydrocodes
"Hydrocodes" is the term used to group all the finite volume and finite difference codes using hydrodynamic formulation to model HVI and their immediate aftermath. This means that these codes work well for the initial instants after the impact occurs (in the range of few μs), but are not able to model accurately the rest of the system after longer time periods. Indeed, the full hydrodynamic assumption holds only until the relaxation waves arise.
The first examples of 2D eulerian finite element simulations to hypervelocity impacts are available in the literature from the early days of HVI studies [33, 43]. A 2D formulation cannot easily account for oblique impacts and orthotropic materials, limiting the complexity of the geometries to be studied [26], but allows for a simplified solving procedure.
Eulerian codes have an easier implementation when compared to lagrangian finite elements or particle

methods, since their equations are formulated in the deformed configuration. However, they need to account for mass conservation, making the introduction of free boundaries more complex, have issues with keeping track of time-history related material properties and might present mass flow numerical dissipation [26].

CTH

The most widespread and utilized eulerian hydrocode for hypervelocity impacts is CTH [44], which has been created explicitly to solve for shock-waves and large deformations in a 3D formulation. It is able to account for hypervelocity impacts, multi-material target and impactor, explosive detonations, fracture and fragmentation [43]. The solving process is composed of two steps: in the first step the lagrangian mesh cells deform and follow the material deformation, and the material properties are computed. In the second step the distorted mesh is mapped back to the initial configuration.
The code works with strict limits on the volume change of the cells and on the time step of the simulations, to make sure that pressure waves do not travel more than one cell per iteration. A finite volume, cell-centered approximation is used to solve the first lagrangian step. Velocities and stress deviators are node centered to ease the solving process, which makes more complicated to account for sliding and fracture between different materials [26]. Since the finite volume approximation utilized does not strictly meet energy conservation, artificial viscosity is used to control the mechanical energy related to volume change.
The code can account for thermodynamics and phase changes, estimating each phase fraction (solid, liquid, vapour, mixed) for each mesh cell. For multi-material cells, mechanical work is divided between materials based on mass fraction and it is then used to compute the temperature of each material based on the equation of state.
CTH has been the code of choice to model high energy impacts and to estimate the importance of phase changes on the evolution of impact generated debris clouds, as seen for example in [45]. This study focuses on "thin" targets, explicitly addressing the problem of impacts on satellite structures and the effect of pre-heating of both target and projectile. Some of the limitations of older CTH versions are visible, like the need to model impacts on a Whipple shield into two stages, one for the projectile-bumper impact and one for the debris cloud-second plate impact. More recent works like [24] have used CTH to address the effect of the impactor shape.

2.2.4. Coupled methods

In an effort to get the best of both worlds, multiple authors have tried to combine different methodologies. The two most widespread combinations are lagrangian finite elements- lagrangian particle methods and lagrangian finite elements- eulerian hydrocodes.
The results of these methodologies are often very satisfactory, but the methodologies are very complicated, very computationally expensive and often introduce a degree of arbitrariety due to the coupling function necessary to link different formulations.

Finite Element - Meshless methods

Traditional lagrangian FE method are stable, efficient and are not subjected to phenomena like tensile instability, but are not able to accurately capture the debris cloud behaviour [36]. Particle methods, instead, excel at modelling the debris clouds but show issues in accurately predict fracture morphology and complicated geometries. In an effort to combine the best aspects of the two methods, coupled FE-particle methods have been introduced in HyVI simulations. Each element is coupled with a defined amount of particles. When damage occurs and the element would be eroded in traditional FE simulation, particles are released with the same properties, velocity and acceleration of the "master" element.
Due to its popularity in HVI simulations, one of the most common particle methods used for the coupling is SPH [36]. The coupled methodology shows good results and lowers the mesh size dependency observed in SPH and NSFE [36]. The methodology is highly sensible to the type of contact chosen for particle-elements interactions, which also heavily influences the simulation outcome,. In fact, element-particle interpenetration is commonly observed [46]. Similar effect has the coupling algorithm between FE and particles [47]. Additionally, the method has a low computational efficiency and does not solve the issue related to retrieving shape and size of solid debris fragments from SPH particles.
Parallel computing is a standard feature of LSDyna, and thus can be applied in FE-SPH simulations.

Another type of coupled method is the FE-DEM algorithm, which has shown very promising results: it reached good agreement with experimental results using a linear EOS (which greatly lowers the amount of material data needed) and ignoring frictional potential for particle-particle and FE-particle contacts [32].

Lagrangian Finite Elements -Eulerian Hydrocodes
The most common example in this category is the Arbitrary Lagrangian-Eulerian (ALE) method. Usually, limited subdomains where large deformations are expected are meshed with an eulerian grid, while the rest of the domain is meshed with FE. These subdomains can be moved arbitrarily in the lagrangian global mesh used for the rest of the problem domain. The eulerian mesh rezoning areas can be set by the user or advanced remeshing algorithms can be used to optimize element number and shapes. The former option is much more efficient but there is a great risk of the user influencing excessively the simulation outcome, while the latter becomes extremely computationally expensive.
From its earlier implementations the method showed issues in handling multi-material problems [48]. In [49] the adaptive remeshing approach shows good results in many highly non-linear problems, like metal forming and cutting, but its implementation is very complex. Additionally, most ALE show issues in accounting for problems where the convective terms are dominant, like HVI [26].
These methods are not too commonly applied to HVI. Indeed, most of the available works in literature focus on problems involving fluid-structure interaction. For example, they have been successfully employed to model impact welding processes in [50]. ALEs also allow to include aerodynamic models in the simulations, which can prove very useful in modelling HVI impacts on high-speed aircrafts [51].

2.3. Conclusions from literature

A summary of strengths and limitations of the currently utilized simulation approaches is given in table 2.1 for $1km/s \leq v \leq 10km/s$. The previously mentioned simulation methods are given scores based on the same 5 categories previously utilized. "+" identifies good agreement of simulations with experimental results or overall higher performance than average, and "-" identifies the opposite. When "++" or "–" are used, it means that the methodology has respectively given very good or very poor agreement to experimental measurements in that category or has performed exceptionally well or bad compared to the average.

Table 2.1: Comparison of available simulation methods for HVI. DEM, FE-SPH and FE-DEM seem to be the most apt for HVI modelling, but none can compare with the efficiency of FE.

Method	Computational efficiency	Damage and fracture	Debris cloud	Thermomechanical coupling
LFEEE	++	-	- -	-
CTH	-	-	+	++
SPH	- -	- -	+	+
DEM	- -	+	++	-
FE-SPH	+	+	+	+
FE-DEM	+	++	-	-
NSFE	-	+	+	-
ALE	-	- -	-	++

It can be inferred that, when neglecting extremely high velocities where thermal effects and phase changes are dominating, FE and particle methods perform better than eulerian hydrocodes. Ideally, one would try to obtain a simulation method with the computational efficiency of FE, the accuracy in the volumetric distribution of the debris cloud of SPH and the ability to capture solid fragments in the cloud of DEM. Such methodology could represent a good option for velocities in the range of $1km/s \leq v \leq 6km/s$ even without the inclusion of thermal effects.
NSFE and DEM represent the most promising candidates as starting points. However, they have issues which are keeping them from being of more widespread use. The former is computationally inefficient and suffers from a heavy mesh dependency, and it is not clear if it has been implemented for parallel computations. The latter requires a complicated numerical convergence process and cannot model complicated geometries as FE does. Both of these issues limit their scalability to full-scale simulations.

Thus, one would need a novel HVI simulation framework: a good candidate would be using an expanded Dicontinuous Galerkin/Cohesive Zone Method (DG/CZM) approach.
DG/CZM is a lagrangian FE where the integration is performed by parts on each individual element, instead of on the entire domain. This introduces a discontinuous displacement field with jumps at element boundaries from the start of the simulation. The discontinuities are handled with additional inter-element boundary integrals which ensure stability to the simulation [52].
Therefore, there is no need to consider non-physical erosion routines like traditional FE. Additionally, the methodology is expressely developed for massive parallel computations, and therefore is much more computationally efficient than NSFE. Moreover, all of this is obtained within a FE framework, meaning that complicated geometries and low energy states can be accuratly modelled, unlike SPH, and the damage laws are directly inferred from material properties, avoiding the complicated convergence of DEM. Moreover, since only one consistent formulation is used, no arbitrary coupling formulation is needed as for FE-SPH or FE-DEM methods.
The methodology has been applied to high-speed impacts ($v \sim 0.5 km/s$) and to the dynamic fracture of brittle materials [13, 52]. At these velocities, material strength is dominating and for brittle materials plastic waves cannot be observed. Moreover, the energies involved are not high enough for a full fragmentation of the projectile or for a fully hydrodynamic material behaviour. Thus, multiple features would need to be added to this framework:

1. a contact algorithm to model the fragmentation of the projectile, which is the main cause ofthe typical HVI debris cloud,
2. an hydrodynamic formulation coupled with a relevant EOS for HVI, like the Mie-Grüneisen,
3. thermomechanic coupling, temperature computation based on volumetric changes and a HVI material model like Johnson-Cook's.

Implementing all of these three features represents an impervious research effort, and therefore this thesis work will be focused on feature item 1. Additionally, to allow for high computational peformance, all three of these features should be implemented with parallel computation in mind, which almost doubles the necessary features. In 2.8, a summary sketch is given on the features necessary for DG/CZM to become an HVI modelling technique.

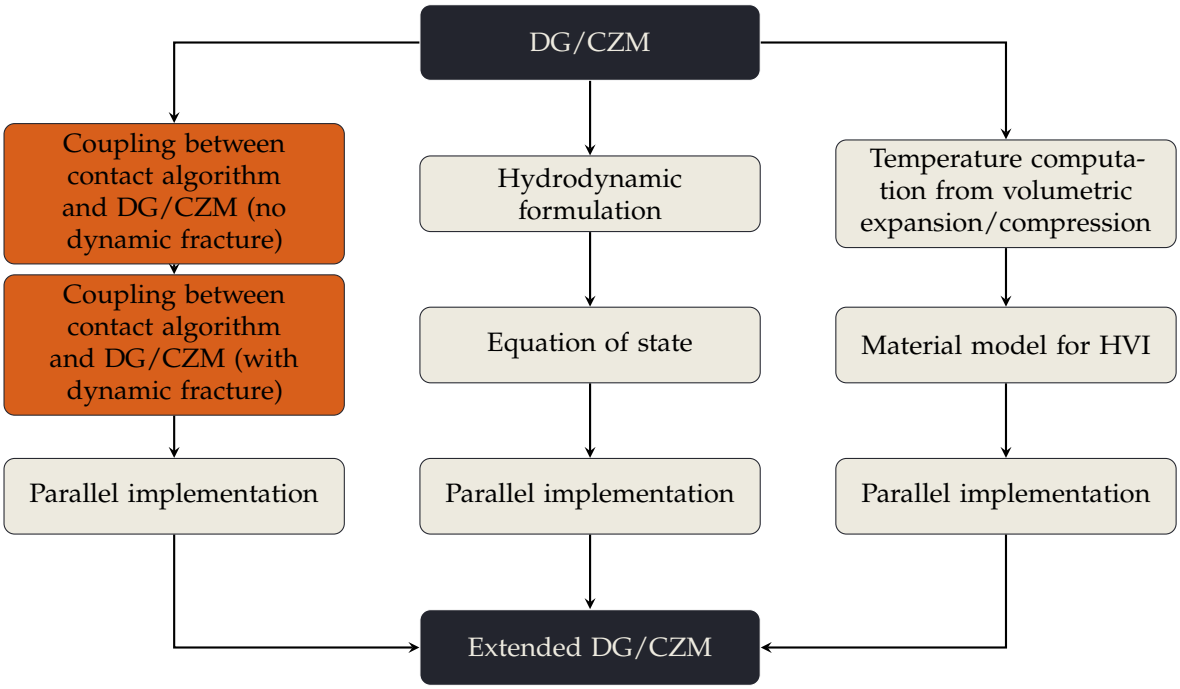


Figure 2.8: Graphical sketch to summarize the necessary steps necessary to use the DG/CZM framework for HVI modelling. In white boxes are shown all the steps to make DG/CZm a framewor able to model HVI, while in orange are shown the steps which will be taken in this thesis work. The focus has been limited to meet the time constraints of a master graduation work.

3

Background theory and tools

As established in chapter 1, the objective of this thesis is to couple the Discontinuous Galerkin/ Cohesive Zone Method (DG/CZM) formulation for solid mechanics with the Decomposition Contact Response (DCR) contact algorithm, in order to lay the foundations for a methodology able to model hypervelocity impacts using lagrangian finite elements.
This chapter provides the background information on the chosen Finite Element (FE) formulation (DG/CZM) and contact algorithm (DCR). In section 3.1, the DG/CZM methodology is explained and then the DCR algorithm is treated in section 3.2. Section 3.3 shows the layout of a Summit simulation and the software architecture of the SFC library, which implements the DCR algorithm.

3.1. The Discontinuous Galerkin/ Cohesive Zone Method (DG/CZM)

The DG/CZM method is created as a 3D framework for solid mechanics problems involving dynamic fracture and crack propagation. Usually, dynamic fracture is treated with the use of cohesive elements. Two approaches are commonly used in the literature: the first is the traditional "intrinsic" approach, which consists in inserting cohesive elements at the start of the simulation and assuming an elastic response before the opening. This creates issues with elastic wave propagation [53], which is expected to hinder HVI modelling, and is known to be a mesh dependent effect [54]. The second possible approach is the "extrinsic" one, which consists in inserting the cohesive elements at the onset of fracture at inter-element boundaries meeting a chosen fracture criterion. This is computationally demanding and has issues with compressive reloading of the cracked interfaces.
To overcome these issues, DG/CZM handles the fracture behavior without introducing morphological changes in the mesh by using of a mixed "intrinsic-extrinsic" approach to cohesive elements insertion. It inserts cohesive interfaces at the start of the simulation, but they are activated only at the inter-element boundaries where fracture onsets. Thus, before fracture onset the material behaves as a continuum and there is no need to assume an elastic response of the cohesive elements. After fracture onset, an "extrinsic" cohesive law is used to recreate the traction-separation response.
Since the whole fracture onset and crack closure reloading process is handled at the constitutive level by the DG boundary integrals, the issues in wave propagation are solved and the approach allows for a highly scalable parallel implementation [13].

3.1.1. The Discontinuous Galerkin formulation

To better explain the methodology, consider a body B_0 bounded by ∂B_0. The boundary can be split in Neumann part loaded by surface traction $\partial_N B_0$ and a Dirichlet part loaded by displacement $\partial_D B_0$. The union of the two parts gives the entire boundary and there is no intersection between the two. One can express the conservation of linear momentum as:

$$\begin{aligned} &\rho_0 \ddot{\boldsymbol{\varphi}} = \nabla_0 \cdot \mathbf{P}^{\boldsymbol{T}} + \rho_0 \mathbf{B} \text{ in } B_0 \\ &\boldsymbol{\varphi} = \overline{\boldsymbol{\varphi}} \text{ on } \partial_D B_0 \\ &\mathbf{P} \cdot \mathbf{N} = \overline{\mathbf{T}} \text{ on } \partial_N B_0. \end{aligned} \tag{3.1}$$

Where subscript $_0$ indicates initial values, $\mathbf{B}$ is a body force per unit mass, $\mathbf{T}$ the surface traction and $\overline{\boldsymbol{\varphi}}$ and $\overline{\mathbf{T}}$ the Dirichlet and Neumann boundary conditions respectively.
Consider an element-wise continuous polynomial discretization of the displacement field $\boldsymbol{\varphi}_h$ on the discretization $B_{0h} = \bigcup_{e=1}^{E} \left(\Omega_0^e \cup \partial\Omega_0^e\right)$, where Ω_0^e is the open element domain, and $\partial_I B_{0h}$ as the boundary of the discretization and $\mathbf{N}^-$ as the outward unit surface normal of a given element.
Now, the weak form of the discretized linear momentum conservation can be obtained with discontinuous trial functions $\delta\boldsymbol{\varphi}_h$. The integration is performed by parts element by element since both trial functions and domain are discontinuous, holding:

$$\begin{aligned}&\int_{B_{0h}} \left(\rho_0 \ddot{\boldsymbol{\varphi}}_h \cdot \delta\boldsymbol{\varphi}_h + \mathbf{P}_h : \nabla_0 \delta\boldsymbol{\varphi}_h\right) dV + \int_{\partial_1 B_{0h}} [\![\delta\boldsymbol{\varphi}_h \cdot \mathbf{P}_h]\!] \cdot \mathbf{N}^- dS \\ &= \int_{B_{0h}} \rho_0 \mathbf{B} \cdot \delta\boldsymbol{\varphi}_h dV + \int_{\partial_N B_{0h}} \delta\boldsymbol{\varphi}_h \cdot \overline{\mathbf{T}} dS.\end{aligned} \tag{3.2}$$

Where the jump operator is defined as $[\![\bullet]\!] = [\bullet^+ - \bullet^-]$, $\mathbf{P}_h$ the first Piola-Kirchhoff stress and the superscripts $^+$ and $^-$ represent respectively right and left element of an interface. The second term on the left side of equation equation (3.2) represent the DG boundary integral. By defining a numerical flux $\mathbf{h}\left(\mathbf{P}^+, \mathbf{P}^-, \mathbf{N}^-\right)$ and using the average operator $\langle\bullet\rangle = \frac{1}{2}\left[\bullet^+ + \bullet^-\right]$, one can find:

$$\begin{aligned}&\int_{\partial_I B_{0h}} [\![\delta\boldsymbol{\varphi}_h \cdot \mathbf{P}_h]\!] \cdot \mathbf{N}^- dS \rightarrow \int_{\partial_I B_{0h}} [\![\delta\boldsymbol{\varphi}_h]\!] \cdot \mathbf{h}\left(\mathbf{P}^+, \mathbf{P}^-, \mathbf{N}^-\right) dS \\ &\int_{\partial_I B_{0h}} [\![\delta\boldsymbol{\varphi}_h \cdot \mathbf{P}_h]\!] \cdot \mathbf{N}^- dS = \int_{\partial_I B_{0h}} [\![\delta\boldsymbol{\varphi}_h]\!] \cdot \langle\mathbf{P}_h\rangle \cdot \mathbf{N}^- dS + \int_{\partial_I B_{0h}} \left\langle\delta\boldsymbol{\varphi}_h\right\rangle \cdot [\![\mathbf{P}_h]\!] \cdot \mathbf{N}^- dS.\end{aligned} \tag{3.3}$$

Considering that the jump in $\mathbf{P}_h$ does not require to be penalized the last term in equation (3.3) can be neglected, and one obtains the following equation according to [55]:

$$\mathbf{h}\left(\mathbf{P}^+, \mathbf{P}^-, \mathbf{N}^-\right) = \langle\mathbf{P}_h\rangle \cdot \mathbf{N}^-. \tag{3.4}$$

The displacement continuity equation $\boldsymbol{\varphi}_h^- - \boldsymbol{\varphi}_h^+ = 0$ is enforced weakly to ensure numerical stability with a quadratic stabilization term in $[\![\boldsymbol{\varphi}_h]\!]$ and $[\![\delta\boldsymbol{\varphi}_h]\!]$, function of a penalty parameter β_s, of the mesh size h_s and of the Lagrangian tangent moduli $\mathbb{C} = \frac{\partial \mathbf{P}}{\partial \mathbf{F}}$. The final weak form of the DG formulation is thus:

$$\begin{aligned}&\int_{B_{0h}} \left(\rho_0 \ddot{\boldsymbol{\varphi}}_h \cdot \delta\boldsymbol{\varphi}_h + \mathbf{P}_h : \nabla_0 \delta\boldsymbol{\varphi}_h\right) dV + \int_{\partial_1 B_{0h}} [\![\delta\boldsymbol{\varphi}_h]\!] \cdot \langle\mathbf{P}_h\rangle \cdot \mathbf{N}^- dS + \\ &+ \int_{\partial_1 B_{0h}} \left\{[\![\delta\boldsymbol{\varphi}_h]\!] \otimes \mathbf{N}^- : \left\langle\frac{\beta_S}{h_S}\mathbb{C}\right\rangle : [\![\boldsymbol{\varphi}_h]\!] \otimes \mathbf{N}^-\right\} dS = \int_{B_{0h}} \rho_0 \mathbf{B} \cdot \delta\boldsymbol{\varphi}_h dV + \int_{\partial_N B_{0h}} \delta\boldsymbol{\varphi}_h \cdot \mathbf{T} dS.\end{aligned} \tag{3.5}$$

The DG formulation penalty parameter influences the simulation stable time step [56]:

$$\Delta t < \Delta t_{\text{crit}} = \frac{h_s}{\sqrt{\beta_s c}}. \tag{3.6}$$

3.1.2. Adding the Cohesive Zone formulation

The DG formulation as shown in the last paragraph is used in the simulations until the chosen fracture criterion is met for the first time. Then, the DG flux terms stop to operate and a new term based on TSL is activated. A binary parameter $\alpha \in [0, 1]$ is utilized to control the switch between DG constitutive integral and TSL.

$$\begin{aligned}&\overbrace{\int_{\partial_I B_{0h}} \alpha \mathbf{T}\left([\![\boldsymbol{\varphi}_h]\!]\right) \cdot [\![\delta\boldsymbol{\varphi}_h]\!] dS}^{\text{TSL term}} + \\ &+ \underbrace{\int_{\partial_I B_{0h}} (1-\alpha)[\![\delta\boldsymbol{\varphi}_h]\!] \cdot \langle\mathbf{P}_h\rangle \cdot \mathbf{N}^- dS + \int_{\partial_I B_{0h}} (1-\alpha)[\![\delta\boldsymbol{\varphi}_h]\!] \otimes \mathbf{N}^- : \left\langle\frac{\beta_s}{h_s}\mathbb{C}\right\rangle : [\![\boldsymbol{\varphi}_h]\!] \otimes \mathbf{N}^- dS}_{\text{DG constitutive integrals}}\end{aligned} \tag{3.7}$$

In DG/CZM the cohesive law is enforced at each quadrature points on the interface, which means that a crack can start exactly where the fracture criterion is met. An interface could also be partially open if only some of the quadrature point meet the fracture criterion. Moreover, this methodology is independent from the chosen TSL [13]. Thus, it is possible to chose the one more closely representing the behaviour of the material considered. In the case of compressive crack closure, the DG interface integral terms are activated for the normal response and the TSL term governs the tangential response, so that if the TSL includes it the frictional behaviour between the open crack interfaces could be included in the simulation.
For the scope of this thesis, a reversible linear softening cohesive law [57] has been initially chosen which assumes the cohesive behaviour to be isothermal, isotropic and to be dependent only on the displacement jump $[\![\boldsymbol{\varphi}_h]\!]$.
The two variables used to define the state of an interface are the effective separation δ and the cohesive traction per unit undeformed area $\mathcal{T}$, which are defined as:

$$\begin{aligned} \delta &= \sqrt{\gamma^2 \Delta_m^2 + \Delta_n^2}, \\ \mathcal{T} &= \mathcal{T}(\delta, \mathbf{q}). \end{aligned} \tag{3.8}$$

In this equation, γ is an input parameter used to assigned different weights to normal and tangential separation, Δ_m is the tangential separation along the local tangent $\mathbf{m}$, $\Delta_n > 0$ is the positive normal separation along the local normal $\mathbf{n}$ and $\mathbf{q}$ is a set of internal variables describing the history of the decohesion.
The TSL is activated at an interface when the following fracture criterion is met:

$$\sqrt{(\boldsymbol{\sigma}_h : [\mathbf{n} \otimes \mathbf{n}])^2 + \gamma^{-2} (\boldsymbol{\sigma}_h : [\mathbf{n} \otimes \mathbf{m}])^2} \geqslant \sigma_c. \tag{3.9}$$

After activation, the components of the traction vector $\mathbf{T}$ resisting to the opening of the crack are computed as:

$$\mathbf{T} = \frac{\mathcal{T}}{\delta} \left(\gamma^2 \Delta_m \mathbf{m} + \Delta_n \mathbf{n} \right) \tag{3.10}$$

Based on the value of T, the crack could be in a loading or an in unloading state until reaching full opening. The functional form of the law for the loading state is given by:

$$\mathcal{T}(\delta, \delta_{\max}) = \sigma_c \left(1 - \frac{\delta}{\delta_c} \right) \frac{ds}{dS} \text{ for } \dot{\delta} \geqslant 0, \delta = \delta_{\max}. \tag{3.11}$$

Where $\frac{ds}{dS}$ is the surface change from deformed to reference configuration and the critical opening δ_c is obtained from the work of separation $W_{sep} = \frac{1}{2} \sigma_c \delta_c \frac{ds}{dS} = G_c$.
The law used in the unloading state is:

$$\mathcal{T}(\delta, \delta_{\max}) = \frac{\mathcal{T}_{\max}}{\delta_{\max}} \delta \text{ for } \dot{\delta} < 0, \text{ or } \delta < \delta_{\max}. \tag{3.12}$$

Additionally, in the case of interpenetration, with $\Delta_n < 0$, the crack is considered rehealed. This means that T is set to 0 and thus the damage process is stopped and restarted from the consecutive iteration. Complete fracture happens happens with $\mathbf{T} = 0$ and $\delta \geqslant \delta_c$. Then, the TSL terms are switched off again and the DG/CZM boundary integrals are switched off for $\Delta_n \sim 0$ and $\dot{\Delta}_n < 0$ (i.e., crack closure) and thus compressive crack closure is treated with the material continuum response.

3.1.3. Space and time discretization of DG/CZM equations

The space discretization discretization of the weak formulation into finite elements is obtained using the standard shape function N_a for the deformation field, its first variation and its time derivatives:

$$\begin{aligned} \boldsymbol{\varphi}_h(\mathbf{X}) &= N_a(\mathbf{X}) \mathbf{x}_a \\ \delta\boldsymbol{\varphi}_h(\mathbf{X}) &= N_a(\mathbf{X}) \delta\mathbf{x}_a \\ \ddot{\boldsymbol{\varphi}}_h(\mathbf{X}) &= N_a(\mathbf{X}) \ddot{\mathbf{X}}_a. \end{aligned} \tag{3.13}$$

Where $a, b \in [1, n]$ is the node index for n nodes in the problem. One can combine equation (3.5) and equation (3.7) and rewrite the weak form to obtain the following set of ordinary differential equations:

$$\begin{aligned}
&M_{ab}\ddot{\mathbf{x}}_b + \boldsymbol{f}^i_a(\mathbf{x}) + \boldsymbol{f}^s_a(\mathbf{x}) = \boldsymbol{f}^e_a \\
&M_{ab}\ddot{\mathbf{x}}_b = \int_{B_{0h}} \rho_0 N_a N_b dV \ddot{\mathbf{x}}_b, \\
&\boldsymbol{f}^i_a = \int_{B_{0h}} \mathbf{P} : \nabla_0 N_a dV, \\
&\boldsymbol{f}^s_{a^\pm} = \pm \int_{\partial_1 B_{0h}} (1-\alpha)\langle \mathbf{P} \rangle \cdot \mathbf{N}^- N^S_a dS \\
&\qquad \pm \int_{\partial_l B_{0h}} (1-\alpha) \left[\left\langle \frac{\beta_s}{h_s} \mathbb{C} \right\rangle : [\![\mathbf{x}_b]\!] \otimes \mathbf{N}^- \right] \cdot \mathbf{N}^- N^S_a N^S_b dS \\
&\qquad \pm \int_{\partial_1 B_{0h}} \alpha \mathbf{T}([\![\mathbf{x}]\!]) N^S_a dS \\
&\boldsymbol{f}^e_a = \int_{B_{0h}} \rho_0 \mathbf{B} N_a dV + \int_{\partial_N B0h} \overline{\mathbf{T}} N^S_a dS.
\end{aligned} \tag{3.14}$$

Where M_{ab} is the mass matrix, $\pm$ is used to indicate the boundaries of two elements with the same interface, and $\boldsymbol{f}^i_a, \boldsymbol{f}^s_{a^\pm}, \boldsymbol{f}^e_a$ are respectively internal, interface and external forces. The time integration is performed using a typical Newmark integration scheme [58] for explicit dynamics:

$$\begin{aligned}
\dot{\mathbf{u}}_{j+1} &= \dot{\mathbf{u}}_j + (1-\gamma_{NM})\Delta t \ddot{\mathbf{u}}_j + \gamma_{NM}\Delta t \ddot{\mathbf{u}}_{j+1} \\
\mathbf{u}_{j+1} &= \mathbf{u}_j + \Delta t \dot{\mathbf{u}}_j + \frac{\Delta t^2}{2}\ddot{\mathbf{u}}_j \\
M\ddot{\mathbf{u}}_{j+1} &= \mathbf{f}^e_{j+1} - \mathbf{f}^i_{j+1} - \mathbf{f}^s_{j+1}.
\end{aligned} \tag{3.15}$$

Where the Newmark parameters are $\gamma_{NM} = 0.5$ and $\beta_{NM} = 0$. u_i is the displacement at time t_i, M is the mass matrix and $\Delta t = f_t \cdot t_{stable}$ is the time step. $f_t \in [0, 1]$ is the time factor and t_{stable} the stable time step from the contact solver.

3.2. The Decomposition Contact Response Algorithm (DCR)

The Decomposition Contact Response (DCR) method is an explicit contact algorithm for FE discretized solids with smooth and non-smooth geometries [14]. It allows for shorter computation times when compared to augmented lagrangian contact algorithms [14] and it does not need to be calibrated numerically as the penalty stiffness methods. Moreover, it computes post-impact velocities with a closed form solution obtained from momentum decomposition, which means that no iterative procedures (e.g., like the Newton-Rapson method) are needed to solve the geometrical constraint equation. Additionally, the use of only local quantities for the solution of each impact event looks promising in the perspective of a future parallel implementation.

For a general solid mechanics problem, the standard time discretization technique is to divide the total time interval t_{tot} in a determined set of n_t sub-intervals, with this number changing based on the chosen space discretization and FE formulation. In the case of a DG/CZM mesh, for example, the total number of sub-intervals is determined based on the stable DG time step shown in equation (3.6). The computational cost thus scales linearly as $C_I \propto n_t$, given that one numerical integration is needed for each step in time.

The introduction of contact can greatly increase the computational cost. Indeed, assuming to solve each contact iteration at the its exact time of impact, the numerical integration would need to be stopped for each impact. Thus, to each time integration step n_i in the contact-less mesh would correspond a set of sub-steps directly proportional to the time step's number of impacts C_i, leading to $C_I \propto \sum_{i=0}^{n_t} C_i$. This corresponds to at least an order of magnitude increase in the computational cost for a highly chaotic system like that expected in the debris cloud following a HVI.

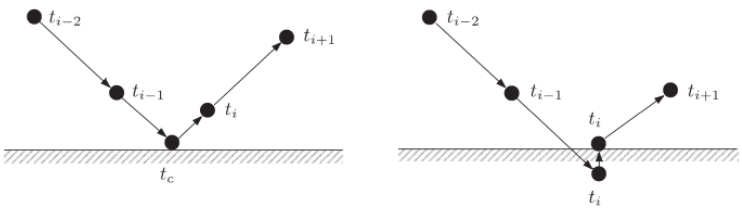


Figure 3.1: A graphical representation (courtesy of Cirak's work on the DCR algorithm [14]) showing the difference between solving for contact at the exact time (on the left) and using the DCR approach (on the right). The solid mechanics system is solved ignoring contact constraints and the node collides with the surface at t_i. Then, the intersection is removed and used to compute the velocity after impact without advancing the simulation time. Lastly, a new solid mechanics step is started to get to t_{i+1} with the updated velocity.

To tackle this issue, the main underlying assumption of the DCR method is that all given $C = \sum_{i=0}^{n_t} C_i$ impact events in a simulation time step happen at the same time, as shown in figure 3.1. The time integration step is assumed small enough for all the contact events to happen simultaneously at a given time $\bar{t}_c$. This assumption is likely to hold throughout the simulation, since the DG/CZM time step is reduced by a factor $\frac{1}{\sqrt{\beta}}$, as shown in equation (3.6). This allows to use the predictor-corrector scheme shown in figure 3.2 and to solve the contact interactions once per time step, greatly reducing the computational cost C_I.

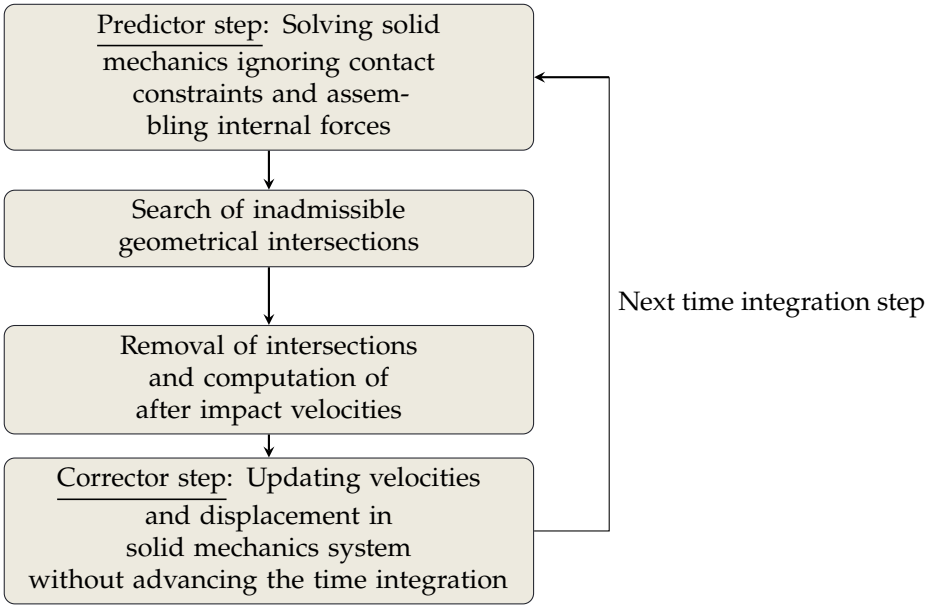


Figure 3.2: Flow chart illustrating the contact algorithm for one simulation time step. The solid mechanics is solved once in a predictor step, that ignores the contact constraints. Then, the inadmissible interactions are found and evaluated, the post-impact velocities computed, and finally the system velocities are updated and the forces recomputed considering the post-impact velocities in a corrector step.

3.2.1. Closed form solution of momentum conservation equations

Solving the predictor step and ignoring contact constraints will lead to some inadmissible geometrical intersections between the the mesh elements, which are quantified by a constraint function $g(\mathbf{x})$. This function is defined by the admissible deformation set:

$$A = \{\mathbf{X} \in Q \mid g(\mathbf{X}) \leq 0\}\,.$$

It depends only on the nodal positions $\mathbf{X}$ and gives contact without any penetration when $g = 0$. The gradient of the constraint function $\nabla g(\mathbf{X})$ defines the normal direction to the contact boundary ∂A. The DCR algorithm can be applied for a generic $g(\mathbf{X})$. The specific expression of the constraint function is a modelling choice, which will be discussed in the following.

Under the above mentioned assumptions, one can obtain the following quadratic equation to compute the post-impact velocities from the system's action integral at the equilibrium configuration by using variational calculus [14]:

$$(\lambda \nabla g + \mathbf{p}_i)^T M^{-1} (\lambda \nabla g + \mathbf{p}_i) - \mathbf{p}_i^T M^{-1} \mathbf{p}_i = 0$$

Where $\mathbf{p} = M\dot{\mathbf{u}}$ is the momentum just before the impact, M is the mass matrix, $\dot{\mathbf{u}}$ is the nodal velocity field, and λ is a scalar parameter. This equation can be solved with an iterative procedure (e.g., the Newton-Rapson method) or by using a closed form solution obtained from momentum decomposition. The DCR methodology uses the latter approach.
Considering contact with sliding, the momentum vector is decomposed as follows:

$$\mathbf{p} = \underbrace{\mathbf{p}_{slide} + \mathbf{p}_{fix}}_{\text{Tangential component}} + \mathbf{p}_{norm}$$

Where only $\mathbf{p}_{slide}$ gives rise to a relative motion of the two surfaces involved in contact.

3.2.2. Triangle-Triangle collisions

An implementation of the DCR algorithm is available in the SFC library. This library has been developed explicitly for tetrahedral meshes, a 3D dimensional space and Continuous Galerkin (CG) finite element formulation. Given that only the outer surface of the finite elements is involved in the impacts, the contact mesh elements are 2D triangles. Two main cases are considered for triangle to triangle collisions: node-to-face collision and edge-to-edge. In both cases, 4 nodes are involved in the contact interaction. However, they are subdivided between leader and follower triangle differently.
In the node-to-face case, three nodes define the impacted triangle surface and the fourth is the impacting follower node. In the edge-to-edge node, two nodes belong to one of the edges of the leader surface and two belong to the follower impacting edge. The constraint function is defined for both cases using only local quantities as the signed volume of the tetrahedron formed by the colliding node and the leader face's vertices (for node-face penetrations) or by the two edges' nodes (for edge-edge penetrations).
In figure 3.3, the two cases are illustrated with their constraint functions.

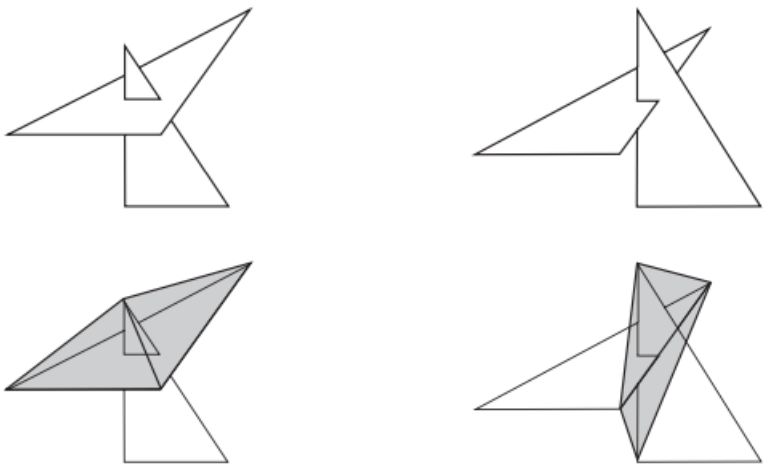


Figure 3.3: The two possible triangle to triangle collisions, courtesy of [14]. On the top left is point to face collisions, and on the bottom left the constraint function is illustrated as the volume of the tetrahedron built using the triangle and the colliding node as vertices. On the top right is shown an edge to edge collision, and on the bottom right the constraint function is illustrated as the volume of the tetrahedron built from the two segments' nodes.

The constraint function $g(\mathbf{X})$ is in both cases is expressed as:

$$g(\mathbf{X}) = \frac{(\mathbf{X}_d - \mathbf{X}_a) \cdot [(\mathbf{X}_b - \mathbf{X}_a) \times (\mathbf{X}_c - \mathbf{X}_a)]}{6}.$$

And its gradient:

$$\nabla g(\mathbf{X}) = \begin{bmatrix} \nabla_{\mathbf{X}_a} g(\mathbf{X}) \\ \nabla_{\mathbf{X}_b} g(\mathbf{X}) \\ \nabla_{\mathbf{X}_c} g(\mathbf{X}) \\ \nabla_{\mathbf{X}_d} g(\mathbf{X}) \end{bmatrix} = \frac{1}{6} \begin{bmatrix} (\mathbf{X}_d - \mathbf{X}_b) \times (\mathbf{X}_c - \mathbf{X}_b) \\ (\mathbf{X}_c - \mathbf{X}_a) \times (\mathbf{X}_d - \mathbf{X}_a) \\ (\mathbf{X}_d - \mathbf{X}_a) \times (\mathbf{X}_b - \mathbf{X}_a) \\ (\mathbf{X}_b - \mathbf{X}_a) \times (\mathbf{X}_c - \mathbf{X}_a) \end{bmatrix}.$$

Where a, b, c are the leader's triangle vertices and d the impacting vertex for the node-to-face case, while a, d are the leader's edge vertices and b, c the follower's vertices in the edge-to-edge case.

3.2.3. Node-to-face impact evaluation and constraint function computation

Starting from node-to-face contact, in figure 3.4 the approach to compute $\mathbf{g}$ is shown. $\mathbf{u}, \mathbf{v}$ are the side vectors of the leader triangle of vertices a, b, c and $\mathbf{n} = \frac{\mathbf{u}\times\mathbf{v}}{\|\mathbf{u}\times\mathbf{v}\|}$ is its outward unit surface normal vector. The follower vertex d has velocity $\dot{\mathbf{u}}_d$.

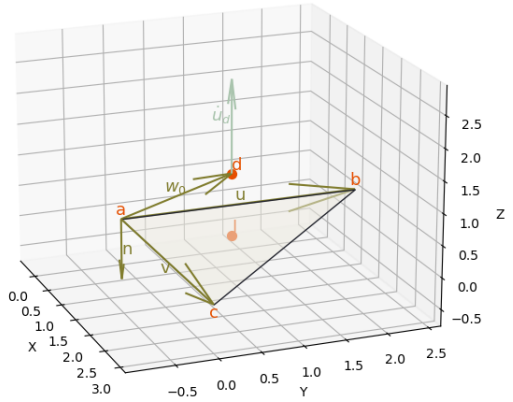


Figure 3.4: Sketch showing the geometry used in computing the gap vector $\underline{g}$ between the triangle $\overset{\triangle}{abc}$ and the follower vertex d.

Vector $\mathbf{w}_0$ describes the distance between a and d. Assuming d to impact the leader triangle, one can estimate the time to impact as $t_{imp} = \frac{\mathbf{n}\cdot\mathbf{w}_0}{\mathbf{n}\cdot\dot{\mathbf{u}}_d}$ and compute the projected position of d as:

$$\mathbf{X}_I = \mathbf{X}_d - t_{imp}\dot{\mathbf{u}}_d$$

The gap vector $\mathbf{g}$ can finally be computed as:

$$\mathbf{g} = \mathbf{X}_d - \mathbf{X}_I$$

Once that the gap vector $\mathbf{g}$ is computed, the constraint function is numerically evaluated as:

$$g = \mathbf{g} \cdot \mathbf{n}$$

3.2.4. Edge-to-edge impact evaluation and constraint function computation

To explain how the gap vector for edge-to-edge triangle on triangle impacts are currently treated in the SFC, consider the case in figure 3.5, where the interaction between a leader triangle $\overset{\triangle}{abc}$ and a follower edge defined by vertices s_0, s_1 is illustrated.

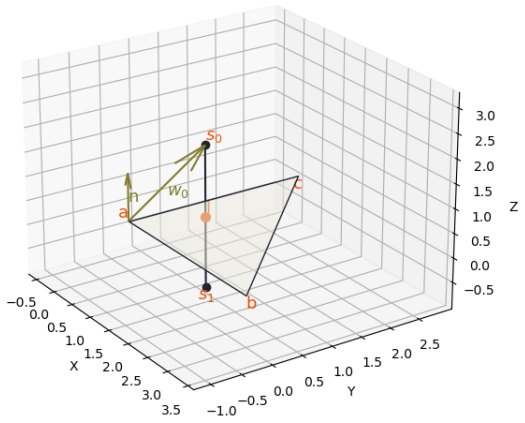


Figure 3.5: Sketch showing the geometry use to find the intersection point **I** between the leader triangle $\overset{\triangle}{abc}$ and the follower edge defined by s_0, s_1. The unit surface normal vector of the master triangle **n** and the position vector w_0 of impacting vertex s_0 are shown as well.

Vectors $\mathbf{u}, \mathbf{v}, \mathbf{n}$ are defined as in figure 3.4, while for edge-to-edge $\mathbf{w}_0 = \mathbf{X}_{s_0} - \mathbf{X}_a$ and $\mathbf{d} = \mathbf{X}_{s_1} - \mathbf{X}_{s_0}$. By defining $d_1 = \mathbf{n} \cdot \mathbf{w}_0$, $d_2 = \mathbf{n} \cdot \mathbf{d}$, one can assess if the follower edge intersects the leading triangle. Depending only on the coordinates of a, b, c, s_0 and s_1:

- $d_2 \sim 0$: the edge is parallel to leader triangle's plane.
- $(d_2 \sim 0) \bigwedge (d_1 \sim 0)$: the edge lies entirely on the leader triangle's plane.
- $d_1, d_2 > 0$: s_0 is not on the leader's triangle surface and the edge is not intersecting it.
- $(d_1 > 0) \bigwedge (d_2 < 0) \bigwedge (d_1 > -d_2)$: s_0 is not on the leader's triangle surface and the follower edge is shorter than distance between a and d_0, meaning that the follower edge is not intersecting the leader triangle.
- $(d_1 > 0) \bigwedge (d_2 < 0) \bigwedge \overline{(d_1 > -d_2)}$: s_0 is not on the leader's triangle surface and the follower edge is longer than distance between a and d_0, meaning that the follower edge is intersecting the leader triangle.

Their intersection point is computed as $\mathbf{X}_I = \mathbf{X}_{s_0} - \frac{d_1}{d_2}\mathbf{d}$.
At this point, it needs to be assessed with which of the leader triangle's edges $s_0 s_1$ has impacted. The leader edge is found by projecting the intersection point I on each of the leader triangle's edges and then selecting the edge at the minimum normal distance g_i from I, as shown in figure 3.6.

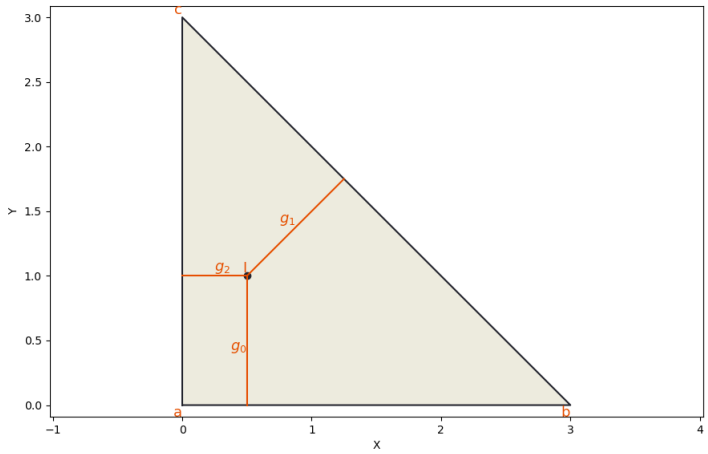


Figure 3.6: Sketch to show the geometry in computing the gap vector **g** for an edge to edge contact case. In this case, g_2 is the shortest normal distance and therefore the leader's edge is going to be ac. Not that the velocities are not considered in this entire calculation process, and thus only the coordinates **X** influence the selection of the leader edge and the calculation of the gap vector.

The gap vector is finally defined as:

$$\mathbf{g} = \{\min(\mathbf{X}_I - \mathbf{X}_{P_i}), i \in [0, 1, 2]\}$$

Evaluate the momentum components
After computing the gap vector $\mathbf{g}$ and/or the constraint function g, is necessary to evaluate the components of the pre-impact momentum and compute the post impact momentum. The general equation for the post-impact momentum is:

$$\mathbf{p}^+ = \mathbf{p}^- + \mathbf{I}^+_{slide} + \mathbf{I}^+_{norm}$$

For both cases, the normal component I^+_{norm} of the pre-impact momentum is considered, while the tangential component I^+_{slide} is evaluated only for the node-to-face impacts.
Using the subscript $^+$ to identify the quantities at the instant just after the impact and $^-$ the ones just before, the normal component of the momentum is:

$$\mathbf{I}^+_{norm} = -(1+e)\mathbf{p}^-_{norm} = -(1+e)\frac{(\nabla g)^T \mathbf{M}^{-1}\mathbf{p}^-}{(\nabla g)^T \mathbf{M}^{-1}(\nabla g)}\nabla g \tag{3.16}$$

Where e is a restitution coefficient $e \in [0, 1]$ which allows to model both elastic and inelastic material response. $e = 0$ would give a fully inelastic and $e = 1$ a fully elastic response.
The normal response is thus obtained by projecting the pre-impact momentum on the gradient of the constraint function.
The tangential component is evaluated by further decomposing it in fixed and non-fixed components. The non-fixed component is the one giving rise to relative movement between the impactor and the impacted surface and can be computed as:

$$\mathbf{p}^-_{non-fixed} = \frac{(\nabla \mathbf{h})^T \mathbf{M}^{-1}\mathbf{p}^-}{(\nabla \mathbf{h})\mathbf{M}^{-1}(\nabla \mathbf{h})^T}(\nabla \mathbf{h})^T \tag{3.17}$$

Where $\mathbf{h}(\mathbf{X}) = \mathbf{X}_I - \mathbf{X}_d$ is the separation vector between the impacting vector d. One can express $\mathbf{X}_I = (1 - \xi - \eta)\mathbf{X}_a + \xi\mathbf{X}_b + \eta\mathbf{X}_c$ as a function of the triangle coordinates and two scalar parameters ξ, η which can be obtained from $\mathbf{h}(\mathbf{X}) = 0$. Then, the jacobian of the separation vector $\nabla \mathbf{h}$, of dimensions 3×12 is obtained as:

$$\nabla \mathbf{h} = \begin{cases} (1 - \xi - \eta) & \text{if } (i, j = 1, 1) \bigvee (i, j = 2, 2) \bigvee (i, j = 3, 3) \\ \xi & \text{if } (i, j = 1, 4) \bigvee (i, j = 2, 5) \bigvee (i, j = 3, 6) \\ \eta & \text{if } (i, j = 1, 7) \bigvee (i, j = 2, 8) \bigvee (i, j = 3, 9) \\ -1 & \text{if } (i, j = 1, 10) \bigvee (i, j = 2, 11) \bigvee (i, j = 3, 12) \\ 0 & \text{else} \end{cases}$$

Then, the sliding component can be computed as:

$$\mathbf{p}^-_{slide} = \mathbf{p}^-_{non-fixed} - \mathbf{p}^-_{norm} \tag{3.18}$$

Using Coulomb's law for friction [59], the sliding traction depends on a coefficient of friction μ and gives rise to a stick-response. The sliding impulse can thus be expressed as:

$$\mathbf{I}_{slide} = \begin{cases} -\mathbf{p}^-_{slide} & \text{if } \left\|\mathbf{p}^-_{slide}\right\|_{M^{-1}} \leq \mu \left\|\mathbf{p}^-_{norm}\right\|_{M^{-1}} \\ -\mu \frac{\|\mathbf{p}^-_{norm}\|_{M^{-1}}}{\left\|\mathbf{p}^-_{slide}\right\|_{M^{-1}}}\mathbf{p}^-_{slide} & \text{in other cases} \end{cases} \tag{3.19}$$

Where $\|\mathbf{p}\|_{M^{-1}} = \mathbf{p}^T M^{-1}\mathbf{p}$.

3.3. Implementation of DG/CZM (Summit software) and of DCR (SFC library)

The Summit software is FE software developed in C++ with the scope of modelling materials in extremely large deformations regime and extreme conditions [60, 61]. It is used in B. Giovanardi's research group as the standard engineering simulation tool. The software is organized with an object oriented architecture.
Object-oriented programming (OOP) is a programming paradigm based on objects, which can contain data (also called attributes) and procedures (also known as methods). It allows to use many useful concepts, such as:

- composition: objects can contain other objects in their instance variables
- inheritance: classes can be organized in a hierarchical way, and "lower-level" classes cannot access object of "higher-level" classes
- overloading: multiple copies of the same object with different inputs can exist and perform different tasks

3.3.1. Structure of a Summit contact simulation with Continuous Galerkin formulation

The OOP architecture of Summit means that most of the already implemented methods can be reutilized in scopes different from their original application, and that many different functions have already been implemented. The procedure for a Summit contact simulation utilizing a CG finite element discretization is the already available in the Summit library and is summarized in 3.7:

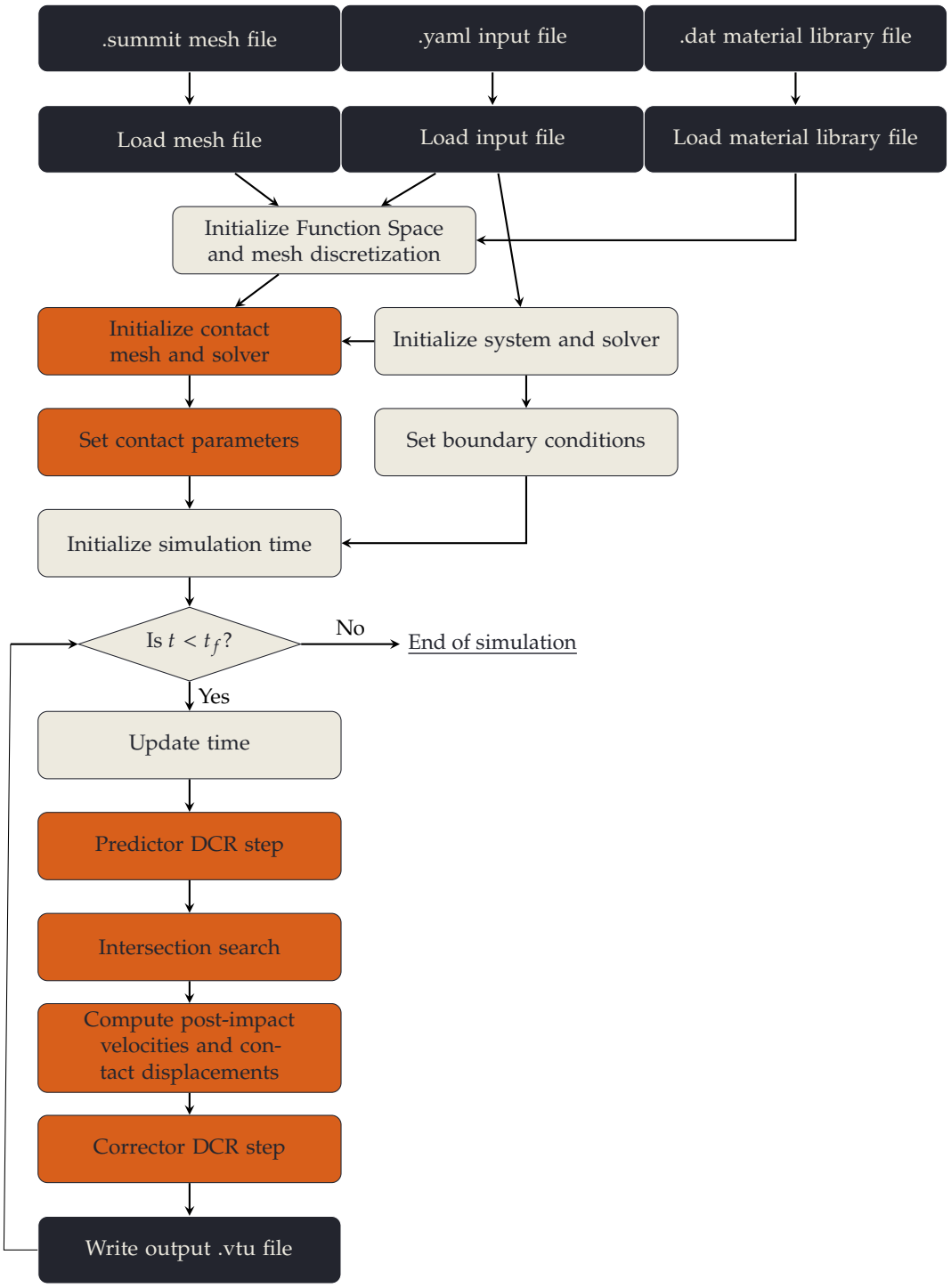


Figure 3.7: Flow chart illustrating the structure of a Summit contact simulation. The dark blue boxes represent input/output blocks, the white boxes Summit methods, and in orange boxes DCR methods.

Three input files are needed to start a simulation:

- a mesh file containing all the information about the mesh and the FE formulation and discretization,
- an input file containing a set of parameters (e.g., other input files, DCR restitution and friction factor e, μ, time factor f_t, etc.) which should not require compiling of the simulation file to be changed,
- a material library with all the material data and properties to be used in the simulation.

Once that the inputs have been loaded, the *FunctionSpace* (FS) object is initialized. It contains all the information about the discretization type, the order and type of elements and, depending on the chosen formulation type, generates the superposed DG nodes from the geometrical vertices found in the mesh file and the interface elements.
Following the FS, the *System* object is created. It defines the constitutive equation (or *WeakForm*) and computes the integrals, either boundary or volume ones. The initial boundary conditions are set directly in the simulation file (e.g., by imposing initial velocities or constraining certain degrees of freedom (DOF)).
Then, the DCR methods are called. The DCR algorithm is implemented with a legacy code library called SFC. A Summit class called *ContactSurface* (CS) is implemented as the coupling between Summit and SFC. This object is initialized and the contact parameters (i.e., the restitution and friction coefficients e, μ) are set. The contact mesh is also instantiated at this point.
Once the problem is completely set, the time variables are created and the solving loop starts. Until the final time t_{tot} is reached, the solving loop repeats the following sequence of steps:

1. Time step update: the stable time step is computed in the *Solver* and is set as the integration Δt.
2. DCR predictor step: the solid mechanics system is advanced in time ignoring the contact constraints.
3. Intersection search: the time loop is paused, the SFC search structures are populated, and the inadmissible geometrical intersections are found.
4. DCR: calculations: the DCR formulation is used to compute post-impact velocities and the intersection are removed, obtaining contact displacements and velocities $u_c, \dot{u}_c$. These are applied to the solid mechanics system nodes.
5. DCR corrector step: the solid mechanics system is advanced in time considering the $u_c, \dot{u}_c$.
6. output: the output file for the time step is written and saved.

3.3.2. Coupling between Summit and SFC

Summit and the SFC library are born as two stand-alone OOP objects, and thus information is moved around following specific paths.
The *ContactSurface* object must be called in any Summit simulation involving DCR calculations, and is developed explicitly to link Summit and SFC. This object is the only path for information to travel between Summit and SFC, as shown in figure 3.8.

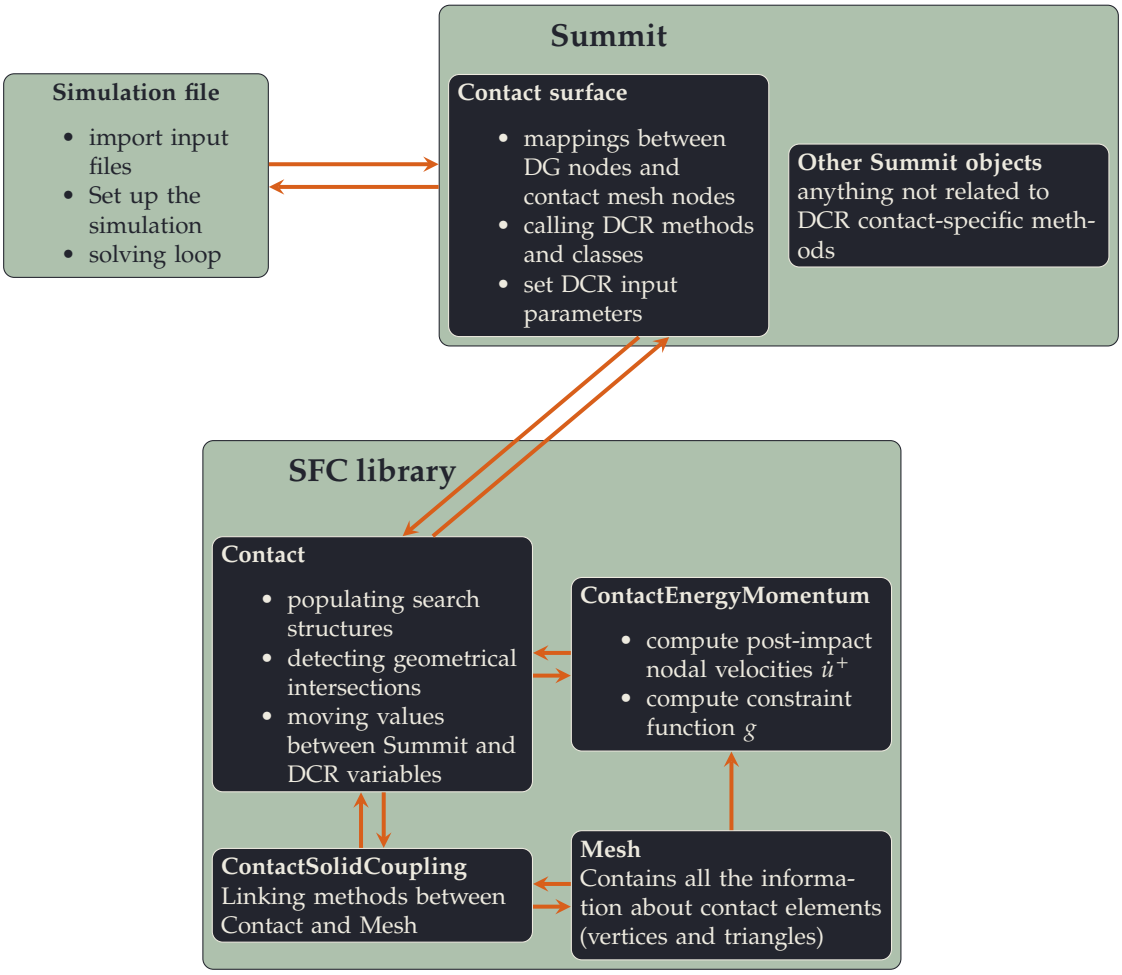


Figure 3.8: Flow chart illustrating the exchange of information between Summit and the DCR libraries. Note how the only coupling between Summit and SFC is the information flow between the*ContactSurface* and *Contact* class instances. It is also relevant to note that the `Mesh` object contains only vertices and triangles. It does not store any information about the tetrahedra composing the CG mesh.

Specifically, it is linked only with the *Contact* class of SFC. This class represent the main cornerstone of SFC: it contains the methods for the geometrical search structures and all the methods dedicated to move and copy information. Then, the second fundamental object is *Mesh*, which contains all the information about nodes and elements involved in the contact simulation. Information can flow in both direction between *Contact* and *Mesh*, but the two objects are never directly interacting. The link between the two is a dedicated *ContactSolidCoupling* class. Lastly, the *ContactEnergyMomentum* (CEM) class contains all the methods and data necessary to apply the DCR method and evaluate the constraint functions, compute post-impact velocity and displacement. It receives information from both *Mesh* (e.g., position and velocity of the contact nodes) and from *Contact* (e.g., the geometrical intersections), but only passes information back to *Contact*.
These classes are further subdivided into methods performing each specific tasks, as shown in figure 3.9.

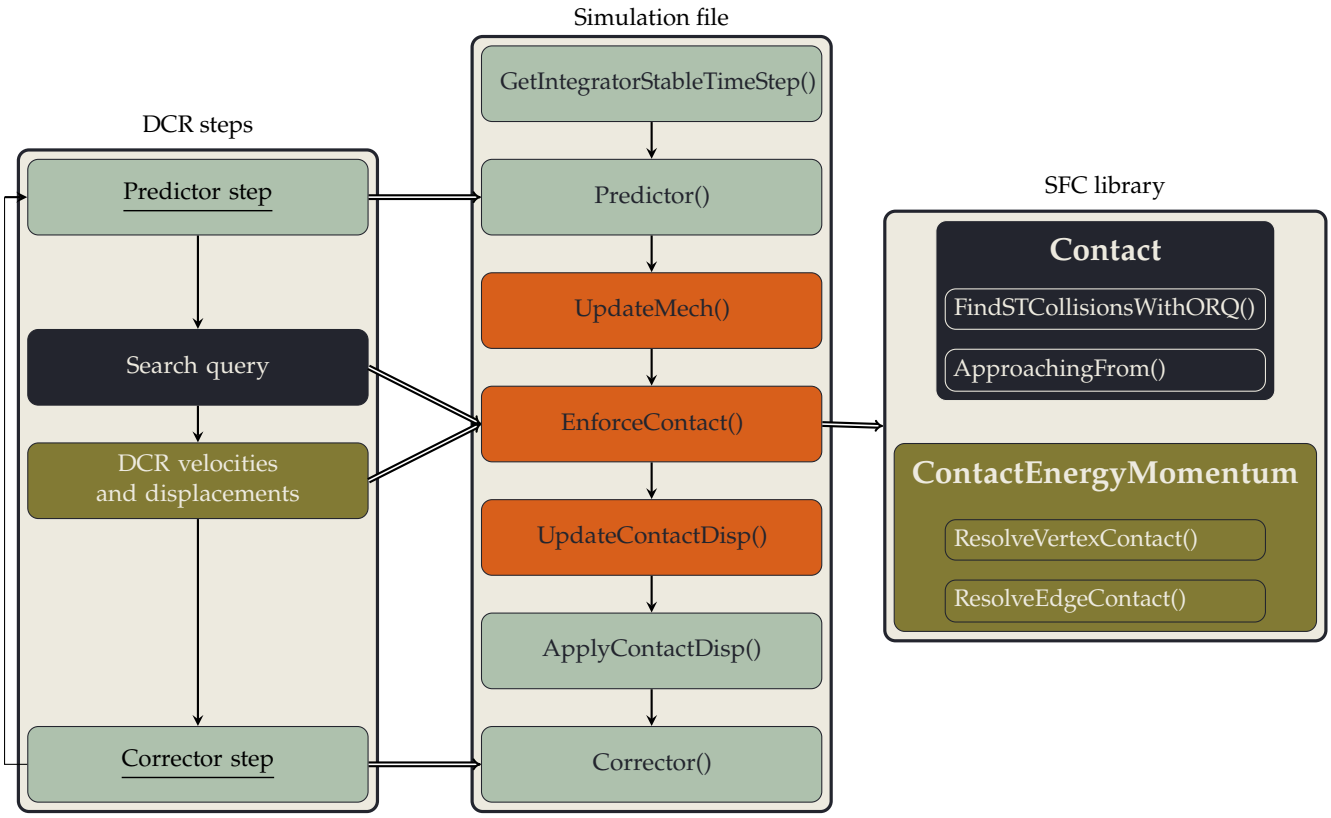


Figure 3.9: Flow chart illustrating the methods and classes called during the solving of one time step of the DCR algorithm. On the left, the DCR algorithm tasks are listed. In the center, the methods called in the simulation file, and on the right the SFC classes performing each of the DCR tasks. In green are indicated the methods of Summit's *Solver* object and their corresponding tasks. In orange the methods from Summit's *Contact surface*. In dark blue the methods from the *Contact* class and their DCR tasks, and finally in olive the methods from *ContactEnergyMomentum* and their corresponding DCR tasks.

The methods to set the stable time step, to perform the predictor and corrector's numerical integration and to copy the variables from *ContactSurface* to *System* are respectively `GetIntegratorStableTimeStep()`,`Predictor()`,`Corrector()` and `ApplyContactDisp()` and they all belong to *System*.
The `UpdateMech()` copies $u, \dot{u}$ from *ContactSurface* to the corresponding SFC *Mesh* variables, while `UpdateContactDisp()` performs the specular task. The copying of variables from *System* to *ContactSurface* is omitted in figure 3.9.
Finally, the geometrical search and the DCR computations are performed in `EnforceContact()`. Specifically, the first task is performed in *Contact*. Here, `FindSTCollisionsWithORQ()` populates the bounding boxes of all the contact mesh triangles with the vertices and edges within the given tolerance, while `ApproachingFrom()` performs the search and establishes which vertices are violating the geometrical intersection constraints. Lastly, in *ContactEnergyMomentum* the post-impact velocities are calculated and the intersections removed. `ResolveVertexContact()` tackles the node to face interactions, while `ResolveEdgeContact()` solves the edge to edge interactions.

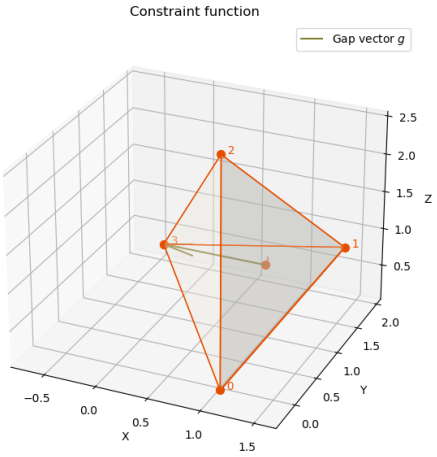
4

Analysis of the available DCR implementation for continuous finite elements

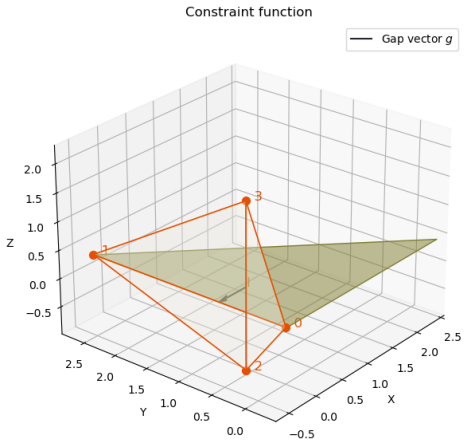
Implementing a numerical algorithm for an analytical methodology always introduces limitations which can be due to limits of the chosen programming languages, urge to meet time constraints, unforeseen issues, coding habits and many more possible reasons. After describing the theoretical background utilized for the DCR algorithm in section 3.2, its formulation and implementation will be analyzed in this chapter. In section 4.1 the edge-to-edge contact case will be inspected and examined, while section 4.2 will focus on the indexing dependency created by the current implementation for continuous Finite Elements (FEs). Lastly, in section 4.3 the findings will be summarized and possible solutions to discovered limitations will be listed. This solutions have not been implemented due to the time constraints of master graduation project.

4.1. Limitations of the edge-to-edge contact analytical formulation and of its implementation

Considering a 3D tetrahedral mesh of a solid mechanics problem, the corresponding SFC contact mesh is composed of 2D triangles. This a precise code design choice which follows from the use of a CG FE formulation: since no fracture is modeled, all the solids involved in the simulation can only experience contact with each other on their outer boundary. Within the domain of non-smooth geometries (i.e., solids with sharp edges, like cubes) two types of impact can happen, namely node-to-face and edge-to-edge, as shown in subsection 3.2.2. To evaluate these impact events, the DCR algorithm defines the constraint function $g(\mathbf{X})$ as the volume of the tetrahedron defined by the 4 edges involved in the impact. In figure 4.1a and figure 4.1b, the constraint functions are shown for the two cases. In these figures and in the rest of this document, *leader* identifies the control entity in the contact interaction (i.e., the impacted triangle or edge) and *follower* the subordinate entity (i.e., the impacting node or edge).



(a) Example of how the constraint function $g(\mathbf{X})$ is defined for a node-to-face impact. Node 3 is impacting on leader triangle $\overset{\triangle}{012}$. Point I is the projection of 3 on the leader triangle. The constraint function is shown as the orange tetrahedron. The olive triangle boundaries are hidden from the constraint function sides.



(b) Example of how the constraint function $g(\mathbf{X})$ is defined for an edge-to-edge impact. Segment $2-3$ is impacting on leader triangle. Point I is the intersection of $2-3$ and of the leader triangle. The constraint function is shown as the orange tetrahedron. The leader edge is $0-1$.

For the node-to-face case, the chosen constraint function allows to define a gap vector $\mathbf{g}$ as the normal distance between the impacting node and the leader triangle, which is then used to define g, ∇g and the normal post-impact momentum $\mathbf{I}^+_{norm}$ as shown in equation (3.16). This is a consistent process, and the tangential motion is then tackled using $\mathbf{I}^+_{slide}$ and $\mathbf{h}(\mathbf{X})$. In figure 4.2 a sketch showing orientation of the constraint function's gradient for the node-to-face case is given, which allows to visualize the gradient vector used to calculate $\mathbf{I}^+_{norm}$.

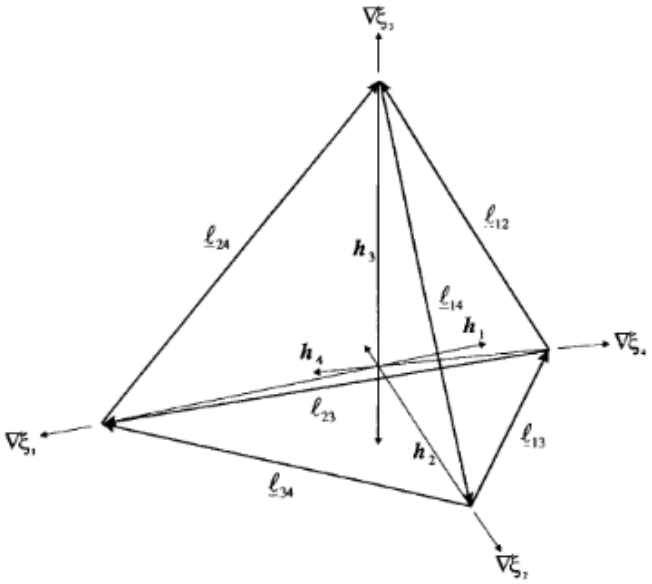


Figure 4.2: Tetrahedron edge, height and gradient vectors illustration, courtesy of [62]. The edge vectors are identified by $\updownarrow_{ij}$, the height by $\mathbf{h}_i$ and the gradient vector by $\nabla\xi_i$.

For the edge-to-edge case the constraint tetrahedron is not defined consistently with the momentum decomposition. Thus, the solution for post-impact velocities is not split between normal and tangential component as in the node-to-face case. Indeed, having two follower vertices makes not immediate to compute a separation vector $\mathbf{h}(\mathbf{X})$ analogous to the one used for the tangential momentum component in the node-to-face case. An analogous separation vector could for example be computed with the projection of both ends of the impacting segment, but it would require a much more complicated

formulation.
Additionally, the gap vector **g** is defined on the plane of the leader triangle, instead of following the impact direction as for the node-to-face case. Thus, on has $g = \mathbf{g} \cdot \mathbf{n} \sim 0$ by definition. This basically corresponds to assuming the impact to happen parallel to the leader's triangle plane. This is also reinforced by the fact that the velocities of the two impacting vertices s_0, s_1 are completely ignored when computing the gap vector, as mentioned in subsection 3.2.4.
Ignoring the velocities of the impacting edge can also lead to situations like that in figure 4.3, where choosing the edge at shortest distance from the intersection I leads to selecting the wrong leader edge even in the case of a perfectly tangential impact.

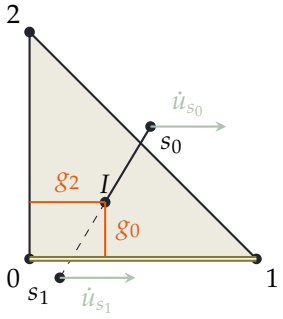
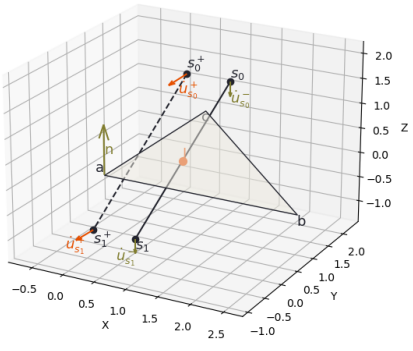
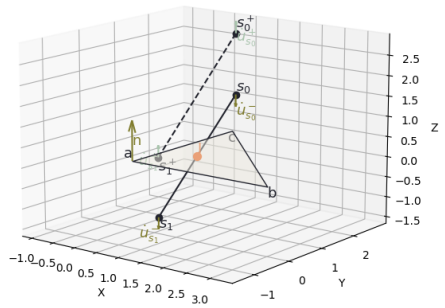


Figure 4.3: Graphical example for a case in which the current edge-t-edge algorithm would select the wrong leader edge based on the criterion of minimum normal distance from the intersection of triangle and edge I. Edge $s_0 - s_1$ is impacting edge $0 - 2$, as shown by the velocities $\dot{u}_{s_0}$ and $\dot{u}_{s_1}$, represented by the light green arrows. Nonetheless, the gap vector (represented by the orange segments) with the smaller magnitude is g_0, and thus the leader edge for this interaction is erroneously inferred to be segment $0 - 1$ (highlighted in olive).

Moreover, not considering the velocity of impacting edge can lead to the post-impact velocity having a skewed orientation, leading to non-physical violation of the energy conservation principle in the simulation. Indeed, if the impact happens in normal direction w.r.t. the leader triangle surface, its velocities will still be computed using a tangential gap vector. Therefore, the post-impact velocity component of the impacting edge will be tangential to the leader triangle's plane, instead of normal and opposite to the initial velocity. This situation is illustrated with an example in figure 4.4.



(a) Graphical example showing the resulting velocities and the projection obtained by using the current implementation of the edge-to-edge contact. The resulting velocities (the orange vector) now have a component in the $-X$ direction and a spurious displacement has been applied in $-X$ direction to remove the intersection.



(b) Graphical example showing the resulting velocities (the light blue vector) and the projection expected for an edge-to-edge elastic impact between rigid bodies. The intersection has been removed and the velocity is oriented in $+Z$ direction.

Figure 4.4: Segment $s_0 - s_1$ is impacting the leader triangle traveling in the $-Z$ direction. Its initial position (obtained after the predictor step) is represented with the continuous line, while its position after the projection is represented by the dashed segment. Figure 4.4a shows the result of the current implementation, while figure 4.4b shows the expected physical behaviour for an elastic impact between rigid bodies.

Some additional limitations are embedded in the edge-to-edge routine due to the chosen implementation. As mentioned in subsection 3.3.2, this routine is implemented in `ResolveEdgeContact()`,

a method of the `ContactEnergyMomentum` class. Three features that are not directly derived from the analytical formulation but have been embedded in the method are:

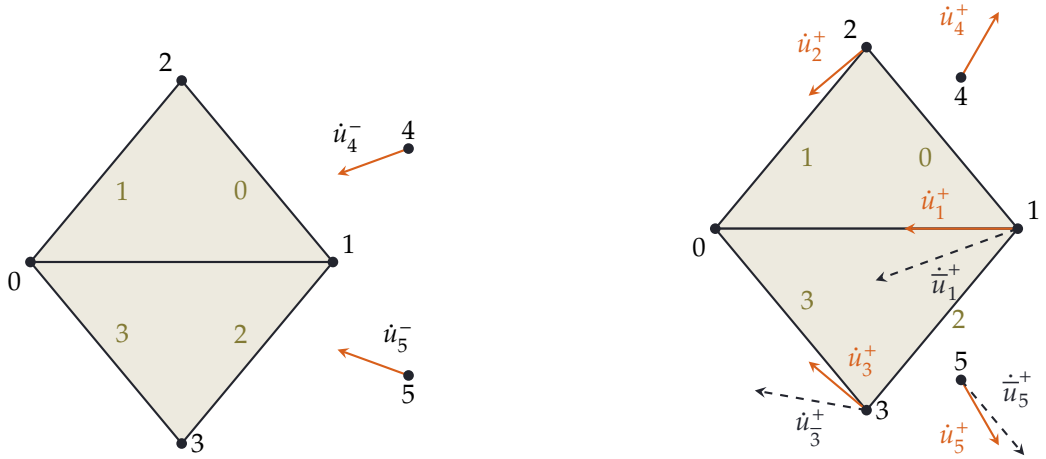
- arbitrary definition of leader and follower: the edge-to-edge contact routine does not allow to clearly identify a base triangle and a vertex for the constraint function. Indeed, two nodes belong to the impacting edge and two to the impacted edge. Therefore, the leader-follower relations end up being defined by the internal indexing of the contact mesh. This means that when removing the inadmissible intersection, the impacted FE could be moved instead of the impacting one, since it might be considered as the follower edge by `ResolveEdgeContact()`. This could become an issue if different interaction are projected erroneously in opposite or normal directions, since it would create fictitious contact displacements.
- the impacting edge is assumed to be rigid: this assumption is created by projecting both of its vertices of the same quantity to remove the contact intersection. This is not necessarily true, since the impacting edge might shorten, elongate and/or rotate during the impact.
- no search is performed to flag edge-to-edge interactions and their constraint function is not assessed: the gap vector $\mathbf{g}$ on the impacted triangle plane implies that $g \sim 0$ by definition, as previously mentioned. Thus, the constraint function for edge-to-edge cannot be evaluated as it is done for the node-to-face case. This means that the intersection is not re-evaluated during the solving loop of `ResolveEdgeContact()`. Hence, the edge-to-edge interactions that are removed or created by the node-to-face routine `ResolveVertexContact()` could still be tackled by the edge-to-edge one. Additionally, the edge-to-edge routine solves all of the interactions which are not directly excluded, instead of only those flagged by an appropriate search.

4.2. Internal indexing dependency

Based on a thorough inspection of the methods and classes utilized, additional limitations have been discovered due to the implementation of the coupling between SFC library and the solid mechanics Summit libraries. Two main limitations are particularly important to point out: the use of direct overwriting during post-impact velocity computations and the indexing dependency arising from multiple impacts happening on the same finite element.

4.2.1. Direct overwriting of nodal velocities

In the contact `Mesh` object each vertex is saved in a unique memory position, and then the elements (triangles) are created as a set of pointers to the vertices. Each vertex has attributes like velocity, position, intersection flags, etc. When performing calculations, the vertex velocity is called and directly overwritten. Therefore, the internal indexing of the elements is affecting the resulting velocities. Consider the 2D example of a double node-to-face impact in figure 4.5 as an example.



(a) Velocities before the impacts

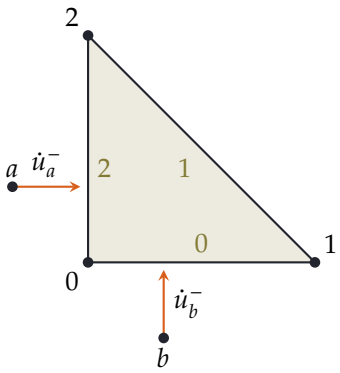
(b) Velocities following the impact

Figure 4.5: Example of a situation involving one node in multiple impact events. The mesh of the example is 2D for simplicity, and thus the elements are 2D triangles and the contact mesh is composed of 1D segments. Nodes 4 and 5 are set to respectively impact face 0 with velocity $\dot{u}_4^-$ and face 3 with velocity $\dot{u}_5^-$. All other nodes are initially not moving. The olive numbers represent the contact faces, while the dark blue numbers the nodes. The orange arrows represent the expected velocities and the dark blue dashed arrows show the numerical results. The erroneous velocities are obtained due to the order in which the impact are solved. Indeed, the impact of node 4 is solved first, and thus for node 5 the post impact velocity is computed using $\dot{u}_1^+$ instead of $\dot{u}_1^-$.

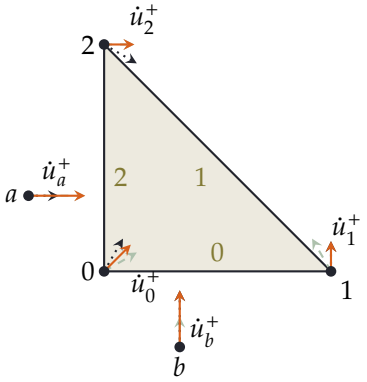
Element 1 is impacted by node 4, while element 2 is impacted by node 5. Given the chosen element indexing, the first impact solved is node 4 on contact face 0: pre-impact nodal velocities $\dot{u}_1^-, \dot{u}_2^-, \dot{u}_4^-$ are fed to `ResolveVertexContact()` to compute post-impact velocities $\dot{u}_1^+, \dot{u}_2^+, \dot{u}_4^+$. Then, the process proceeds with the impact of node 5 on face 2 using as pre-impact velocities $\dot{u}_1^+, \dot{u}_3^-, \dot{u}_5^-$. Therefore, the post-impact velocities $\dot{u}_{1,f}^+, \dot{u}_{3,f}^+, \dot{u}_{5,f}^+$ would end up being a function of the DCR post-impact velocities. With a different element indexing, the post-impact velocities would not be the same.

4.2.2. Multiple impacts on the same finite element

In the case of multiple impacts registered on one finite element in the same time step, the indices of the impactors affect the resulting velocity. The SFC code has been developed for a continuous finite element formulation and for tetrahedral meshes, meaning that no fracture is modeled during the simulation. Thus, only the outer boundary of the solids involved in the contact is relevant for modelling. Therefore, the mesh is structured around surfaces and not around bulk (volume) elements. Under these assumptions, it is reasonable to assume that each 3D element in the mesh will have at maximum one 2D face involved in the contact mesh, especially if the mesh size is small enough.
Nonetheless, this assumption can create issues in some edge cases. For example, consider the impact of nodes a and b on triangle $\overset{\triangle}{012}$. The impacts are set to happen at the same physical time, but their solving order is determined by the relationship between the node indices. If $a < b$, the impact of a will be solved first, and thus the post-impact velocities $\dot{u}_0^+, \dot{u}_1^+$ will be computed using the post-impact value of $\dot{u}_0$, leading to non-physical values. If $a > b$, the opposite will happen and non-physical values will be obtained for $\dot{u}_0^+, \dot{u}_2^+$.



(a) Pre-impact situation. Nodes a, b are set to impact with velocities $\dot{u}_a^-, \dot{u}_b^-$ respectively on faces 2 and 0 of the finite element. The velocities are shown as the orange vectors.



(b) Post-impact situation. Nodes a, b have impacted respectively on faces 2 and 0 of the finite element. The expected velocities are shown as the orange vectors, the velocities for $a < b$ are shown as the dashed light blue arrows and the velocities for $a > b$ are shown as the dotted dark blue arrows.

Figure 4.6: Graphical example to show the possible outcomes for multiple impacts on the same 2D finite element. Nodes a, b are set to impact with velocities $\dot{u}_a^-, \dot{u}_b^-$ respectively on faces 2 and 0. In figure 4.6a the pre-impact situation is shown, with the two impacting nodes moving and the finite element standing still. In figure 4.6b the post-impact situation is shown, with the expected velocities and the velocities which would be obtained for both $a < b$ and $a > b$.

4.2.3. Arbitrary precedence of node-to-face over edge-to-edge interactions

Lastly, the node-to-face impacts have been assigned arbitrary precedence over the edge-to-edge impacts, due to their implementation order. Indeed, as shown in listing 4.1, the node-to-face method `ResolveVertexContact()` is called before the edge-to-edge `ResolveEdgeContact()`. To tackle this issue they have been implemented in SFC within a `while` loop, allowing for multiple iterations for each time step.

```
1
2     // building bounding boxes and search grid for all mesh faces
3     findSTCollisionsWithORQ();
4     // finding node-to-face intersections
5     approachingFrom();
6
7     int changed;
8     int count = 0;
9     int maxIter = 1;
10
11    // solving loop
12    do {
13        changed = 0;
14        // node-face collisions
15        changed = resolveVertexContact();
16        // edge-edge collisions
17        changed += resolveEdgeContact();
18        ++count;
19    } while ((changed) && (count < maxIter));
```

Listing 4.1: The `while` loop used in SFC to implement a numerical convergence process on the contact solution. The loop continues until the maximum numbers of iteration *maxIter* is reached and until some contact iteraction is solved. Indeed, most C++ compilers transform any `int` > 0 into `true`, meaning that *changed* = `false` only if it remains 0.

By changing the value of `MaxIter`, one could ensure that for each time step of the solid mechanics simulation the contact calculations are performed multiple times until reaching convergence. The `Changed` variables ensure that at least one impact is solved for iteration. This solution lowers the computational efficiency of the simulation, and in the worst case the direct superposition of contact displacements u_c used to remove the geometrical intersections could hinder the numerical convergence. Therefore, it has not been employed in this thesis work.

4.3. Summary of the limitations

To summarize, multiple limitations have been found in the current SFC implementation. These limitations can be divided into two main subgroups, one of arising from the edge-to-edge interaction

formulation and one from implementation choices.
The limitations of the edge-to-edge formulation are:

- $E1$ The gap vector is not defined consistently with the height of the tetrahedron used for the constraint function, and thus $g = \mathbf{g} \cdot \mathbf{n} \sim 0$ by definition. Therefore, the edge-to-edge intersections cannot be evaluated in the same way as the node-to-face ones.
- $E2$ The tetrahedron chosen to define the constraint function and how the leader edge is selected are forcing the post-impact velocities to be computed in the same direction regardless of the relative motion of the two edges.
- $E3$ Neither the following edge velocity nor the leading edge's are considered for the purpose of the contact. This might lead to un-physical velocities. Considering the relative velocity between leading and following edge should avoid this by inferring information about the contact direction.
- $E4$ The edge-to-edge routine selects leader and follower based on only the internal indexing and removes the intersection by projecting just the follower edge. Thus, non-physical displacements could be introduced in the system if the impacted object is considered as the follower.
- $E5$ The impacting edge is considered to be rigid and is the only one that is moved during the removal of the inadmissible intersection.

The limitations created by the implementation choices are mostly related to an inherent indexing dependency. This choices have been made envisioning very large meshes and limited volatile memory, but thanks to the progress of the hardware obtained in the last 15 years this architecture could be changed without excessive cost in terms of computational time. The limitations which have been discovered during the analysis of the code are:

- $I1$ When mesh nodes are involved in multiple impact events (i.e., impacts happen in the same time step on neighbouring finite elements), the internal indexing of the contact mesh elements affects the post-impact velocities.
- $I2$ When multiple impacts happen on the same element, the order in which the impacts are solved affects the post-impact velocity. This order is solely determined by the internal indexing of the impacting nodes, which adds chances to obtain non-physical results.
- $I3$ Impacts resolved by the node-to-face routine have been arbitrarily been chosen to have precedence over edge-to-edge.
- $I4$ There is no intersection search performed for edge-to-edge interactions, meaning that only the node-to-face flags of the edge vertices can be used to assess the intersection. Since an edge-to-edge intersection could happen without their vertices being flagged and vertices could be flagged without an edge-to-edge intersection, this could lead to solving for spurious interaction.

Some of these limitations could be tackled with minimal modifications to the software architecture of SFC, while some would need a deep redesign of the code. The former group is tackled in this thesis work, while the latter will be only discussed on a conceptual level. Indeed, the time constraint of this master graduation project do not allow for a complete reshaping of SFC.
All of limitations discovered in the edge-to-edge contact routine (items $E1$ to $E5$) have not been tackled in this thesis work. Possible solution have been proposed in section 4.4. Regarding the implementation limitations, items $I1$ and $I2$ are tackled with the measures described in section 5.3. Limitation item $I4$ is partially considered in subsection 5.4.1, with the addition of criteria to limit the possibility of `ResolveEdgeContact()` solving a spurious interaction. Item $I3$ and the addition of an edge-to-edge contact search (item $I4$) would have required to heavily modify the SFC architecture, and thus have not been solved in this thesis work. A possible approach for their solution is mentioned in section 4.4.

4.4. Possible solutions to algorithmic limitations

Not all of the limitations discovered in the DCR algorithm could be tackled within the tight time constraints of a master thesis work. This section is intended to give an overview of possible solution approaches for the edge-to-edge formulation limitations items $E1$ to $E5$ and for the implementation limitations items $I3$ and $I4$.

4.4.1. Changing the mesh unit from surface to volume elements (items $I1$, $I2$ and $E4$)

The first step in fixing the current edge-to-edge solving routine would be to perform a search analogous to what is done for nodes on the edges, in order to flag those creating inadmissible geometric intersection at the end of the predictor step. This could also allow to intersect the results of the node and of the segment searches to avoid solving twice the same interaction. For example, using the signed volume approach mentioned in [63] for the search would allow to first flag the intersecting elements and then determine which routine to use for dealing with their interaction.
The current SFC library implementation uses a contact mesh which is based on vertices and triangles. The triangles do not have any information about the bulk element of the FE mesh they originate from. Moreover, the edges of the triangles do not use a dedicated data structure and are saved simply as pairs of nodes. Performing a search on the edges and intersecting node and edge results would be made much easier by using as the base unit of the contact mesh bulk elements, instead of surface ones, and by creating a dedicated data structure for edges. Additionally, implementing the `Edge` and `Tetrahedron` classes would allow to employ the concept of simplicial complexes [64]. These are sets composed of points, segments, triangles and tetrahedra (when stopping at 3 dimensional objects). For a given simplicial complex K, each face of a simplex from K is in K and the non-empty intersection of two simplices a, b from k is a face belonging to both a and b. This means that by defining each FE as a simplex, one would have direct access to the set of its faces, edges and nodes and to their connectivity. Similarly, each face could already store the information about its edges and vertices, and so on.

4.4.2. Modifying the edge-to-edge gap vector (items $E1$ to $I4$)

Using simplicial complexes and bulk elements would also allow to change the gap vector definition for edge-to-edge interactions. Some examples of possible approaches for the detection of edge-to-edge intersection are the signed volume approach for non-smooth geometries based on the use of isoparametric coordinates explained in [63] with emphasis on hexaedral quadrilateral elements or the so-called "mortar method"[65], which uses an average of the gap function along the length of the impacting segment. The latter could be implemented in the current architecture as well, while the former would need more effort but would allow to avoid having two intersection search loops. Indeed, the search loops could be performed on the bulk element, checking for a volume intersection. Then, different criteria could be implemented to assess which kind of interaction is observed (e.g., node-to-face or edge-to-edge). In this way the risk of solving multiple times the same interaction would also be removed. Moreover, edge-to-edge contacts would not be using node-to-face intersection flags and neither of the two routines would have arbitrary precedence.
Another possible way to fix the edge-to-edge interaction would be to consider the relative velocity between the impacting edge and impacted triangle to determine the impact direction and select the correct impacted edge. For example, one could estimate a time to impact in a similar way to what is done in subsection 3.2.3, and then use it together with the velocities of impacted triangle and impacting edge to find if at any point in time two edges are intersecting.

4.4.3. Removing the rigid impactor assumption (item $E5$)

Lastly, the rigid body assumption for the impacting edge needs to be reconsidered. The impacted object is assumed to be rigid and extensible for both the 2D and 3D cases [14]. However, when an edge is the impactor, the current DCR implementation assumes a rigid body motion and projects both of the segment vertices of the same quantity. Depending on the kinematics of the system this assumption could hold or not. Considering the velocities of the nodes participating to the edge-to-edge interaction and combining it with sampling the interpenetration of the impacting edge w.r.t to the impacted triangles at multiple points could allow to determine eventual rotations and elongations/contractions of both of the edges involved in each interaction, obtaining a more physically sound post-impact solution.

5

Extending the DCR methodology to discontinuous finite elements

Chapter 4 discusses the current limitations observed in the Decomposition Contact Response (DCR) formulation and in its implementation, while in chapter 3 are given the theoretical background for the Discontinuous Galerkin/Cohesive Zone Method (DG/CZM) framework and for the DCR algorithm. The following step is to adapt the DCR algorithm to a Discontinuous Galerkin (DG) setting. This entails redefining the contact mesh to include the discretized DG nodes, with a series of measures described in section 5.1. Then, displacement compatibility at element boundaries must be enforced following the intersection removal process of DCR. This has been tackled by introducing the concept of *DG sibling node set*, which is explained in section 5.2. Additionally, direct overwriting of the nodal velocities have been removed by exploiting more modern C++ data structures in section 5.3, with the objective of removing the internal indexing dependency highlighted in section 4.2. Lastly, the presence of inter-element boundaries is expected to be the source of spurious contacts detections and therefore in section 5.4 the steps implemented to avoid the detection of DG numerical contacts before fracture onset are illustrated. For each of these sections, analytical examples and/or simulations are used for verification of the effectiveness of the measures taken. Additionally, in section 5.5 two numerical simulations are performed to assess the effect of the modifications introduced until this point.

5.1. Creating a Discontinuous Galerkin contact mesh

In a problem using the Continuous Galerkin (CG) formulation the contact mesh includes only the faces on the outer boundary of the Finite Element (FE) domain, since no fracture is expected. Therefore, the two meshes have different morphologies and a mapping between FE and contact mesh nodes is needed to pass displacement, velocities and other variables between SFC and Summit. The continuous contact mesh is created with a geometrical search which assumes the nodal coordinates to be unique, which thus can not be directly translated to the DG formulation.
Moreover, when using the DG/CZM formulation any FE face could take part in a contact interaction due to the possible onset of fracture and multiple discretization nodes are superposed to each topological vertex. Thus, two groups of contact mesh elements can be identified: internal faces and external (or boundary) faces. The latter correspond to the faces utilized by the CG mesh, while the former are the faces corresponding to the DG inter-element interfaces.
Two approaches may be followed to model contact at internal faces, *intrinsic* and *extrinsic*. The former would consist of inserting all the contact faces (including the internal ones) from the start of the simulation and use DCR to enforce displacement compatibility. This would unreasonably expensive from the computational point of view and would mean to lose the wave propagation properties of DG/CZM, and so it was discarded.
A second approach would be to add them progressively to a continuous contact mesh as the fracture progresses. A mapping would be needed between the continuous contact mesh and the DG/CZM finite element mesh, and then the contact mesh should be updated at every time step involving fracture onset or propagation. This *extrinsic* approach to the creation of the contact mesh has been discarded for three

main reasons: on a first stance, it would need to heavily modify the contact mesh for every iteration where the fracture pattern evolves, hindering the expected computational efficiency. Secondly, the choice of the mapping algorithm would be somewhat arbitrary, given that to each node in the contact mesh would correspond multiple in the FE mesh. Lastly, the updating process of the contact mesh could severely limit the parallel implementation of this methodology, since the mesh generation would need to stop the computation of all processors for assigning the newly introduced nodes. Hence, an *hybrid intrinsic-extrinsic* approach chosen for this thesis seems more favorable.
The *hybrid intrinsic-extrinsic* approach consists in creating the contact faces at the start of the simulation and in activating them only when the corresponding DG/CZM interface is cracked. Apart from the more complicated implementation, the main drawback of the *hybrid* approach would be the possible instabilities arising from spurious detection of contact between internal faces of the mesh. As soon as fracture onsets, the same problems would occur also when using either of the two other approaches, and consequently the *hybrid* approach is chosen.
In the current implementation of the SFC library the contact mesh is initialized and assembled in the constructor of the `Mesh` class shown in figure 3.8. At first, a mapping between FE nodes and contact nodes is created , and then two variables are populated to pass the FE mesh morphology to SFC: a vector linking contact nodes to their spatial coordinates and a vector linking the contact mesh faces to the contact nodes. SFC has been developed explicitly for 3D tetrahedral meshes. Thus, the contact faces are 2D triangles and a left-handed, cartesian, orthonormal reference system is used. Thus displacements, coordinates, etc. have the form $\mathbf{a} = [a_x, a_y, a_z]^T$.
Taking the *hybrid* approach means that all the FE nodes have to be included in the contact mesh. Therefore, the FE and contact nodes have a 1-to-1 correspondence:

$$ID_{i,contact} = ID_{i,DG}$$

For each i-th node. The $ID > 0$ is an integer which uniquely identifies a node in the mesh. It follows that the discretized coordinates from the FE can be directly used to link contact mesh nodes to their coordinates. Afterwards, the nodes need to be mapped to the contact faces.

5.1.1. Assembling the internal contact faces

The internal faces are completely absent in the contact mesh of a simulation utilizing the CG framework. These faces arise from the DG inter-element interfaces. Each DG interface element is responsible for computing the DG/CZM boundary integrals shown in equation (3.5) and of the constitutive response of the DG/CZM cohesive elements once that these are activated. In figure 5.1 a graphical example of an interface element in a 2D mesh is shown. It is important to note that only one interface element exists for each inter-element boundary, meaning that to every DG interface element correspond two FEs (tetrahedra for a 3D mesh) and two contact faces (triangles for a 3D mesh).

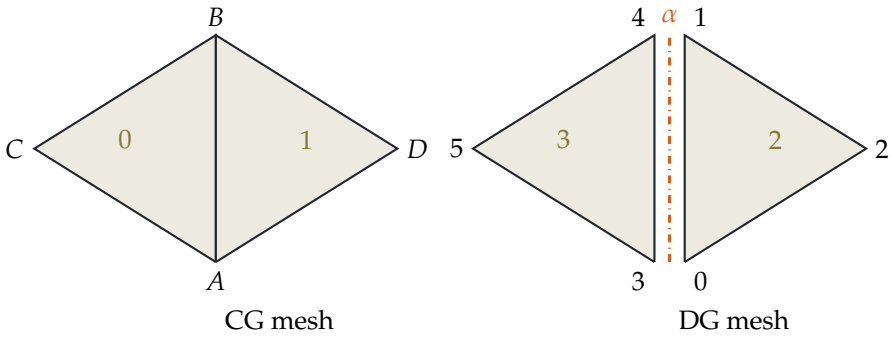


Figure 5.1: Graphical example of a 2D mesh, showing the difference in discretization between the CG and a DG formulation. In CG the nodes are shared between elements and only one face exists for each inter-element FE interface (on the left). In DG, each FE possesses its own nodes and to each inter-element interface correspond two contact faces. Additionally, in DG an interface element (here called α) is inserted at each inter-element boundary to evaluate the DG/CZM boundary integrals and to manage the cohesive behaviour of the interface.

Each DG interface element possesses the information of which nodes belong to the two tetrahedra linked by the interface an attribute. This information is stored in a class attribute with the following structure:

$$[ID_0^+, ..., ID_4^+, ID_0^-, ..., ID_4^-]$$

Where superscripts $^+$ and $^-$ indicate respectively the left and right tetrahedra. The node indices are ordered to ensure that the volume of each tetrahedron is computed as positive. The order in which the nodes are assigned to the contact faces influences the orientation of their surface unit normal vector. Thus, it is necessary to save the triplet of nodes defining each face in the correct order.
Considering a tetrahedron defined by the set of vertices $V = \{v_i \mid i \in [0, 1, 2, 3]\}$, its faces are given by the four possible combinations without repetitions of its vertices. In SFC the surface normal of a face with vertices V_a, V_b, V_c is computed as $\mathbf{n} = (\mathbf{X}_{V_c} - \mathbf{X}_{V_a}) \times (\mathbf{X}_{V_b} - \mathbf{X}_{V_a})$. Consequently, assuming the first vertex of each face to always have the lowest index $0 \leq i \leq 3$, only one ordered set of combinations gives 4 faces with an outward surface normal vectors:

$$F = \{(V_1, V_3, V_2), (V_0, V_2, V_3), (V_0, V_3, V_1), (V_0, V_1, V_2)\} \tag{5.1}$$

This can be used to test the $^+$ and $^-$ sets of vertices for each of the combination in the set of equation (5.1). Indeed, the two contact faces corresponding to the DG interface will be the two ordered triplets with the same centroid. The assembly process of the internal faces can be summarized as:

1. the interface connectivity is retrieved
2. the nodes are split between tetrahedra
3. for each i-th combination of 3 nodes on the $^+$ tetrahedron
   (a) the combination's centroid $Cg_i^+$is computed
   (b) for each j-th combination of 3 nodes on the $^-$ tetrahedron
      (i) the combination's centroid Cg_j^- is computed
      (ii) if $\mathbf{X}_{Cg_i^+} \sim \mathbf{X}_{Cg_j^-}$, the i-th combination is save as the $2I$-th contact face and the j-th combination is saved as the $(2I + 1)$-th contact face
      (iii) if no match is found, the j is increased
   (c) if no match is found for $0 \leq j \leq 3$, i is increased

At this point, contact mesh has has size $2n_{interfaces}$, with every even-odd consecutive pair of contact faces corresponding to the same DG interface element.

5.1.2. Assembling the boundary contact faces

The centroid of the boundary faces is unique, and thus the `Summit.Mesh.Face` objects can be used to retrieve the node triplets corresponding to each face. Nonetheless, Summit does not store a mapping between DG nodes and the mesh faces, and thus this information needs to be created.
Each `Summit.Mesh.Face` contains only the unordered set of nodes belonging to its tetrahedron. The nodes are thus mapped to each FE boundary face in ordered triplets during the discretization of the FE mesh, following a process analogous to what is done for the internal faces:

1. the centroid Cg_i of the i-th `Summit.Mesh.Face` is computed
2. the leader tetrahedron and its DG nodes are retrieved
3. for each j-th combination of nodes in the set F of equation (5.1)
   (A) the centroid Cg_j of the combination is computed
   (B) if $\mathbf{X}_{Cg} \sim \mathbf{X}_{Cg_i}$, the nodes are saved. If not, j is increased

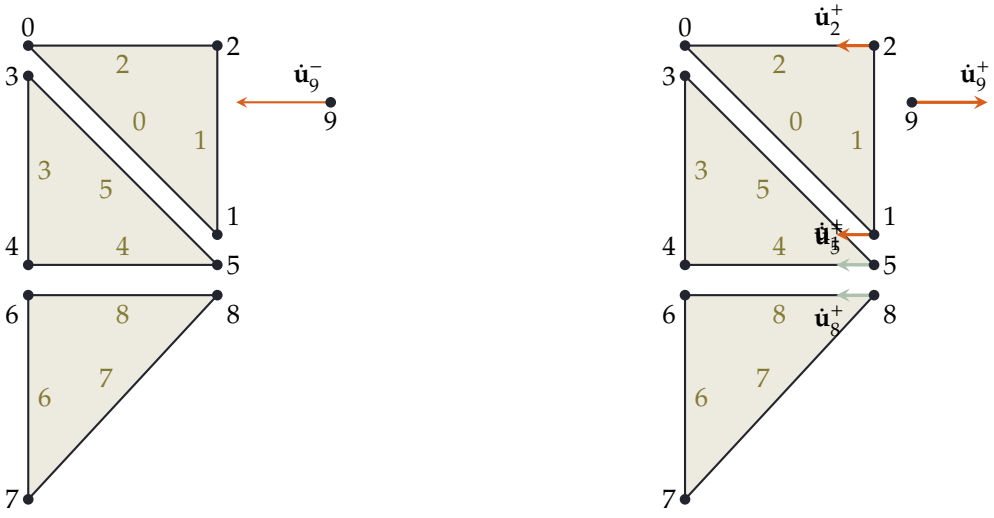
Afterwards, the boundary faces are first picked from the set of all mesh faces. It is important to note that this set includes both external and internal geometrical faces. A geometrical search on the centroids is used to discriminate between boundary and internal faces, since the internal faces of the contact mesh have already been created. The process to create the mapping can be summarized as:

1. the centroid CG_i of the i-th `Summit.Mesh.Face` is computed
2. if $\mathbf{X}_{CG_i}$ does not correspond to the coordinate of the centroid of any DG interface element
   (A) the corresponding nodes are stored as the $2n_{interface} + i$-th element in the contact mesh

The use of two separated loops allows to store the number of boundary faces n_b in the contact mesh. This is useful to distinguish between internal and boundary faces in further stages of the simulation. Indeed, with f_{DG} being the total number of faces, n_{int} the number of internal faces and n_b the total number of boundary faces in the contact mesh, the elements with index $0 \leq I < f_{DG} - n_b = n_{int}$ are internal faces and those with $n_{int} = f_{DG} - n_b \leq I < f_{DG}$ are boundary faces.
In a CG mesh FE mesh and contact mesh nodes have different numbering, meaning that mapping variables and search methods are needed when passing information between Summit and SFC. The two mapping methods utilized to do so `UpdateMech()` and `UpdateContactDisp()` are shown in figure 3.9. Given the 1-to-1 correspondence of DG and contact nodes, the mappings in these two functions are superfluous when using the DG formulation. Therefore, copying routines are implemented to move values between Summit and SFC.

5.2. Enforcing displacement compatibility before fracture onset

The DG/CZM boundary integrals manage the interaction between the bulk elements sharing the same DG interface. This means the DCR algorithm should not detect and/or solve interactions between neighbouring FE before the onset of fracture. The DCR algorithm applies velocities and displacements to the nodes involved in the impacts, but not to their neighbouring elements. In the CG formulation, to each topological vertex corresponds one discretization node. Therefore, applying the displacement computed by DCR's intersection removal routine to a node would influence all the elements sharing it. In the DG formulation each element possesses its own discretization node, which is superposed to those sharing the same spatial coordinates. Thus, applying the intersection removal displacement to only one node would lead to a violation of displacement compatibility. Same holds for the post-impact velocities. A graphical example is given in figure 5.2, where node 9 is set to impact on face 1. Note that a 2D mesh is utilized for an easier representation, meaning that FE mesh elements are 2D triangles and that contact mesh faces are 1D segments.



(a) Velocities before the impact.

(b) Velocities following the impact.

Figure 5.2: Example of an impact on a simplified 2D DG mesh. The elements are 2D triangles and the contact mesh is composed of 1D segments. Node 9 is set to impact contact face 1 with velocity $\dot{\mathbf{u}}_9^-$, which is shown in orange. All other nodes are initially not moving. The DCR algorithm only computes post-impact velocities $\dot{\mathbf{u}}_1^+, \dot{\mathbf{u}}_2^+, \dot{\mathbf{u}}_9^+$. Hence, to enforce displacement compatibility node 5 and 8 should also be given velocity $\dot{\mathbf{u}}_5^+ = \dot{\mathbf{u}}_8^+ = \dot{\mathbf{u}}_1^+$. $\dot{\mathbf{u}}_5^+, \dot{\mathbf{u}}_8^+$ are shown in light green in the figure.

The DCR algorithm will compute the post-impact velocities of the nodes 1,2,9. This means that after the DCR computations, node 1 will have a different velocity compared to nodes 5 and 8. Thus, displacement compatibility will be violated when advancing in time. To avoid this violation, the concept of DG sibling node set is introduced.

5.2.1. The DG sibling node sets

For each DG node, its sibling set is defined as the set of DG nodes stemming from the same topological vertex. Given that SFC does not possess any topological information about the DG discretization, the sibing sets are assembled by using the fact that all the members of a given set are sharing the same geometrical coordinates at the start of the simulation. The set of node i can be defined as:

$$S_i = \left\{ j,\ \forall j \mid \mathbf{X}_j(t = t_0) \sim \mathbf{X}_i(t = t_0) \right\}.$$

Where $\mathbf{X}_j(t = t_0)$ are the coordinates of node j at the initial time t_0.
To assemble such sets, a geometrical search on the coordinates of the FE nodes is performed at the start of the simulation and a 2D vector is used to store the result of the search. The entries of this vector are defined as:

$$S_i = \begin{cases} [-1] & \text{no superposed nodes} \\ \left[j,\ \forall j \mid \mathbf{x}_j \sim \mathbf{x}_i \right] & \text{if superposed nodes exists} \end{cases} \tag{5.2}$$

Using -1 for the nodes with an empty sibling set allows to have a continuous vector and to use the index as the node ID.
This data structure is then used to apply the DCR post-impact velocities to all the superposed DG nodes and enforce displacement continuity during the intersection removal process. This means that the nodes in the siblings set of the follower nodes are projected back to the contact face.
In addition, the sibling set are also used to consider the mass of the neighbouring FE during the calculation of the post-impact velocities. Indeed, each node in a CG mesh possesses the mass contribution of all the elements that share it. Instead, in a DG mesh to each node is assigned only the mass contribution of the element to which it belongs. For example, assuming that all nodes in figure 5.2 have equal mass m, during the momentum calculation the mass matrix entries would look like:

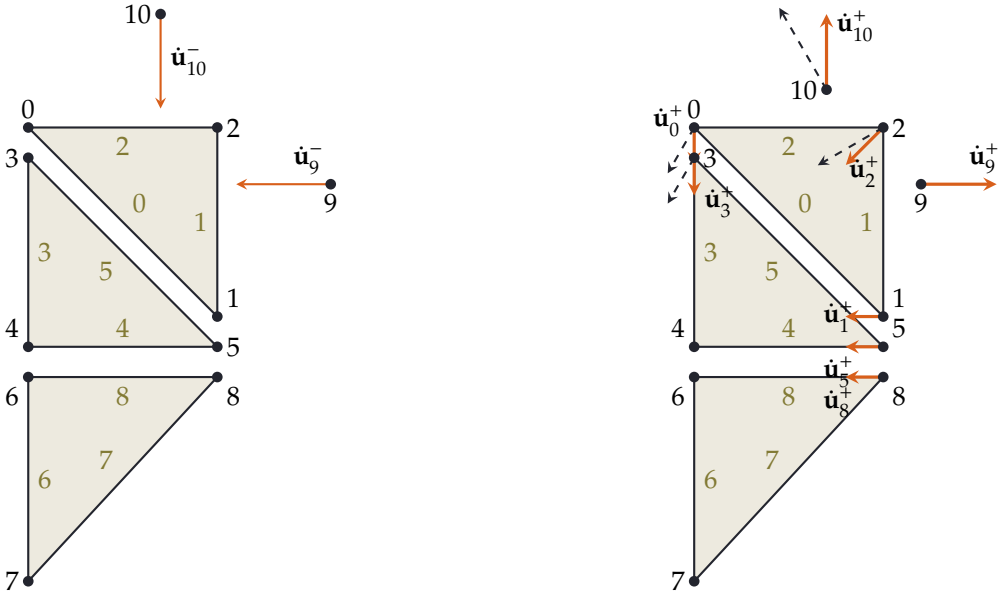
$$\mathbf{M} = \begin{cases} m & \text{for nodes } 2, 9 \\ 3m & \text{for node } 1 \end{cases} \tag{5.3}$$

Because of nodes 5,8 being siblings of node 1. Therefore, for a node a with n_{S_a} sibling nodes, the contribution to the mass matrix can be written as:

$$M_{aa} = m_a + \sum_{j=0}^{n_{S_a}} m_j.$$

5.3. Removing direct overwriting of post-impact velocities

The use of sibling nodes enforces displacement continuity,. Nonetheless, it does not solve the indexing dependency issues of the contact response highlighted in section 4.2. For example, consider the situation in figure 5.3. Once again, the problem is set-up in 2D for simplicity. Two impacts are expected, node 9 on face 4 and node 10 on face 1.



(a) Velocities before the impact.

(b) Velocities following the impact. In orange are shown the expected velocities and in dark blue the velocities which would be computed with the current implementation.

Figure 5.3: Example of a double impact on a single FE with a simplified 2D DG mesh. The FE are 2D triangles and faces of the contact mesh are 1D segments. Node 9 is set to impact face 1 with velocity $\dot{\mathbf{u}}_9^-$ and node 10 is set to impact face 2 with velocity $\dot{\mathbf{u}}_{10}^-$. All other nodes are not moving. The DCR algorithm will first compute post-impact velocities $\dot{\mathbf{u}}_1^+, \dot{\mathbf{u}}_2^+, \dot{\mathbf{u}}_9^+$ and then the siblings routine will update the velocity of 5 and 8 using $\dot{\mathbf{u}}_1^+$. Therefore, the post-impact velocities of node 10 will be computed with a wrong set of initial velocities, obtaining flawed velocities $\dot{\bar{\mathbf{u}}}_0^+, \dot{\bar{\mathbf{u}}}_w^+, \dot{\bar{\mathbf{u}}}_{10}^+$. The siblings routine will then trickle down the error updating the velocities of node 3, leaving 4 out of 11 nodes in the simulation with a flawed velocity. If faces 1 and 2 had their index switched, the simulation would have ended up with 5 out of 11 flawed post-impact velocities.

The impact of node 9 will be solved first by the DCR algorithm because the solving loop is performed on the contact mesh faces, and face 1 has a lower index than face 2. Thus, the velocities $\dot{\mathbf{u}}_1^+, \dot{\mathbf{u}}_2^+, \dot{\mathbf{u}}_9^+$ will be computed correctly. Then, the siblings set routine will overwrite the velocity of node 5 and 8 giving $\dot{\bar{\mathbf{u}}}_5^+ = \dot{\bar{\mathbf{u}}}_8^+ = \dot{\mathbf{u}}_1^+$. Afterwards, when the impact of node 10 is solved, the initial set of velocities used for DCR computations will be $\dot{\mathbf{u}}_0^-, \dot{\mathbf{u}}_2^+, \dot{\mathbf{u}}_{10}^-$, meaning that $\dot{\bar{\mathbf{u}}}_0^+, \dot{\bar{\mathbf{u}}}_2^+, \dot{\bar{\mathbf{u}}}_{10}^+$ will not respect momentum conservation. Additionally, the error will trickle down further due to the siblings routine setting $\dot{\mathbf{u}}_3^+ = \dot{\bar{\mathbf{u}}}_0^+$. Out of 11 nodes involved in the simulation, 4 will end up with velocities violating the momentum conservation. This issue is tackled by removing the direct overwriting of nodal velocities and using a temporary variable to store the post-impact velocities until the end of each time step. A new variable is created using a map data structure type, which allows to store a set of key-value pairs. Each time a node i is involved in the contact computations, the map keys are checked. If i is missing, and entry is added as $i : v_i = [0, 0, 0]$. Then, the post-impact velocity is superposed to the entry v_i value. The post-impact velocity is thus stored while the nodal velocity is left unchanged. This is repeated for all contact interactions and only at the end of all contact computations the nodal velocities are overwritten.

5.4. Avoiding the detection of spurious DG contacts

The introduction in the contact mesh of the faces corresponding to the DG interfaces means that the detection of spurious contacts might hinder the simulation results. Spurious contacts are contacts which arise only from numerical approximation or slight interpenetrations of FE and would not be observed with a CG formulation of the problem. For example, during the discretization of a DG/CZM mesh, the FEs are created from a topological mesh description. When the different DG nodes originating from the same topological vertex are created, their coordinates might differ due to numerical approximation.
To avoid the detection of these contacts, two aspects of the code have been addressed: the search for inadmissible intersections and the computation of post-impact velocities for both node-to-face and edge-to-edge impacts. Spurious contacts could be handled more easily if SFC possessed information

about the bulk elements connectivity, but since this not available it has been tackled with the use of numerical tolerances and additional criteria.

5.4.1. Detection of inadmissible intersections

The search for intersections is performed in two steps: at first, an Orthogonal Range Query (ORQ) data structure is created for each contact mesh face and is populated with vertices and edges in its neighborhood. Then, for each triangle i in the contact mesh, a gap vector $\mathbf{g}_{ij}$ is computed with each node j. If the constraint function $g_{ij} = \mathbf{g}_{ij} \cdot \mathbf{n}_i < 0$, node j is flagged as intersecting. As mentioned in section 4.1, this is done only for the nodes. This basically means that the gap vector and outward unit surface normal vector $\mathbf{n}_i$ have opposite direction. It is important to note that:

- edges are not included in this intersection search process,
- the vertices are only flagged as intersecting, but no information is stored w.r.t. which contact face is being intersected,
- this search is performed before the solution of node-to-face contacts and the flags are not updated during the solution.

The search box for the intersections is defined by extruding the face of the leader triangle by $\varepsilon_{norm} = u_{max}$ parallel and opposite to $\mathbf{n}_i$. Additionally, the dimensions of the triangle in its plane are increased in each direction by a factor ε_{tang}.
Two measures are implemented to refine the search and to avoid the detection of spurious contact:

1. a tolerance ε_{DG} is added to the evaluation of g_{ij}, which thus becomes $g_{ij} < \varepsilon_{DG}$ to account for the initial numerical interpenetration which could be found at the start of the simulation
2. the vertices can not be flagged as intersecting if they belong to the sibling node set of any of the leader's triangle vertices

5.4.2. Node-to-face filtering criteria

The use of the `approachingSide` flag is not the only condition used by SFC to determine if an intersection should be dealt with. Indeed, some configurations could give a gap vector satisfying the conditions mentioned in subsection 5.4.1 but not correspond to a contact interaction. Considering the CG implementation of the node-to-face routine, the following cases have been excluded:

$C_{vtx}1$ $|\mathbf{g}|^2 > |\mathbf{u}_{max}|^2$,
$C_{vtx}2$ $|\mathbf{g}| < \varepsilon_{lim}$,
$C_{vtx}3$ if the projection of the follower node position is not within the boundaries of the leader triangle,
$C_{vtx}4$ if the follower node has been already moved.

Criterion $C_{vtx}1$is used to avoid detecting nodes simply being behind the leader triangle and not impacting from being flagged. For example, this could happen in a box surface, where without this condition the nodes on the opposite face w.r.t. the leader element could be seen as in contact. Criterion $C_{vtx}2$ is enforced to make sure that the gap vector $\mathbf{g}$ is numerically relevant. Indeed, if its norm is smaller than the smallest value obtainable with a C++ `double`, it can be assumed that $\mathbf{g} = [0, 0, 0]$ and that there is no relevant intersection.
The introduction of the sibling sets and of the internal faces in the contact mesh for the DG formulation led to the addition of three criteria for ignoring the interaction:

$C_{vtx,DG}1$ the follower node belongs to the sibling set of any of the leaders,
$C_{vtx,DG}2$ $g > 0$,
$C_{vtx,DG}3$ the follower node belongs to the same tetrahedron as the leader triangle.

Criterion $C_{vtx,DG}1$ is implemented to avoid solving multiple times for the same contact interactions. Criterion $C_{vtx,DG}2$ is added because the use of sibling sets for intersection removal can sometimes lead to nodes with a `approachingFrom()` flag indicating an intersection being already moved by other interactions being solved earlier.
In the case of very large deformation of the elements, it is expected that the some tetrahedra could get

close to a null volume due to excessive deformation and flattening. Therefore, criterion $C_{vtx,DG}3$ is added to check if the impacting node does not belong to the same tetrahedron as the impacted triangle. The information is created during the internal faces creation loop described in subsection 5.1.1 and saved in SFC into a newly created class attribute for each triangle during the contact mesh creation process.

5.4.3. Edge-to-edge filtering criteria

The criteria used to discriminate between spurious and non-spurious intersection for edge-to-edge impacts are similar to those used for the the node-to-face routine. The ones used in the CG formulation are:

$C_{edg}1$ $|\mathbf{g}|^2 > |\mathbf{u}_{max}|^2$,
$C_{edg}2$ $|\mathbf{g}| \sim 0$,
$C_{edg}3$ the follower edge lies entirely on or is parallel to the leader edge's triangle plane,
$C_{edg}4$ any of the vertices of the follower edge have been moved by the node-to-face routine.

The two criteria $C_{edg}1$ and $C_{edg}2$ work similarly to those explained in subsection 5.4.2. Criterion $C_{edg}3$ has been probably implemented to avoid solving for impacts where the impacting edge belongs to the impacted triangle. Criterion $C_{edg}4$ avoids solving with different interaction the same interaction. The condition deriving from the modification introduced to apply the DG formulation is:

$C_{edg,DG}1$ any of the follower vertices belong to the sibling set of any of the leaders,
$C_{edg,DG}2$ both follower vertices or one of the leader vertices have been already moved during the current time step,
$C_{edg,DG}3$ leader and follower edges belong to adjacent faces and are adjacent to each other.

All of these criteria are intended to avoid solving the same contact interactions multiple times and to limit the possibility of detection of spurious contacts due to the inter-element DG interfaces.

5.5. Numerical verification and qualitative validation of the DG DCR algorithm

The modifications to the DCR algorithm described in this chapter have been verified and qualitatively validated with numerical simulations. Very low number of elements have been utilized for multiple reasons: quicker turnaround times, easier meshing process, easier tracing of issues and bugs during the development phase, and expected convergence issues due to the limitations of the edge-to-edge contact routine described in section 4.1.
In subsection 5.5.1 a first simulation involving only 4 mesh elements has been performed to test the sibling sets implementation. Then in subsection 5.5.2, a 8 elements mesh has been used to assess the sensitivity to load rate (by changing the impact velocity) and to assess the effect of inclusion of the frictional contact model.

5.5.1. Verification and qualitative validation of the sibling sets implementation

The implementation of the sibling sets has been tested with the impact simulation of the simplest possible geometry including a DG interface in the impactor, one in the target, and both node-to-face and edge-to-edge contact. Therefore, a solid meshed with two elements traveling with initial velocity $v = 15m/s$ is made to impact with a second solid, also meshed with two elements. A neo-hookean material model [66] is chosen, with the bulk material properties shown in table 5.1. These are chosen to obtain a high compliance of the FEs, while the interface material is a "pure DG" one. This basically corresponds to set $\alpha = 0$ in equation (3.5) for all the DG interfaces in the mesh, meaning that the cohesive elements are never activated during the whole simulation.

Property	Value	Unit
E	0.5	MPa
ρ	5000	kg/m^3
ν	0.3	-

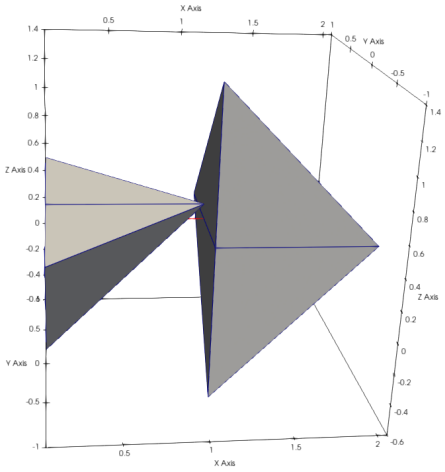
Table 5.1: Bulk material properties for neo-hookean material. The parameters are: Young's modulus E, mass density ρ and Poisson's coefficient ν.

In table 5.2 other simulation parameters are given.

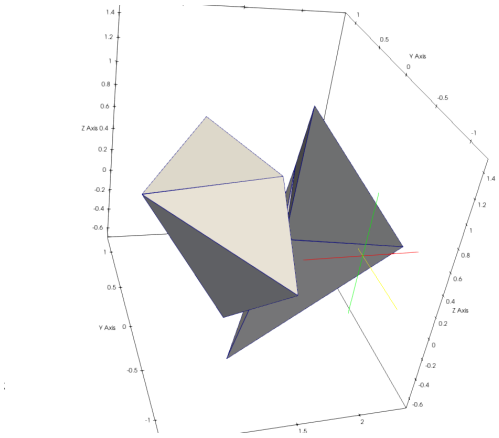
Property	Value	Unit
v	15	m/s
f_t	0.8	-
e	2	-
μ	0	-
ε_{tang}	$1e^{-3}$	-
β_s	100	-
ε_{ang}	$\cos(5^\circ)$	-
ε_{DG}	$1e^{-12}$	-

Table 5.2: Simulation parameters: initial velocity v given to the single tetrahedra in the positive $+X$ direction; time factor f_t, which is used as a safety factor on the DG/CZM stable time step; coefficient of restitution e, which is set to obtain a fully elastic response; Couloumb's friction coefficient μ, tangential tolerance coefficient ε_{tang} (used to define the bounding boxes used for the search of inadmissible intersections), DG/CZM stabilization parameter β_s, ε_{ang} numerical tolerance to determine if two edges are parallel, and ε_{DG} numerical tolerance to avoid detecting DG numerical inter-element interface intersections.

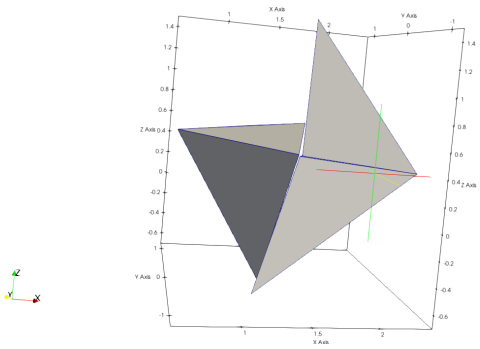
In figure 5.4 the results of this simulation are shown. At time $t = 0.003\ s$ a first node-to-face is registered, shown in figure 5.4a. Then, the simulation progresses with the stress waves traveling through the elements and leading to deformation and rotations of the two elements. Then, at $t = 0.090\ s$ an edge-to-edge contact is registered, which is shown in figure 5.4b. Given the asymmetrical mass distribution of the two solids, this leads to a rotation of the impactor which creates a second edge-to-edge contact at $t = 0.092\ s$. Lastly, at $t = 0.101\ s$ a last edge-to-edge contact makes the two solids to drift apart and no more impact events are observed until the end of the simulation. The latter two impacts are shown respectively in figure 5.4c and figure 5.4d.



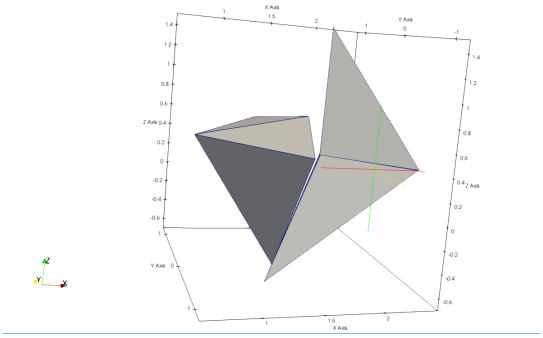
(a) First node-to-face impact at $t = 0.003\ s$



(b) Second overall and first edge-to-edge impact at $t = 0.090\ s$. The right element of the impactor hits the top element of the target. The solids are starting to rotate due to their asymmetrical shape.



(c) Second edge-to-edge impact at $t = 0.092\ s$. The impactor pivots on the impact point of figure 5.4b and thus collides with the lower element of the target.



(d) Third edge-to-edge impact at $t = 0.102\ s$. The impactor pivots around the $Z - axis$ in clock-wise direction, and thus the interaction is again between the right element of the impactor and the bottom element of the target.

Figure 5.4: Screenshots displaying the simulation geometry used to verify the sibling set implementation and the notable events observed during its evolution. Observing figure 5.4a, the solid element on the left represent the impactor and is given initial velocity v. The solid on the right represents the target and is standing still at the start of the simulation.

To assess if the use of sibling sets leads to a violation of energy conservation, the internal work and kinetic energy of the system are plotted against time. Additionally, to measure the effect of the DCR algorithm on the energy conservation, a routine is added for computing the projection energy. Indeed, the predictor step of DCR creates inadmissible intersections between the FE. This intersections are then removed and post-impact velocities are computed based on momentum conservation. The intersections are removed by closest point projections. These projections add numerical energy to the system, which could create some numerical discrepancies. For a given time step, the projection work can be estimated as:

$$W_{proj} = \left(f^i - f^e - f^s \right) \cdot \mathbf{x}_c \tag{5.4}$$

Where f^i, f^e, f^s are respectively internal, external and DG/CZM interface forces and $\mathbf{x}_c = \|\mathbf{g}\|$ is the projection displacement.
The evolution of the system's energy is shown in figure 5.5. The first node-to-face impact shows no effect on the total system energy, and as expected the initial kinetic energy decreases with an equal and opposite increase of the internal work or deformation energy. The following edge-to-edge contacts, create instead oscillations in the kinetic energy of the system. This is due to the issues highlighted in section 4.1. Indeed, the edge-to-edge post-impact velocities are influenced by the assumption of the impact happening tangential to the leader's triangle plane, and therefore the velocities will not exactly

meet energy conservation. Specifically, the bigger discontinuity is visible with the fourth impact, which is also the one with the biggest deviation from a tangential impact out of the three. Moreover, the DG interface are forced to stay closed and the constant readjustments needed to minimize their intersections.

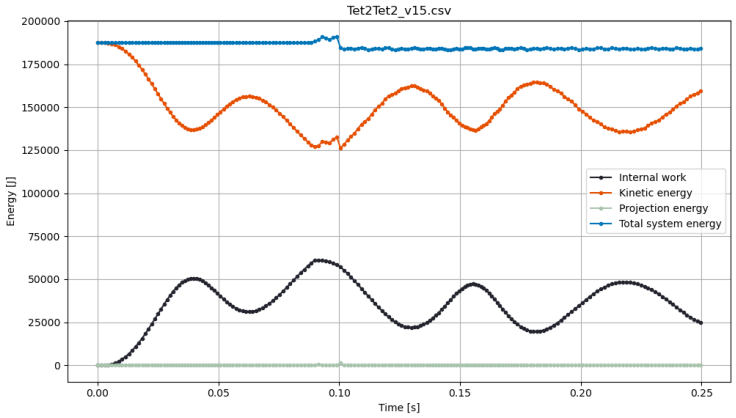


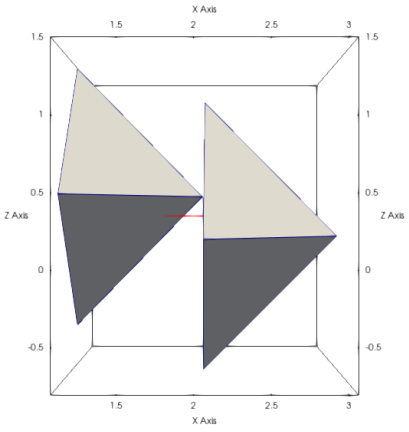
Figure 5.5: Verification of energy conservation over time. In blue is represented the sum of internal work (in black), kinetic energy (in orange) and projection energy (in green). Each dot represents a simulation output value. It is is visible how the first node-to-face impact at $t = 0.009\ s$ did not have any influence on the overall system energy, while the subsequent edge-to-edge impacts at $t = 0.090\ s$, $t = 0.092\ s$ and $t = 0.102\ s$ create discontinuities in the kinetic energy of the system. This is due to the skewed post-impact velocities computed by the edge-to-edge routine, as explained in section 4.1. The oscillation observed for $t > 0.102\ s$ are due to tangential oscillations in the DG interfaces.

5.5.2. Further tests on the DG DCR algorithm

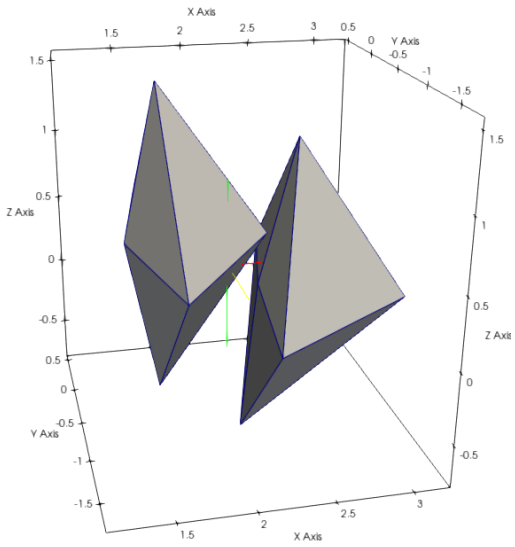
After testing the implementation of the sibling sets, the DG DCR algorithm has been tested with a slightly more complicated geometry. One solid meshed with four elements is traveling with initial velocity v and impacts an identical solid being at a standstill. Once again, a neo-hookean material model is chosen, with same bulk material properties shown in table 5.1. The interface material is also chosen to obtain a "pure DG" interface behaviour. Each of the two solids in the geometry is shaped so that its four elements share one edge. In this way, possible simulation stability issues due to spurious contacts should be highlighted more easily. The simulation parameters are the same as those in table 5.2, with the exception of velocity v and friction coefficient μ. Three initial velocities are tested, being $v = 5, 15, 25\ m/s$. For $v = 15\ m/s$ the simulation is performed for $\mu = 0$ and $\mu = 0.25$, to gain some insight on the effect of the frictional contact formulation.

Solid on solid impact ($v = 5m/s$)

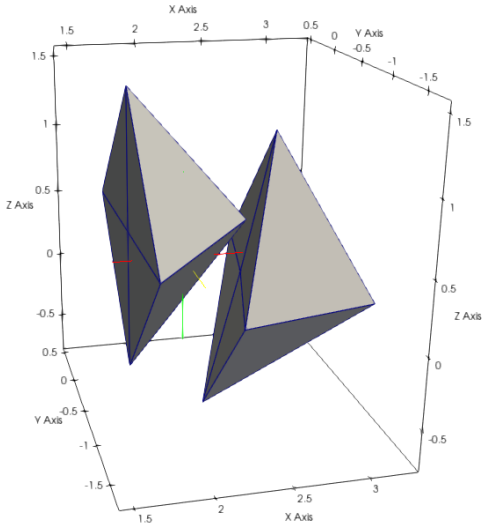
For an initial velocity $v = 5\ m/s$, three impacts are recorded. At $t = 0.012\ s$, the first node-to-face impact is registered, as shown in figure 5.6a. The system energy starts to be divided between kinetic and elastic deformation energy. Then, a second node-to-face impact is observed at $t = 0.144\ s$, visible in figure 5.6b. Finally, a third node-to-face impacts is observed at $t = 0.202\ s$ and is shown in figure 5.6c. These last two impacts happen between the same face and node and are caused by the impact stress waves, which are making the compliant solids to pulsate, creating a periodic to expansion and contraction cycle. This expansion-contraction behaviour is clearly visible from the system energies shown in figure 5.7. This pattern does not show signs of damping since no dissipation is included in the system.



(a) First node-to-face impact at $t = 0.012\ s$



(b) Second node-to-face impact at $t = 0.144\ s$.



(c) Third node to facse impact at $t = 0.202\ s$. It is clearly visible that the same node is impacting the same face as in figure 5.6b.

Figure 5.6: Screenshots displaying the simulation geometry used to verify the sibling set implementation and the notable events observed during its evolution. Observing figure 5.6a, on the left the solid element on the left represent the impactor and is given initial velocity $v = 5/s$. The solid on the right represents the target and is not moving at the start of the simulation.

It can also be observed that the total system energy shows no oscillations. Some minor fluctuations can be observed and are once again due to the adjustments performed by the DG interface to enforce displacement compatibility. The discontinuity created by the third node-to-face impact is more noticeable due to higher value of the intersection.

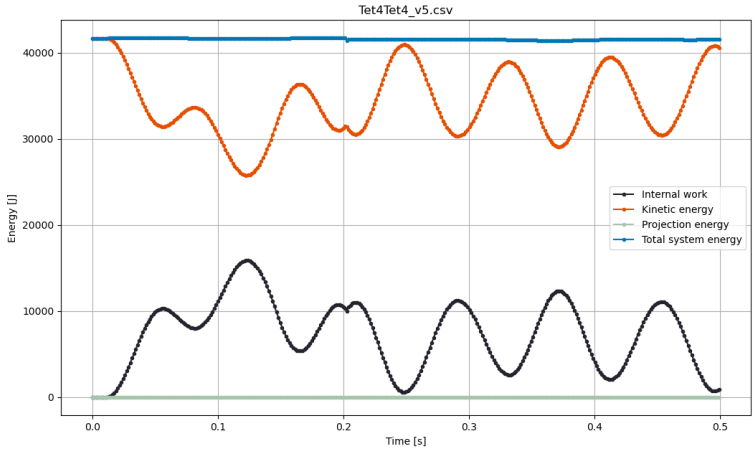
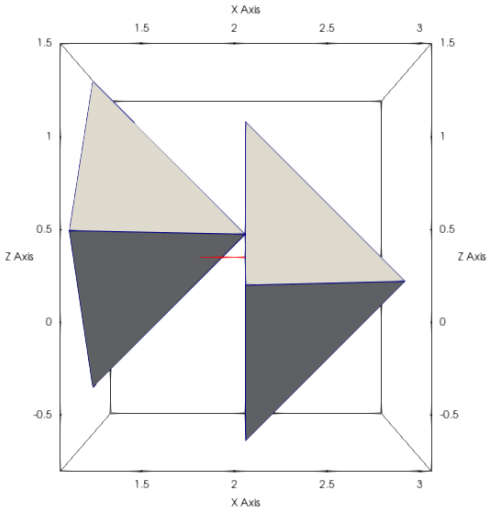


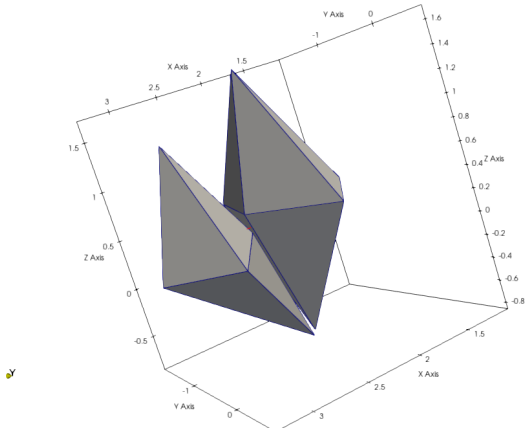
Figure 5.7: Verification of energy conservation over time. In blue is represented the sum of internal work (in black), kinetic energy (in orange) and projection energy (in green). Each dot represents a simulation output value. It is is visible how the impacts do not heavily affect the total system energy, given that the fluctuations are in the range expected due to the adjustments performed by the DG interfaces, which are independent from the DCR algorithm. The discontinuity at $t = 0.202\ s$ is due to the third node-to-face impact. The conservation of energy displayed with multiple impact events is very promising for further work on the implementation of the DCR algorithm for a discontinuous finite elements formulation.

Solid on solid impact ($v = 15m/s$)

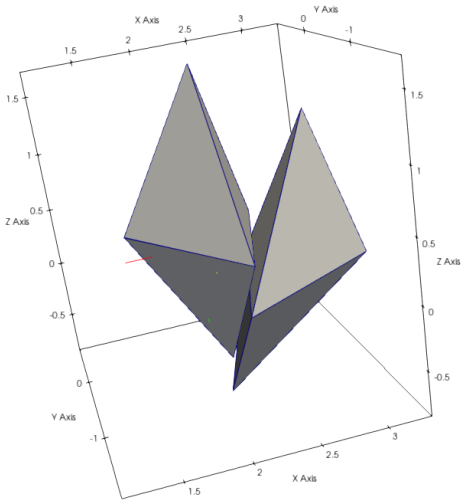
The same geometry has been tested with an initial velocity $v = 15\ m/s$. The results changed noticeably: four hits are recorded and energy conservation is not met. At $t = 0.004\ s$, a first node-to-face impact is registered, shown in figure 5.8a. The system energy starts to be divided between kinetic and elastic deformation energy and energy conservation is not violated. Then, at $t = 0.084\ s$ one node-to-face and two edge-to-edge impacts are observed in the same time step. These impacts are shown respectively in figure 5.8b and figure 5.8c. All three involve the same vertex, which acts as the follower in the node-to-face and as one of the follower's vertices in both edge-to-edge contacts. Obviously, only one of these interactions should be solved in this time step, and thus the conditions added in subsection 5.4.3 are not sufficient. Solving only either of the two edge-to-edge contacts should be sufficient to remove the intersection and compute accurate post-impact velocities. However, the node-to-face routine is called first and therefore there is no way in the current code architecture to flag the vertex for a future edge-to-edge contact solution.



(a) First node-to-face impact at $t = 0.004\ s$. This first impact does not affect the energy conservation.



(b) Node-to-face impact and first edge-to-edge at $t = 0.084\ s$. The impacting vertex belongs to the impacting edge, and thus the same impact event is solved multiple times. This explains the violation of energy conservation shown in figure 5.9.



(c) Second edge-to-edge impact at $t = 0.084\ s$. This third impact also involves the same vertex as those in figure 5.8b, so it is reasonable to infer that it further contributes to violate the energy conservation.

Figure 5.8: Screenshots displaying the simulation geometry used to verify the sibling set implementation and the notable events observed during its evolution. Observing figure 5.8a, on the left the solid element on the left represent the impactor and is given initial velocity $v = 15m/s$. The solid on the right represents the target and is not moving at the start of the simulation.

The violation of energy conservation is clearly shown by figure 5.9. It can be seen how the increase in energy is entirely due to kinetic energy, and this arises by the computation of spurious post-impact velocities. Indeed, all three contact interactions at $t = 0.084\ s$ involve the same vertex, which is thus greatly accelerated.

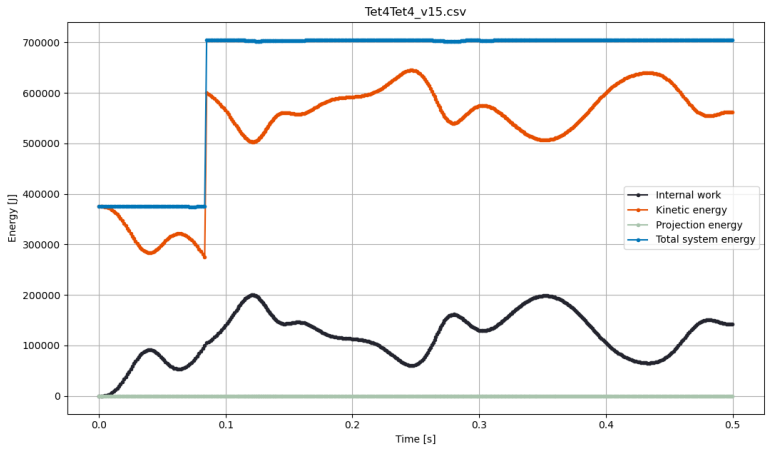
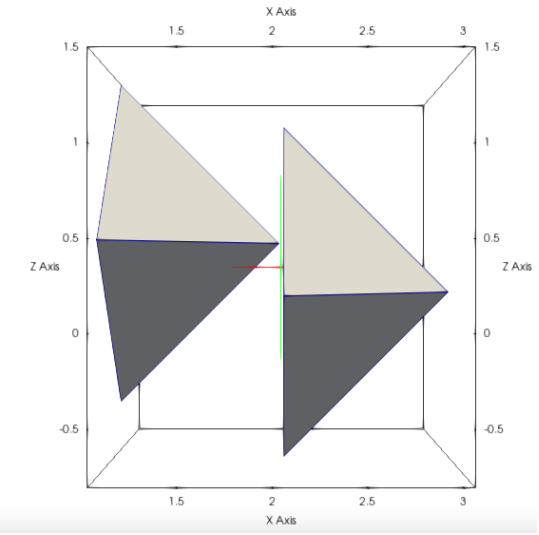


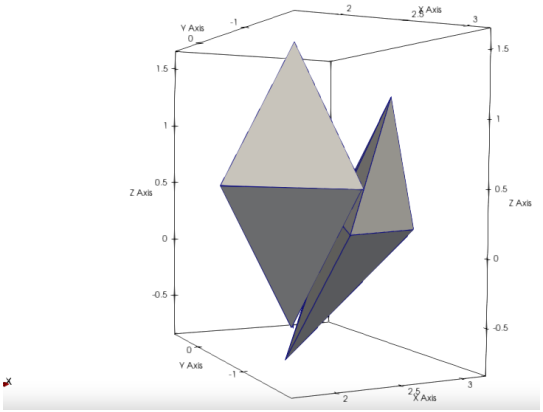
Figure 5.9: Verification of energy conservation over time. In blue is represented the sum of internal work (in black), kinetic energy (in orange) and projection energy (in green). Each dot represents a simulation output value. The main feature standing out in this picture is clearly the kinetic energy injection happening at $t = 0.084\ s$, which creates a non-physical violation of the energy conservation principle. This is due to the detection of multiple impacts involving the same node. Indeed, in this case a node-to-face is solved first, but this doesn't remove the edge-to-edge intersection that caused it, leading to the same physical interaction being solved three times in different numerical interaction. The current data structure used to identify inadmissible intersection are not sufficient to account for this eventuality, and thus should be modified and expanded.

Solid on solid impact ($v = 25m/s$)

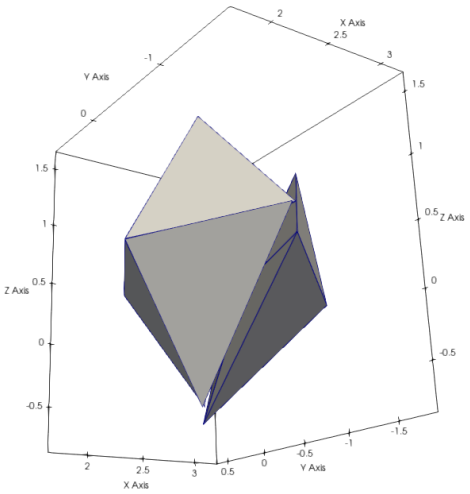
Lastly, the same geometry has been tested for an initial velocity $v = 25\ m/s$. Four impact events have been registered before the crash of the simulation. A first node-to-face impact is observed at $t = 0.002\ s$, shown in figure 5.10a. Then, two edge-to-edge impacts are observed in consecutive time steps at $t = 0.051\ s$ and $t = 0.052\ s$. These impacts are shown in figure 5.10b and figure 5.10c and are involving the same follower edge. In the first impact the edge hits the interface between the two elements of the target, and the computed post-impact velocities make it then impact one of the bottom side in the following iteration. Lastly, the fourth edge-to-edge impact in figure 5.10d is observed at $t = 0.059\ s$. This impact involves the follower edge of the previous two, which is now acting as a master. This can be considered as a proof of the edge-to-edge impact symmetry. Indeed, the kinematics of the impact are not different from the previous two, but the leader and follower relations are inverted.



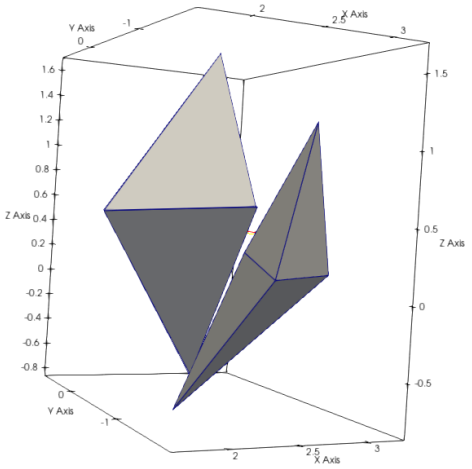
(a) First node-to-face impact at $t = 0.002\ s$.



(b) Second overall and first edge-to-edge impact at $t = 0.051\ s$. The impactor follower edge hits parallel to one of the faces of the target. Thus, two edges are impacted but only one impact is solved. This happens once again due to the assumption of edge-to-edge acting only parallel to the surface of the leader triangle.



(c) Second edge-to-edge impact at $t = 0.052\ s$. Similar consideration can be done as for figure 5.10b, since the follower edge is impacting two edges but only one interaction is solved for.



(d) Third edge-to-edge impact at $t = 0.059\ s$.

Figure 5.10: Screenshots displaying the simulation geometry used to verify the sibling set implementation and the notable events observed during its evolution. Observing figure 5.10a, on the left the solid element on the left represent the impactor and is given initial velocity $v = 25m/s$. The solid on the right represents the target and is not moving at the start of the simulation.

The system's energies are plotted against time in figure 5.11. Oscillations can be seen in the total energy, which can be traced back to the increased work performed by the DG interface elements to minimize their interpenetrations. More work is required by the interfaces due to the higher impact velocity and the following higher deformations. The dip visible at $t \sim 0.18\ s$ can also be explained with the DG interfaces stabilization efforts. This hypothesis is supported by the misalignments visible on the left of figure 5.10c and figure 5.10d at the inter-element interface between the edge of the two elements of the target. The error introduced by this phenomenon is considered acceptable, given that its magnitude is negligible when compared to total system energy.

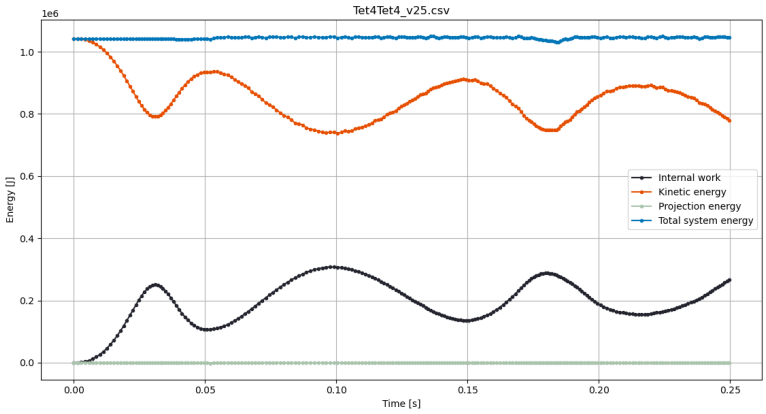
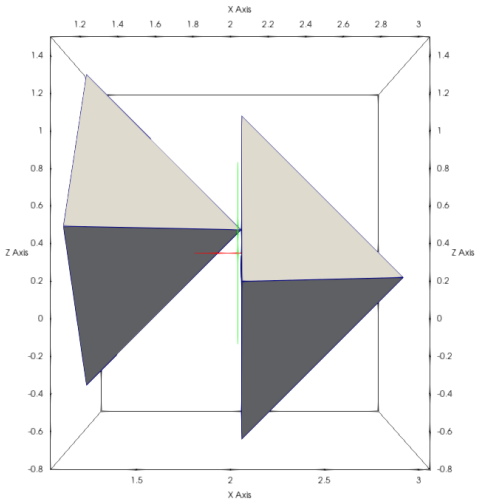


Figure 5.11: Verification of energy conservation over time. In blue is represented the sum of internal work (in black), kinetic energy (in orange) and projection energy (in green). Each dot represents a simulation output value. No clear discontinuities are visible due to the contact routine, but noticeable oscillations can be seen for almost the whole duration of the simulation. This can be explained with a higher impact energy translating to more oscillations in the DG inter-element interfaces. The error introduced by these oscillation is deemed acceptable, given that is magnitude is negligible compared to the total system energy.

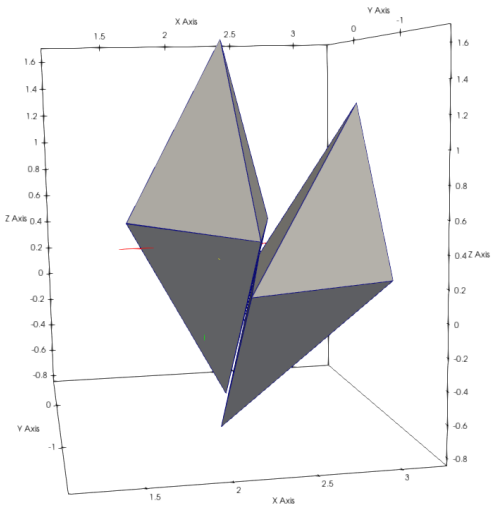
Solid on solid impact with friction ($v = 15m/s, \mu = 0.25$)

The node-to-face solving routine includes a Couloumb's friction model, mentioned in subsection 3.2.4. To qualitatively assess the effect of this model, the same geometry as the previous three simulations has been tested with $v = 15\ m/s$ and $\mu = 0.25$. The remaining simulation parameters are left unchanged compared to table 5.2.

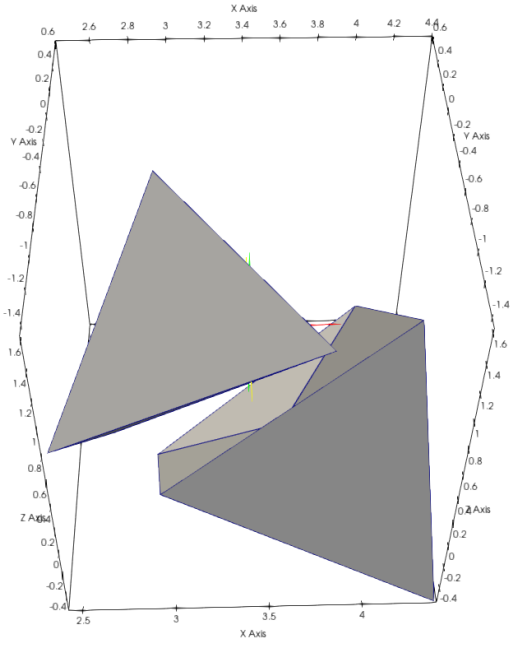
Three node-to-face impact are observed, at $t = 0.004\ s$, $t = 0.084\ s$ and $t = 0.224\ s$. These are shown respectively in figure 5.12a, figure 5.12b and figure 5.12c. The presence of friction leads to a different evolution of the system's kinematics, and thus the impact at $t = 0.084\ s$ happens without creating the edge-to-edge intersections observed in the case with $\mu = 0$ and it is not solved multiple times. Therefore, no injection of kinetic energy is observed in the system, as visible in figure 5.13. This reinforces the idea that a modification of the approach to edge-to-edge impacts is necessary for a complete implementation of the DCR algorithm on FE problems utilizing the DG/CZM formulation.



(a) First node-to-face impact at $t = 0.004\ s$. This first impact does not affect the energy conservation.



(b) Node-to-face impact and first edge-to-edge at $t = 0.084\ s$. The difference with figure 5.8b and the absence of edge-to-edge intersections can be seen clearly.



(c) Second edge-to-edge impact at $t = 0.224\ s$. The high level of deformation observed in this simulation can be seen from the target's shape.

Figure 5.12: Screenshots displaying the simulation geometry used to verify the sibling set implementation and the notable events observed during its evolution. Observing figure 5.12a, on the left the solid element on the left represent the impactor and is given initial velocity $v = 15m/s$. The solid on the right represents the target and is not moving at the start of the simulation.

The use of a non-zero friction coefficient μ leads to the introduction of friction damping in the simulation, which translates to a progressive flattening of the deformation energy and to a decrease of the total system energy. In figure 5.13 the effect of damping can be clearly seen as the stepped reductions in the total system energy, which match exactly the timing of the observed impacts.

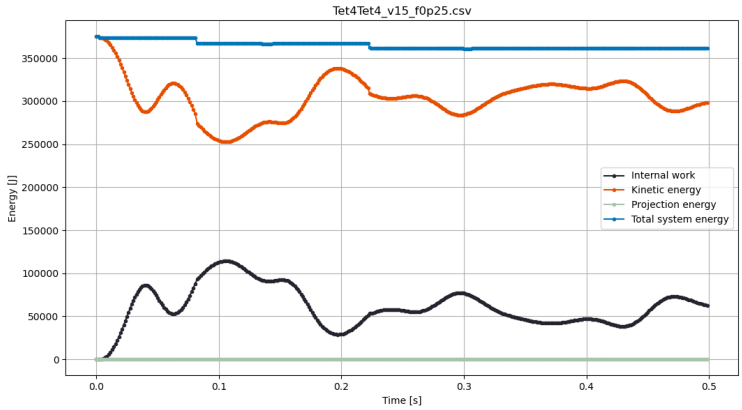


Figure 5.13: Verification of energy conservation over time. In blue is represented the summation of internal work (in black), kinetic energy (in orange) and projection energy (in green). Each dot represents a simulation output value. The stepped decreases in the total system energy can be interpreted as the effect of the friction damping created by the node-to-face impacts. This hypothesis is further reinforced by the fact that those discontinuities are coming from the kinetic energy od the system. Additionally, smaller oscillations of the total system energy are visible when compared to figure 5.8. This can also be explained with a stabilizing effect of the friction damping on the simulation.

5.6. Observations about the numerical results

Numerical experiments made on simple geometries and with a very limited amount of elements already highlighted some issues in the current formulation. Specifically, the same contact interaction has been solved multiple times in subsection 5.5.2. This happened because the current data structures cannot flag a node-to-face interaction for it to be successively solved as an edge-to-edge. Moreover, all edge-to-edge interactions are solved regardless of their interpenetration, given that the no explicit search is made. The reasons behind this are explained in section 4.1.

The solution to this issue would arise from the creation of the `Tetrahedron` and `Edge` data structures, from the use of bulk elements as the base mesh unit, and from the use of simplicial complexes described in subsection 4.4.1. Indeed, the `Triangle` and `Vertex` object do not possess any information about their connectivities and about their topological "parents" elements. This forces all additional information needed for the DG/CZM formulation (e.g., the sibling node sets) about the topological relationships in the FE mesh to be stored in dedicated data structures and, more importantly, to add many complicated criteria and clauses to avoid the detection of spurious contact or to discriminate between admissible and inadmissible intersection. Adding `Edge` and `Tetrahedron` classes and expanding the `Vertex` and `Triangle` with more topological information would greatly help in the detection of spurious contacts and in avoiding the solution of the same interaction with multiple routines. For example the sibling node sets are a node specific property, which are defined at the start of the simulation. Thus, they could be made a class attribute for `Vertex`.

Most importantly, the "sibling set" concept could be expanded to edges and faces. Indeed, in DG/CZM multiple edge stem from the same topological segment and multiple faces can stem from each topological face. This has already been exploited in subsection 5.1.1 for the creation of the internal contact faces. When combining this with the volume-based intersections search described in subsection 4.4.1, one could obtain a very powerful and efficient software architecture.

For example, when evaluating the intersection of two elements, one could use the simplicial boundary and coboundary concepts for calling their nodes and edges. The boundary of a simplex is the set of lower order elements belonging to it (i.e., all the segments defining the edge of a triangle), while its coboundary is the set of higher order elements to which the simplex belongs (i.e., the set of tetrahedra to which the triangle is a face). Imagine that two intersecting elements would be flagged for an edge-to-edge intersection: after determining the leader-follower relation, one could directly check for spurious edge-to-edge contacts due to compressive crack closure by calling the edge sets of the two elements and their edges' sibling sets.

6

Extending the DCR methodology with fracture onset and progression

In chapter 4 the Decomposition Contact Response (DCR) formulation and in its implementation have been analyzed, while in chapter 3 the theoretical background for the Discontinuous Galerkin/ Cohesive Zone Method (DG/CZM) framework and for the DCR algorithm have been given. Then, in chapter 5 the DCR algorithm has been adapted to FE problems using a Discontinuous Galerkin (DG) discretization. In this chapter the differences in the contact formulation between pure DG and DG/CZM are first discussed in section 6.1. Then, the sibling node set updating process is explained in section 6.2. In the following section 6.3 the differences in nodes to consider for velocity superposition and spurious contact detection are shown, and afterwards, the measures taken to account for cracked interfaces recontact are discussed in section 6.4. Similarly to the previous chapter, in section 6.5 simulations are shown for verification and qualitative validation purposes, along with a discussion about the limitations of the developed algorithm. Lastly, in section 6.6 some observations about the causes behind the unsatisfactory results of the numerical simulations are given.

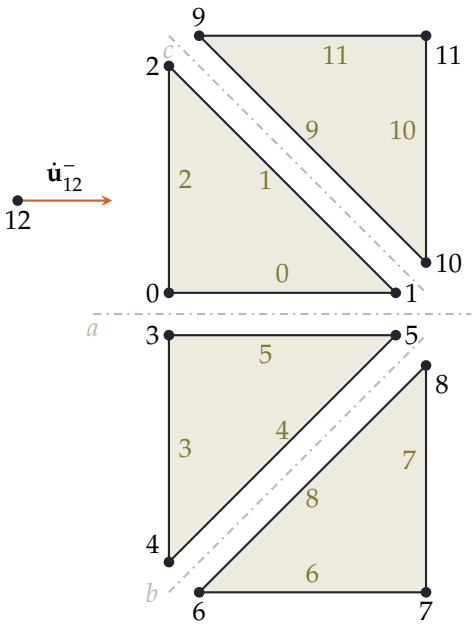
6.1. Coupling after fracture onset

In a "pure" DG formulation displacement compatibility is maintained by exchanging tractions at inter-element boundaries. This does not allow for fracture to onset and propagate in the mesh. Nonetheless, the exchanged tractions can be used to evaluate fracture criteria at inter-element boundaries. By adding cohesive elements to the interfaces meeting a given fracture criterion, one can add fracture onset and progression to the DG framework, expanding it into DG/CZM. A hybrid "intrinsic-extrinsic" approach is taken with the insertion of these cohesive elements at inter-element boundaries to maintain good stress wave propagation properties. The introduction of cohesive elements is a powerful method for fracture modelling, but creates new challenges in the coupling of DG/CZM the DCR algorithm.
Indeed, the DG/CZM formulation allows for any FE interface to separate and start a crack. This has been taken in consideration from the start of the thesis work, with the addition of internal faces in the contact mesh discussed in section 5.1. However, in the simulation of section 5.5 the internal faces have not taken part to the simulation. Indeed, having defined the sibling sets with the initial geometry, the internal faces are not detecting any contact from neighbouring elements, and therefore are effectively deactivated. The onset and evolution of fracture will thus change the composition of the sibling sets, with a progressive reduction of their size. In fact, considering an entirely cracked mesh, every node should possess an empty sibling set.
The fracture criterion is enforced at the DG/CZM interfaces quadrature points, as mentioned in section 3.1.2. Therefore, the status of each interface is not a binary value, but it can be closed, partially open or open. For simplicity, a DG interface is considered for the sibling sets update only once that all its quadrature points are open. For linear, tetrahedral FE three quadrature points are present on each side of an inter-element interface. Thus, six boolean values show the progression of fracture for each DG/CZM interface. To reduce these six values to a single boolean α_I for each I-th DG/CZM interface a mapping algorithm is employed. The chosen strategy is to update α_I only when all the quadrature

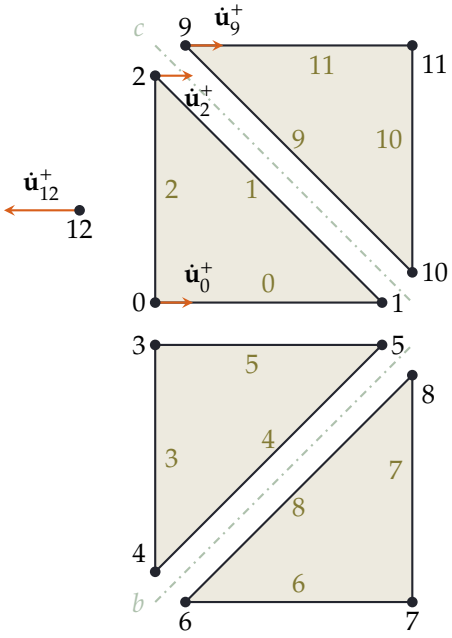
points on both sides meet the fracture initiation criterion.
Additionally, a cohesive law with an irreversible opening phase is chosen, meaning that once that separation process ends, the cohesive law is deactivated and the interactions between adjacent interfaces can only occur via contact. Therefore, the contact interaction corresponding to compressive closure of cracked interfaces is treated by in the DG/CZM formulation only until a critical separation δ_c value is reached.

6.2. Updating the sibling sets based on fracture evolution

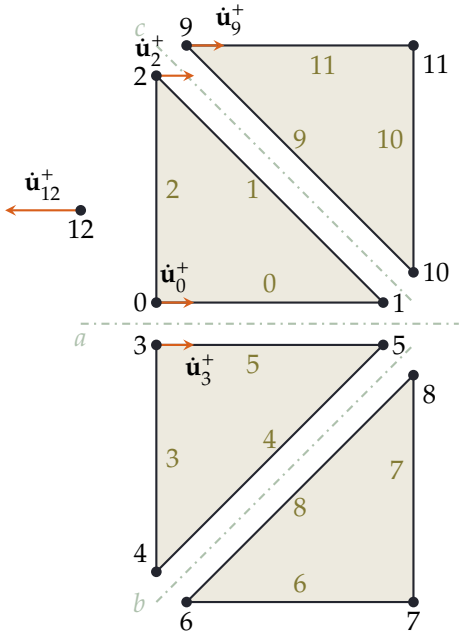
Before fracture onset, the sibling node sets remain constant in time, since the relations between the nodes do not change. However, after fracture the sibling sets introduced in section 5.2 evolve together with the status of the inter-element DG/CZM cohesive elements. For example, consider figure 6.1, where a 2D DG/CZM mesh of four triangles is impacted by a node. Assume that the material used for case 1 has a lower critical stress σ_c and that the impact velocity and mass are the same. In case 1 interface 0 is opened by the impact, while in case 2 is not. The post-impact velocities will be different, due to the sibling nodes sets changing with the opening of the DG/CZM interface.



(a) Pre-impact velocities: Only node 12 is moving with velocity $\dot{\mathbf{u}}_{12}^-$ and is set to impact on face 2. The dash-dotted light green lines represent inter-element DG/CZM interfaces a, b, c. The orange vector the node velocities.



(b) Case 1: velocities following the impact, cracked interface a. The impact has opened interface a, and thus it has been deactivated. Therefore, the sibling routine does not superimpose $\dot{\mathbf{u}}_0^+$ to node 3.



(c) Case 2: velocities following the impact, un-cracked interface. Interface a has not been opened and therefore the post-impact velocity is super-imposed on node 3.

Figure 6.1: Example of a single impact on a simplified DG/CZM mesh. A 2D mesh is utilized for simplicity, and thus the elements are 2D triangles and the contact mesh is composed of 1D segments. Nodes 12 is set to impact contact face 2 with velocity $\dot{u}_{12}^-$. All other nodes are not moving at the start of the simulation. In case 1 interface a cracks, while in case 2 the interface remains it does not. The different post-impact velocities are compared, along with the deactivation of interface a.

When updating the sibling sets, the status of each DG/CZM interface is assessed using the binary variable $\alpha \in [0, 1]$, where 0 means closed and 1 means open. Hence, DCR only handles fracture after a complete opening of the interface, while everything before is to be handled by the DG/CZM boundary integrals. The DG/CZM sibling set $S_a = S_a(t)$ of a given node a becomes therefore a function of simulation time. Indeed, for every time step in which a DG/CZM interface cracks, the set might

need to be updated. For example, if considering figure 6.1a, $S_1(t_i) = \{5, 8, 10\}$, while in figure 6.1b $S_1(t_{i+1}) = \{10\}$.
In figure 6.2 a flowchart is used to show the procedure implemented for updating the sibling sets, based on the evolution of the cracked interfaces pattern in the mesh. In the DG/CZM implementation of the sibling sets, at the start of each iteration, the vector with the sibling sets is emptied. Then, the code loops on all the DG/CZM inter-element interfaces. For each closed interface I, the nodes belonging to the corresponding contact mesh faces $f^+ = 2I$ and $f^- = 2I + 1$ are retrieved.
Given that the two adjacent faces belong to a closed DG/CZM interface and that a tetrahedral mesh is used, for each interface an interface node set N_I can be defined. N_I is composed of three node pairs, where each pair j^+, j^- is defined by $\mathbf{X}_{j^+} \sim \mathbf{X}_{j^-}$. This information is defined at the start of the simulation, and therefore a new variable is created to store it during the mesh assembly process described in subsection 5.1.1. This variable groups all node pairs for each interface. For example, $S_a^I = (ID_{2I}, ID_{2I+1})$ is the set of pair a of interface I and i are the node IDs. Using this variable, for each node pair (ID^+, ID^-) the respective sibling node sets are assembled as $S_{ID^+} = \{ID^-\}$ and $S_{ID^-} = \{ID^+\}$.
Then, each sibling set needs to be expanded based on the neighbouring closed DG/CZM interfaces. This is obtained with a second loop performed on the mesh nodes. For each node i in the mesh, its sibling set S_i is joined with the sibling set S_k of each node $k \in S_i$. The joining is made with repetitions.

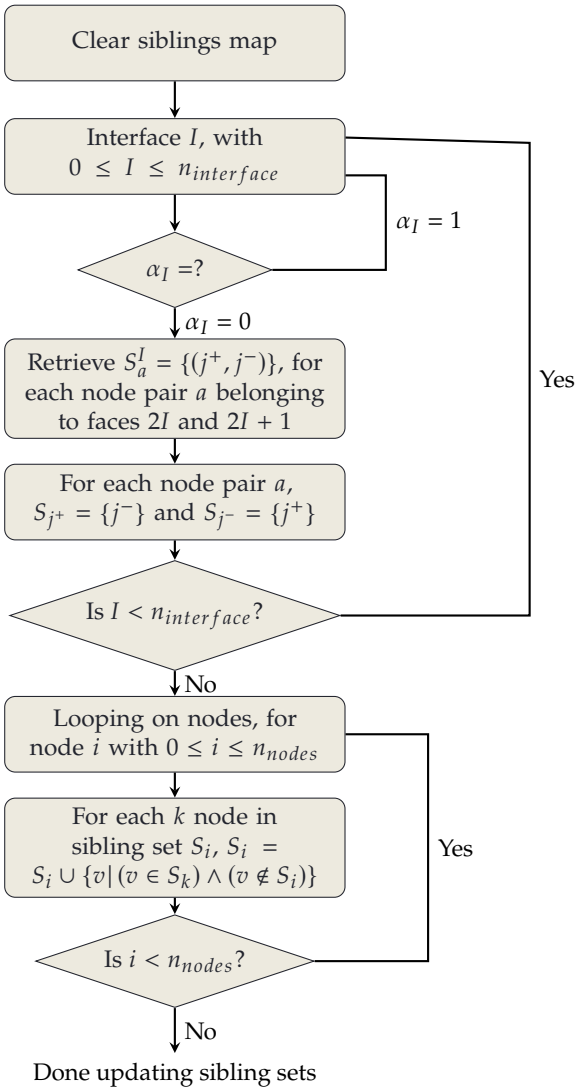


Figure 6.2: Flowchart showing the updating process of the sibling sets based on the evolution of the cracked interfaces pattern. After clearing the sets, a first loop on the DG/CZM interfaces is performed. For each interface, three adjacent node pairs exist. Each pair is defined based on the two nodes having similar coordinates ath the start of the simulation. During this loop, the sibling set of each node belonging to a closed DG/CZM interface is populated with its adjacent node. Then, a second loop is performed on the mesh nodes, where the sibling sets are expanded by joining themselves with all the sibling sets of their members.

The result of figure 6.2 applied to Case 1 ($\alpha_a = 1$) of figure 6.1 is shown in table 6.1. The modifications following the opening of interface a are clearly visible.

Node i	$S_i(t_i)$	$S_i(t_{i+1})$
0	3	$\emptyset$
1	5, 8, 10	10
2	9	9
3	0	$\emptyset$
4	6	6
5	1, 8, 10	8
6	4	4
7	$\emptyset$	$\emptyset$
8	1, 5, 10	5
9	2	2
10	1, 5, 8	1
11	$\emptyset$	$\emptyset$
12	$\emptyset$	$\emptyset$

Table 6.1: Example showing the update of the siblings sets in figure 6.1b following the opening of a crack along interface a. Node 0 and 3's sibling sets are emptied, while node 1 and 5 maintain respectively only the adjacent nodes on interfaces b and c.

6.3. Differentiating the sibling node sets

Before fracture onset, the same sibling node sets can be used for avoiding the detection of spurious contacts and for applying post-impact velocities. Once that some of the inter-element interfaces change status and open, two different sibling sets need to be used for each node. Indeed, once an interface is open the nodes to be excluded from the contact search are not the same as those to which the velocity should be superposed to.
For each node j on face i and interface I, a set $S_{j,s}$ is used for avoiding spurious contact detection and a set $S_{j,v}$ is used for velocities super-imposition. The former can be expressed as:

$$S_{j,s} = \begin{cases} \{ID_k, \forall k \mid k \in S_v \text{ with } v \in V_i\} & \text{if } ((\alpha_I = 1) \bigwedge (i \leq n_{int})) \bigcup (i > n_{int}) \\ \{ID_k, \forall k \mid k \in S_v \text{ with } v \in (V_{adj} \cup V_i)\} & \text{if } (\alpha_I = 0) \bigwedge (i \leq n_{int}) \end{cases} \tag{6.1}$$

Where V_{adj} is the set of vertices belonging to the adjacent face and V_i the set of vertices belonging to face i. This means that if the face belongs to a closed DG/CZM inter-element interface the impacting node is ignored if it belongs to the sibling set of either its own vertices or to the set of vertices of its adjacent face. If the corresponding DG/CZM interface is open or the face lies on the boundary of the mesh, only the sibling sets of its own vertices are considered.
The sibling set for velocity super-imposition can be expressed as:

$$S_{j,v} = \begin{cases} \emptyset & \text{if } (\alpha_I = 1) \bigwedge (i \leq n_{int}) \\ \{ID_k, \forall k \mid k \in S_v \text{ with } v \in (V_i \cup V_{follower})\} & \text{if } ((\alpha_I = 0) \bigwedge (i \leq n_{int})) \bigcup (i > n_{int}) \end{cases} \tag{6.2}$$

Thus, if the corresponding interface is open, the velocity is not copied. If the face is on the boundary or the corresponding DG/CZM interface is open, the velocity is copied to the leader's and follower's nodes and to the nodes in their sibling sets.

6.4. Accounting for recontact of cracked interfaces

The development of crack patterns in a DG/CZM mesh makes the likelihood of edge impacts grow considerably, when compared to a CG or un-cracked DG/CZM mesh. Indeed, a CG contact mesh is continuous, meaning that each contact mesh element will have neighbouring elements. Therefore, an edge-to-edge contact on a smooth, continuous CG mesh is likely to happen only on boundary elements, which have more than one face belonging to the contact mesh. In any other case, the edge-to-edge intersection would be solved as a node-to-face intersection for one of the neighbouring elements. Thus the assumption discussed in section 4.1 of edge-to-edge impacts happening only tangentially to the leader's triangle surface.
However, in a DG/CZM mesh with a developed crack pattern many elements could have more than

one face participating to contact computations. Thus, this assumption is not likely to hold. Moreover, all elements belonging to a crack tip or being close to those on a crack tip are likely to experience a compressive closure of the crack and thus impacts with a normal relative velocity. For example, consider figure 6.3.

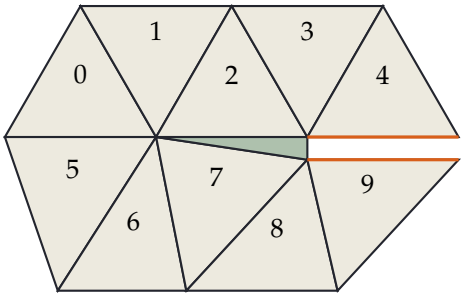


Figure 6.3: Graphical example of a 2D mesh with a developed crack pattern. In light green is shown the DG/CZM interface element between triangles 2 and 7 opening. In orange are highlighted the faces of triangles 4 and 9 likely to experience a normal impact. Indeed, the neighbouring active DG/CZM interface means that their movement is limited, and thus are unlikely to drift apart.

The DG/CZM formulation can deal with compressive recontact of the opened interfaces, but only for those faces which were already in contact at the start of the simulation. This could create inconsistencies in the solution, given that neighbouring faces could potentially end up being treated with different formulations. For example, consider the geometry in figure 6.4. This is the same geometry as figure 6.3, but at later stage, when the interface between 2 and 7 is completely cracked.

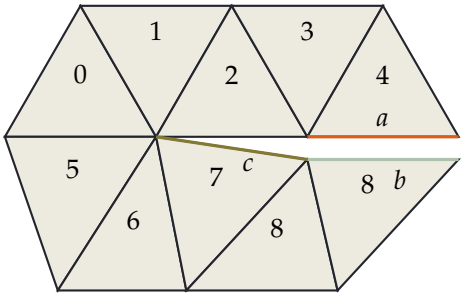


Figure 6.4: Graphical example of a 2D mesh with a developed crack pattern. In light blue and olive are shown edges b, c of respectively triangles 9 and 7. In orange is shown edge a of triangle 4. Depending on the kinematics of the system, edge a could impact either b or c. If the DG/CZM boundary integrals would be left on after the complete opening of the interfaces, this could lead to solving the same interaction with a different routine depending on the which face would be impacted. Thus, the integrals are switched off completely after the opening of their DG/CZM interface.

Assuming a compressive closure and a contact of the two crack edges, edge a is expected to hit the opposite crack interface. Depending on the system kinematics, it could hit edge b of element 9 or edge c of element 7. The former interaction would be solved by the DG/CZM interfaces, while the latter by the DCR algorithm. To avoid this possible conflict, the DG/CZM interface are completely switched off after their complete opening and all contact interactions are treated with DCR.
To avoid the detection of spurious contact which could arise from the edge-to-edge contacts with non-tangential relative velocities, the edge-to-edge routine is switched off in the case of a recontact between two internal contact mesh faces which were adjacent at the start of the simulation. Additionally, edge-to-edge interactions are also not solved if leader and follower edges are parallel within a numerical tolerance. The parallelism of the two edges is assessed by:

$$\frac{\mathbf{l}_1 - \mathbf{l}_0}{\|\mathbf{l}_1 - \mathbf{l}_0\|} \cdot \frac{\mathbf{f}_1 - \mathbf{f}_0}{\|\mathbf{f}_1 - \mathbf{f}_0\|} \leq \alpha_{\|} \tag{6.3}$$

Where $\mathbf{l}_0, \mathbf{l}_1$ and $\mathbf{f}_0, \mathbf{f}_1$ are respectively the leader and follower edge vertices' position vectors and $\alpha_{\|}$ a tolerance parameter.

6.5. Numerical verification and qualitative validation of the DG/CZM DCR algorithm

Similarly to what has been done for the "pure" DG interface material, the modifications introduced to deal with the full DG/CZM formulation have been tested on benchmarks. Given that the scope of this thesis is to test the feasibility of coupling DG/CZM and DCR algorithms, the geometries used in this simulations are simple and with a limited number of nodes. In this way, tracing each observed contact and explaining the source of errors and bugs is much easier.
The first geometry tested is a single tetrahedron impacting on a solid composed of two tetrahedra linked by a DG/CZM interface, in order to study the behaviour of the DG/CZM interface and to assess numerically the interaction between DCR and DG/CZM. This is done subsection 6.5.1.
Afterwards, a second tetrahedron is added to the impactor and similar analyses are performed. Instead of testing different impact parameters, the focus is set to different relative orientations of the solids. Thus, in subsection 6.5.2 the impactor is set to hit the target with its edge. Then, in subsection 6.5.3 the impactor hits with a node the target and its DG/CZM interface is normal to the impact direction. Lastly, in subsection 6.5.4 the DG/CZM interface sis parallel to the impact direction. The algorithm shows some limitations, the sources of which are then discussed.

6.5.1. Tetrahedron on solid impact

The first simulation used to test the newly developed algorithm has been the simplest geometry allowed by the use of tetrahedral elements and of the DG/CZM formulation: the impact of one element on two other element linked with a DG/CZM interface. One bulk and one interface material are used in the mesh. The material model chosen is neo-hookean [66] and its material properties are shown in table 6.2. The Young modulus of the material is increased by an order of magnitude compared to section 5.5 to obtain a lower compliance and focus more on the fracture onset.

Property	Value	Unit
E	5	MPa
ρ	5000	kg/m^3
ν	0.3	-

(a) Bulk material properties for neo-hookean material. The parameters are: Young's modulus E, mass density ρ and Poisson's coefficient ν.

Property	Value	Unit
β_s	100	-
σ_c	$9.4e^{-3}$	MPa
G_c	50	J/m^2
γ	1.5	-
μ	0	-

(b) DG/CZM interface material properties. The parameters are: DG/CZM stabilization coefficient β_s; critical stress for fracture initiation σ_c; energy release rate G_c, the normal/tangential weighting ratio γ and the DG/CZM friction coefficient μ.

Table 6.2: Material properties utilized in the impact simulation between a tetrahedron and a solid.

The values assigned to the material properties do not represent any existing material, but are instead tuned for fracture to initiate due to stress wave propagation instead of directly at the time of the impact event. In table 6.3 other simulation parameters are given, and in figure 6.5 the starting geometry of the simulation is also shown.

Property	Value	Unit
v	15	m/s^2
f_t	0.8	-
e	2	-
μ	0	-
ε_{tang}	$1e^{-3}$	-
$\alpha_{\parallel}$	$\cos(5^\circ)$	-
ε_{DG}	$1e^{-12}$	-

Table 6.3: Simulation parameters: initial velocity v given to the single tetrahedra in the $+X$ direction; time factor f_t, which is used as a safety factor on the DG/CZM stable time step; coefficient of restitution e, which is set to obtain a fully elastic response; Couloumb's friction coefficient μ; the tangential tolerance coefficient ε_{tang} used to define the bounding boxes used for the search of inadmissible intersections; $\alpha_{\parallel}$ numerical tolerance to determine if two edges are parallel; and ε_{DG} numerical tolerance to avoid detecting DG numerical inter-element interface intersections.

A higher number of contact interactions are expected by activating the opening of the DG/CZM interface cohesive elements. Therefore, this analysis will focus on the stress propagation and on the energy conservation or lack thereof. The stresses are sampled at node 11, shown in figure 6.5. The sampled stresses are σ_{11}, σ_{33}, which are shown as a function of time. To the side of the stress plots are shown screenshots depicting the simulation kinematics.

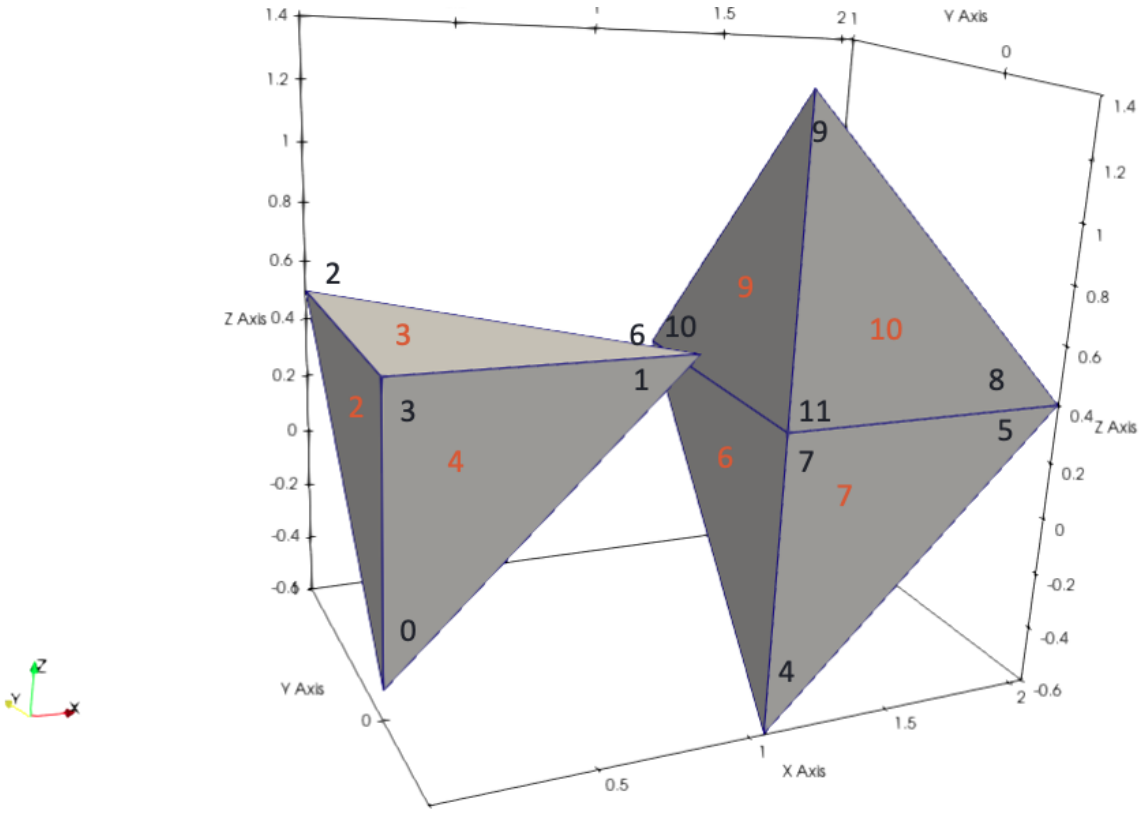


Figure 6.5: Initial configuration for the tetrahedra on solid impact simulation. On the left is shown the impactor and on the right are shown the targets. The impactor is set to travel in $+X$ direction with velocity v as initial conditions. In dark blue are indicated the node IDs and in orange the face IDs. Due to the geometry of the simulation, the internal faces 0 and 1, which belong respectively to bottom and top target tetrahedra, are not visible. Node 1 of the impactor will hit face 9 of the target tetrahedron on the top, and the simulation is run until the three tetrahedra stop interacting.

After starting the simulation, the first relevant event is observed at $t = 3.2\ ms$ in figure 6.6. The impactor hits the target and the node-to-face contact routine is applied for the first time. The stresses rise and an elastic stress wave is observed with the growth of the stress magnitudes happening until $t \sim 15.8\ ms$.
At this instant the DG/CZM interface is starting to open partially at the quadrature points on the opposite side of the sampled node. The stresses decrease as the stress wave travels through the FE, inverting the sign of σ_{33} at $t \sim 24.8\ ms$. The σ_{33} start to rise in tension until $t \sim 26.2\ ms$, when the traction needed to completely open the DG/CZM interface is reached in figure figure 6.7. The discontinuity observed at this point arises from the DCR algorithm tackling the contacts between faces 9 and 6.
In the meantime, the impactor reaches its maximum compression and starts expanding again. This leads to a second impact of node 1 with face 9 in figure figure 6.8. The hit creates the cuspid in the stress plots at $t = 37.3\ ms$. Indeed, the tetrahedra start contracting again and thus the compressive stresses visible until $t \sim 50.8\ ms$. At this point expansion starts again. Given that, the DG/CZM is open at the time of this second impact, the post-impact kinematics are different: the original impactor starts rotating while the upper target start drifting away in $+X$ direction.
The rotation on the target leads to a third impact at $t = 140.8\ ms$ in figure figure 6.9. The edge with nodes 0,1 of the impactor hits the edge with nodes 4,6 of the bottom target tetrahedron. No stress wave is present in the plots because the DG/CZM interface is open and the stress is sampled at node 11 on

the upper target tetrahedron.

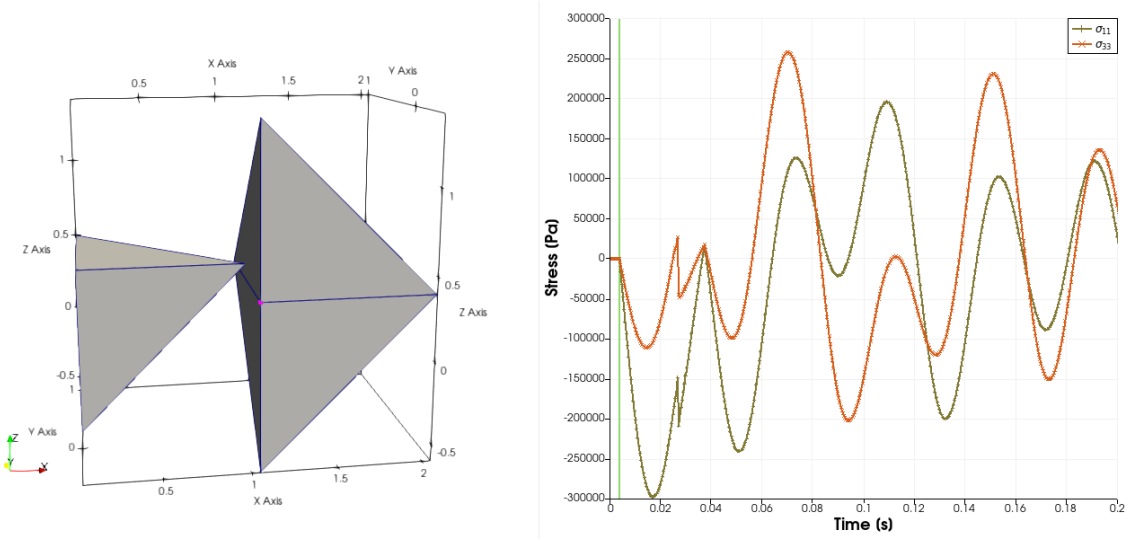


Figure 6.6: System configuration and σ_{11}, σ_{33} in the top target tetrahedron at $t = 3.2\ ms$. Node 1 impacts face 9 of the upper target tetrahedron, and both stresses start to rise in magnitude.

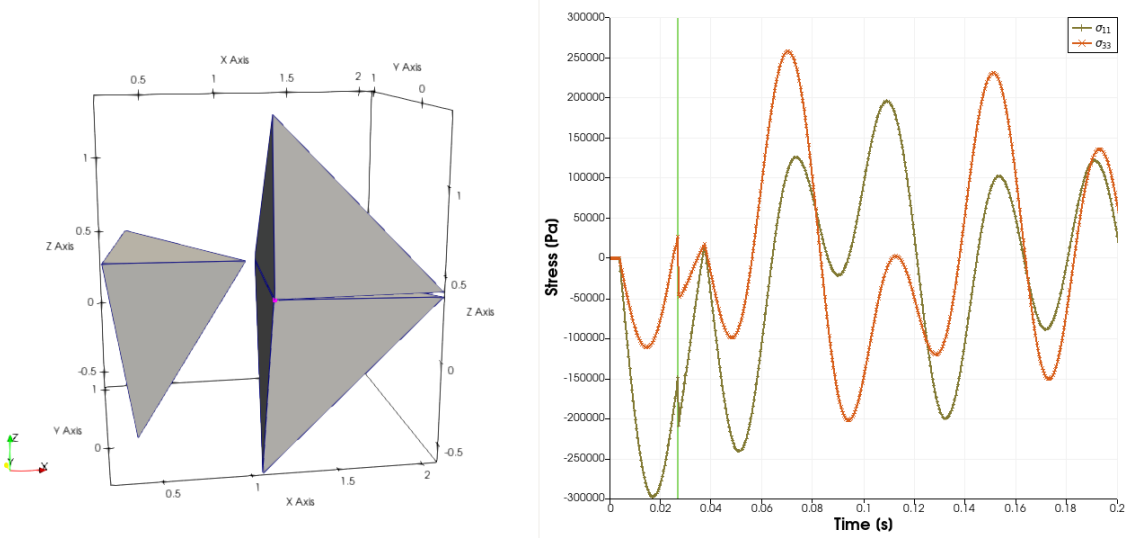


Figure 6.7: System configuration and σ_{11}, σ_{33} in the top target tetrahedron at $t = 26.2\ ms$. The interface between the two target tetrahedra is completely opened by the tensile stress created by the interaction of the returning elastic stress waves created by the interaction of the impact stress wave with the element boundaries.

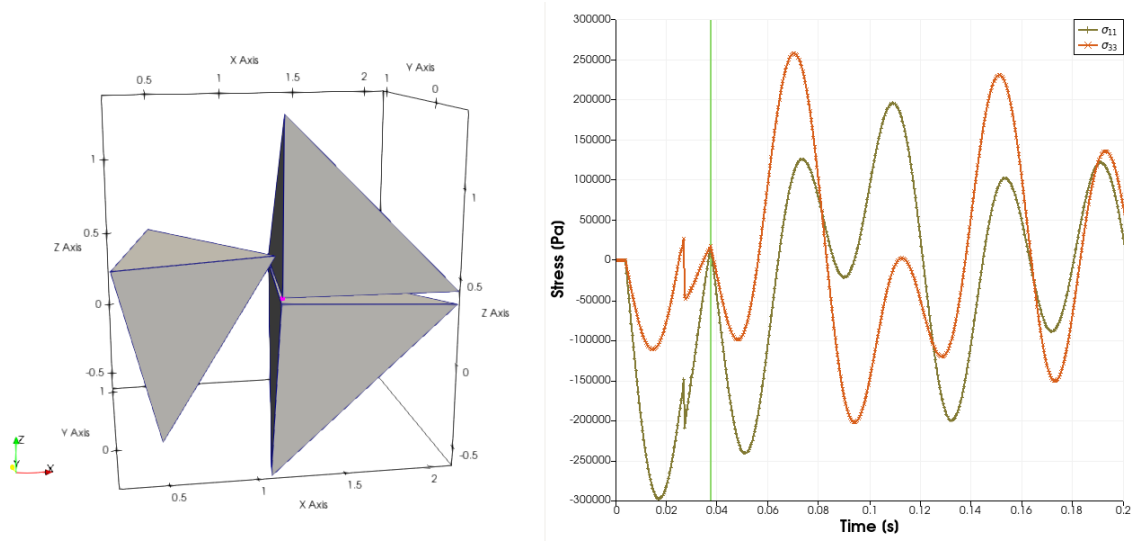


Figure 6.8: System configuration and σ_{11}, σ_{33} in the top target tetrahedron at $t = 37.3\ ms$. The low stiffness of the material makes the impactor to bounce back and hit again face 9 with node 1. Therefore, a new inversion in the slope of the stress is observed, since the target tetrahedron starts compressing again.

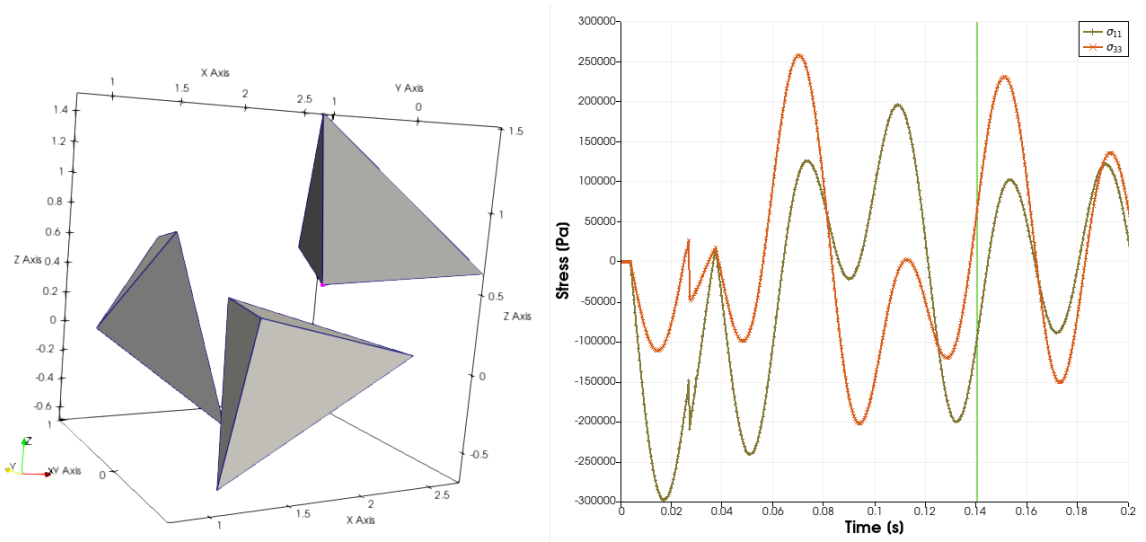


Figure 6.9: System configuration and σ_{11}, σ_{33} in the top target tetrahedron at $t = 140.8\ ms$. The edge with nodes 0,1 of the impactor hits the edge with nodes 4,6 of the bottom target tetrahedron. Since the two target tetrahedra are detached, no stress inversion or discontinuity in stress slope is observed in the plot. This impact is due to the asymmetrical shape of the impacting tetrahedra, which created a rotation after the second impact.

Lastly, system energies have been inspected to assess the magnitude of the artificial projection energy introduced by the DCR algorithm when the DG/CZM interfaces are activated. In figure 6.10 the system's energies are shown.

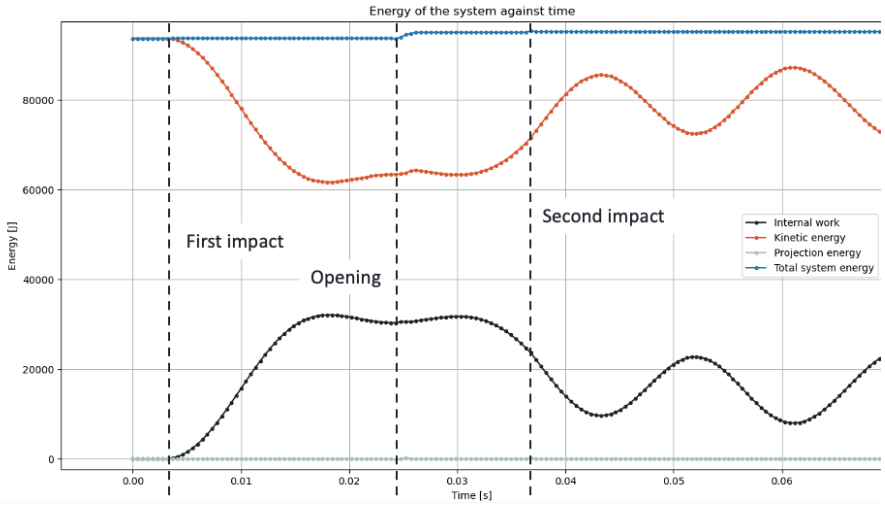


Figure 6.10: Verification of energy conservation over time. The first and second impacts and the opening of the DG/CZM interface are highlighted, while the third impact is omitted from this picture to obtain more clarity. it is clearly shown how the projection energy introduced by each impact modelled with the DCR algorithm is negligible when compared to the total energy of the system.

For better clarity the total energy of the system is shown in figure 6.11 with a smaller interval on the y-axis.

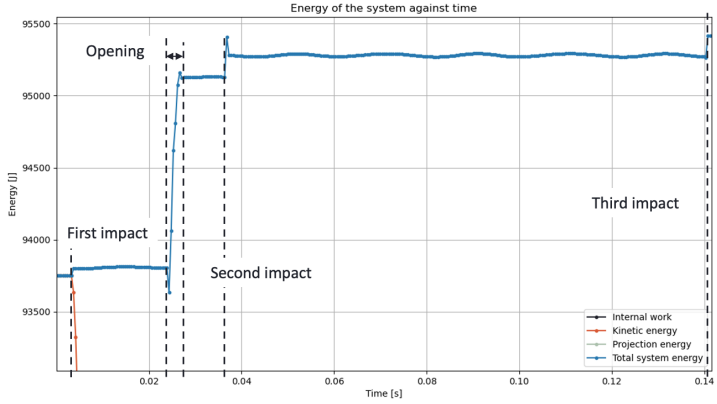


Figure 6.11: Verification of energy conservation over time. The first, second and third impacts and the opening of the DG/CZM interface are highlighted. The increases in energy due to the removal of the element intersection can be clearly seen, together with the effect of the two internal faces remaining almost superposed after the opening of the DG/CZM interface and the removal of the intersection accumulated during the opening phase.

Four main increases in projection energy can be seen, respectively at $t_{first} = 3.2\ ms, t_{opening} \sim 24.8\ ms, t_{second} = 37.3\ ms, t_{third} = 140.8\ ms$. They respectively are due to the first impact, the contact interactions detected after the opening of the target's DG/CZM interface, the second impact and the third impact. In table 6.4 are collected, from left to right, the system energies before and after the each, the net energy introduced and the percentage increase with regard to the initial system energy:

$$E_0 = \frac{m_{impactor} v^2}{2} = 93750\ J.$$

It is clear how the most notable increase is due to the multiple consecutive contact interactions detected during the time between opening and second impact. Indeed, in this time interval the two faces sharing the DG/CZM remain basically superposed and continuously interact until a more stable configuration is reached at $t \sim 27.1\ ms$. This lead to the highest relative increase in energy $\Delta E_{tot} = 1.44\%$. Overall the increase remains under 2% of the initial energy, which is considered an acceptable result even if sampled on a short time interval and for a limited amount of interactions. Additionally, the unusually element's

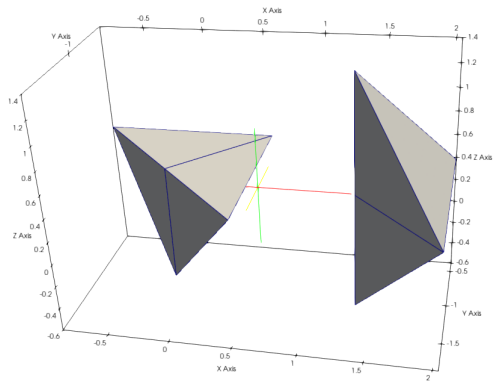
size might have influenced the results. Hence, in the next sections a more complicated geometry will be inspected.

Event	E_{tot}^{-} [J]	E_{tot}^{+} [J]	ΔE_{tot} [J]	ΔE_{tot} [%]	ΔE_{tot}^{0} [%]
First impact	93750	93799	49	0.05%	0.05%
Opening	93805	95157	1352	1.44%	1.50%
Second impact	95127	95404	277	0.29%	1.76%
Third Impact	95270	95414	144	0.15%	1.77%

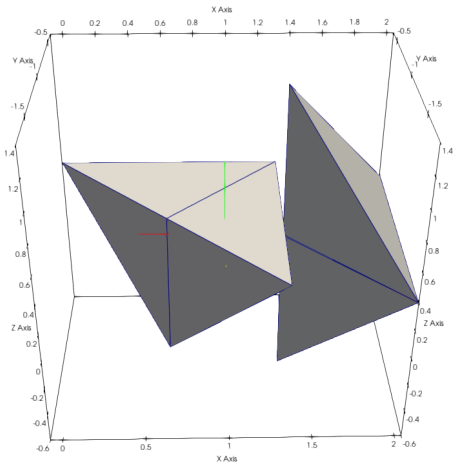
Table 6.4: Increases in total system energy due to the DCR removal of intersections. The values shown are: pre-event energy E_{tot}^{-}, post-event energy E_{tot}^{+}, changed in energy E_{tot}^{-}, percentage change in energy and percentage change in energy w.r.t the initial energy. The change is well within 2% at the end of the simulation, which is considered a satisfactory result, even if measured on a very low time interval.

6.5.2. Edge-to-edge impact between two solids

Following the first verification for stress wave propagation, the coupling between fracture and contact has been tested with more complex geometries. Two solids, each composed of two elements, are impacting each other with different relative orientations.
The material properties of this simulation are the same as those shown in table 6.2a and table 6.2b. The simulation parameters are also left unchanged when compared to table 6.3. The initial geometry of the system is shown in figure 6.12a and in figure 6.12b is depicted the initial edge-to-edge impact at $t = 0.054\ s$.



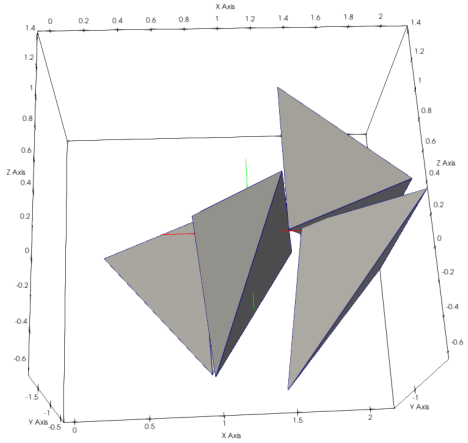
(a) Initial geometry for the solid on solid edge impact case at $t = 0\ s$. On the left the impactor is visible and on the right the target.



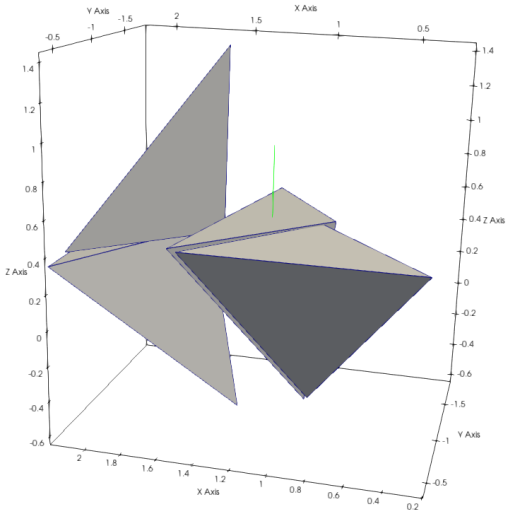
(b) First contact for the solid on solid edge impact case at $t = 0.054\ s$. The impact is set to be an edge-to-edge interaction between boundary faces, in order to avoid the creation of numerical energy at the start of the simulation which could hinder further observations on the evolution of the system.

Figure 6.12: Initial geometry and first contact for the solid on solid edge impact case. At the center of figure 6.12a the reference system is visible, with the X-axis in red pointing right, the Y-axis pointing left in yellow and the Z-axis pointing up in green.

After this impact, the stress waves travel through the two solids, refract on the element boundaries and start the opening of the DG/CZM interfaces. The cohesive elements damage progresses until $t = 0.073\ s$, when the target's DG/CZM interface is completely open and the DCR algorithm is activated for the target's internal faces. This instant is shown in figure 6.13a. The impactor interface opens at $t = 0.075\ s$ and is shown in figure 6.13b. A first conflict between the two formulations is visible in these two screenshots: indeed, the DG/CZM boundary integrals evaluate the intersection of the FEs sharing an interface at quadrature points, which are not located at the vertices of the triangles. Thus, they allow a small intersection of the faces, as long as it is zero at the quadrature point. Thus, this intersection accumulates before the DCR routine is activated. After its activation, the DCR routines continue to ignore these intersection since their gap vector is bigger than the time step's maximum displacement.



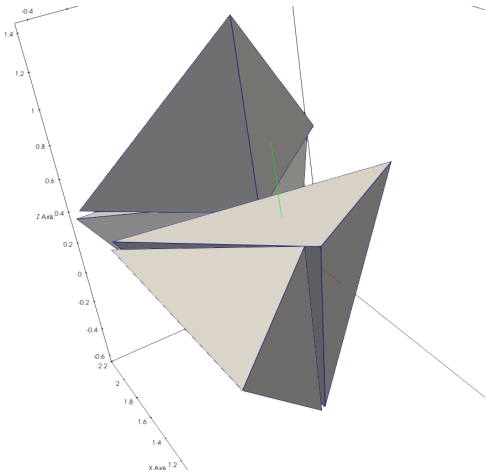
(a) Opening of the target's DG/CZM interface at $t = 0.073\ s$ for the solid on solid edge impact case. The intersection visible on the right is due to the DG/CZM interface evaluating the intersection magnitude at quadrature points which are not superposed to the vertices.



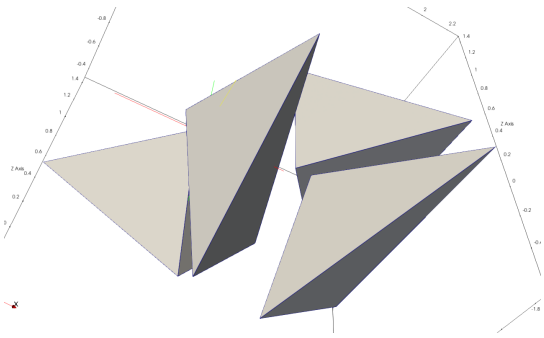
(b) Opening of the impactor's DG/CZM interface at $t = 0.075\ s$ for the solid on solid edge impact case. Similarly to figure 6.13a, the intersection visible on the right is due to the DG/CZM interface evaluating the intersection magnitude at quadrature points which are not superposed to the vertices. The view is rotated by 90° around the X-axis in clockwise direction and by 180° in counter-clockwise direction around the Z-axis.

Figure 6.13: Screenshots showing the instants when the last DG/CZM quadrature points reaches the critical separation and the inter-element interfaces open completely. In the left picture is shown the opening of the target interface, while in the right image the opening of the impactor interface.

This phenomenon leads to what is seen in figure 6.14a and figure 6.14b: the intersections are removed when the kinematics of the system bring the intersecting nodes closer to the intersected face, and a high quantity of numerical energy is injected in the system, violating the energy conservation principle.



(a) Removal of the intersection created by the impactor's DG/CZM interface opening at $t = 0.076\ s$ for the solid on solid edge impact case. The intersection of figure 6.13b is removed by a edge-to-edge interaction, as shown by the node of the left element being in contact with the edge of the right element. This also creates the condition for the two FEs to continue interact in the following time steps.



(b) Removal of the intersection created by the target's DG/CZM interface opening at $t = 0.083\ s$ for the solid on solid edge impact case. The intersection of figure 6.13a is removed by a node-to-face interaction. The large magnitude of the intersection leads to a noticeable injection in the kinetic energy system, visible on in figure 6.15.

Figure 6.14: Screenshots showing the instants when the intersections accumulated during the opening process of DG/CZM inter-element interfaces are removed. This removal created the numerical injection of kinetic energy in the system visible in figure 6.15.

The numerical energy injection is visible in the figure 6.15, where the system energy is plotted against time. The numerical increases of the kinetic energy are visible, together with the cuspid in the

deformation energy created by the target's intersection removal.

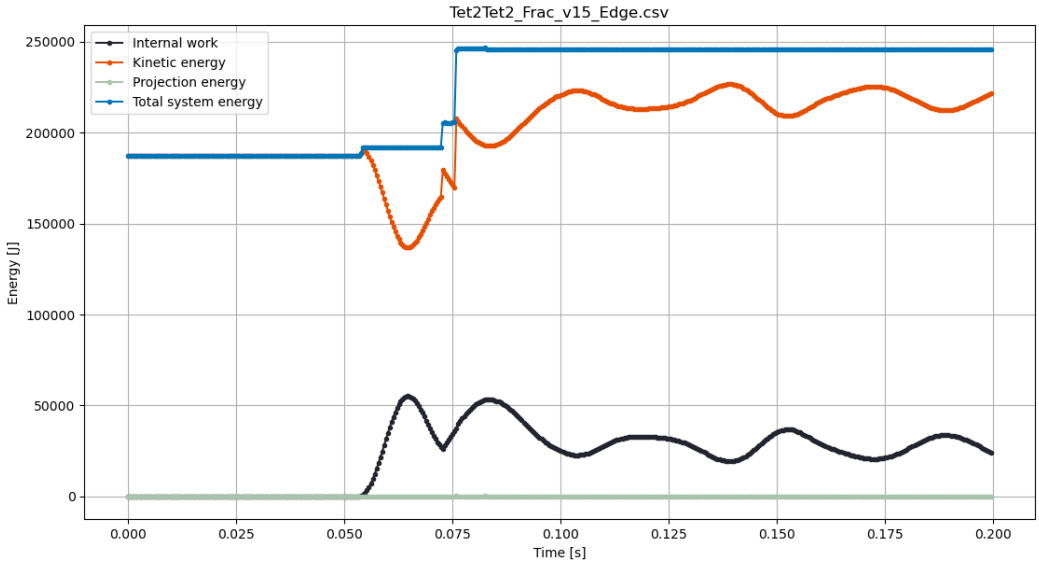
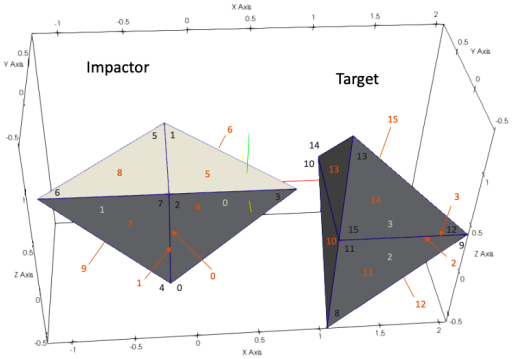


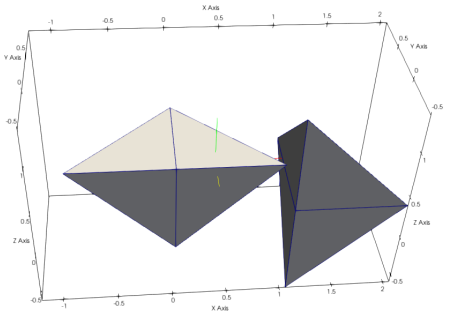
Figure 6.15: Verification of energy conservation over time. In blue is represented the summation of internal work (in black), kinetic energy (in orange) and projection energy (in green). Each dot represents a simulation output value. Three non-physical step-wise increases are visible in the total system energy. The left-most corresponds to the first edge-to-edge interaction, which is known to create small injections of numerical energy in the system due to the limitations discussed in section 4.1. Then, the next two discontinuities in the kinetic energy are corresponding to the interactions between the just activated internal faces and to the removal of the intersection accumulated during the opening process, when the interpenetrations are handled by the DG/CZM interface elements.

6.5.3. Node-to-face impact between two solids, impactor's interface normal to initial velocity

The simulation in subsection 6.5.2 has been run with the same parameters and a different impactor orientation. The bulk and interface material properties are also left unchanged from the previous simulation. The DG/CZM interface has been oriented normal to the initial velocity. In figure 6.16a the initial geometry is shown. The nodes', contact faces' and FEs' IDs are added to the image for an easier description of the simulation evolution in time. In figure 6.16b the first node-to-face impact of node 3 on face 13 is shown.



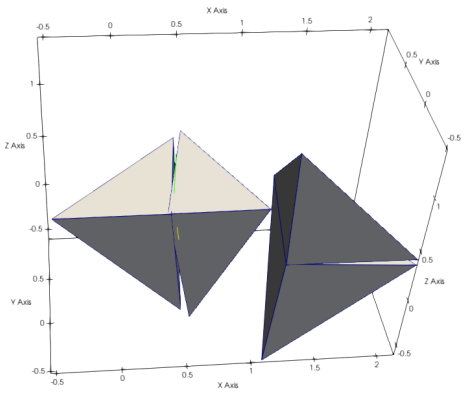
(a) Initial geometry for the solid on solid impact with a normal DG/CZM interface case at $t = 0\ s$. On the left is visible the impactor and on the right the target. In dark blue are given the node IDs, in orange the contact elements IDs and in light green the FEs IDs.



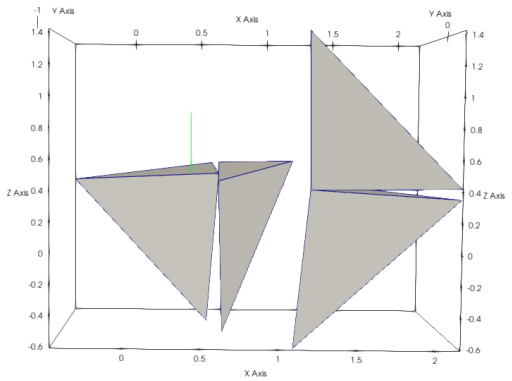
(b) First contact for the solid on solid impact with a normal DG/CZM interface case at $t = 0.018\ s$. The impact is set to be a node-to-face interaction between node 3 and triangle 13.

Figure 6.16: Initial geometry and first contact for the solid on solid edge impact case. At the center of figure 6.16a the reference system is visible, with the X-axis in red pointing right, the Y-axis pointing left in yellow and the Z-axis pointing up in green.

After the first hit, the stress waves start traveling through the two solids and after bouncing on the element boundaries, they start opening the DG/CZM interfaces. A second hit is observed for node 3 on face 13 at $t = 0.045\ s$, thus creating a second elastic wave in the impactor. This hit leads to the opening of the impactor's DG/CZM interface at $t = 0.051\ s$. The opening, combined with element 1 pushing on element 0 due to its inertia creates multiple consecutive interactions between the two, leading to the creation of a high amount of numerical energy. This numerical energy mostly comes from the post-impact velocity imposed on the nodes of faces 0 and 1. The instant of the opening of the impactor is visible in figure 6.17a and that of the target in figure 6.17b.



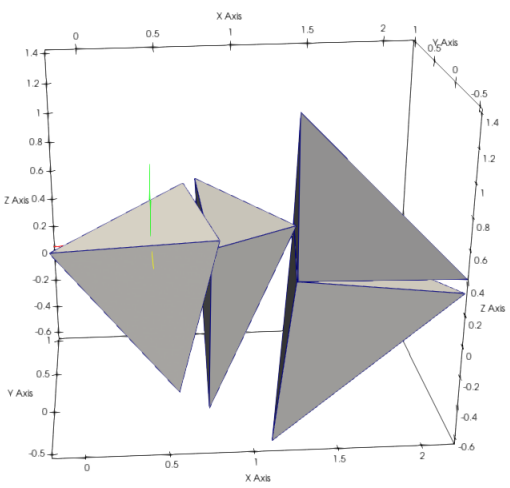
(a) Opening of the impactor's DG/CZM interface at $t = 0.051\ s$ for the solid on solid impact with a normal DG/CZM interface case. The intersection between elements 0 and 1 is clearly visible on the left. On the other hand, is also shown how the target's interface does not show signs of accumulated intersection at this point



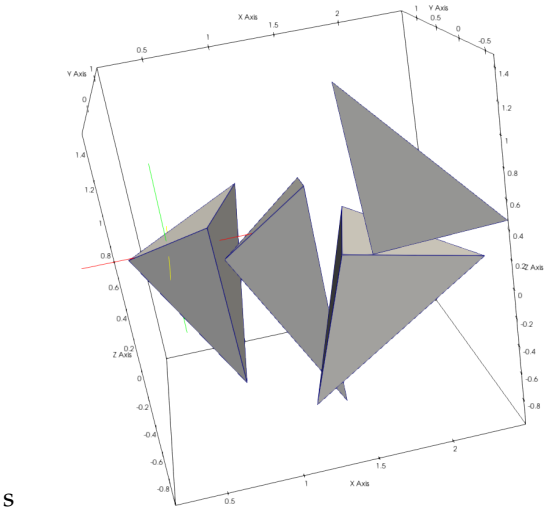
(b) Opening of the target's DG/CZM interface at $t = 0.056\ s$ for the solid on solid impact with a normal DG/CZM interface case. No accumulated intersection is visible for the target's DG/CZM interface, and thus no numerical energy is created. On the left of the picture is also visible that at this point the intersection has been completely removed from the impactor's interface, imposing a significant compression on element 0.

Figure 6.17: Screenshots showing the instants when the last DG/CZM quadrature points reaches the critical separation threshold and the inter-element interfaces open completely. Additionally, on the left of figure 6.17b the highly compressed state of element 0 following the complete removal of its accumulated intersection is shown.

On the left of figure 6.17b the state of element 0 after the removal of its accumulated intersection with element 1 is also shown. This element is highly compressed, and is thus storing elastic energy. Moreover, its nodes have been also accelerated by the continuous contact interaction registered between faces 0 and 1. Thus, when node 3 ends up hitting face 13 at $t = 0.062\ s$, element 3 is also accelerated and starts compressing. This event is shown in figure 6.18a. Then, element 3 begins expanding back and traps element 0 between itself and element 1, which is still traveling forward. This leads to a series of contact interaction which end up in three step-wise increments of the system's kinetic energy. Indeed, element 1 starts spinning counter-clockwise around its baricentral axis parallel to the Z-axis and transforms its accumulated elastic energy into kinetic energy. Figure 6.18b shows one of the last impacts between element 0 and element 1 at $t = 0.082\ s$.



(a) Impact of the compressed element 0 with element 3 at $t = 0.062\ s$ for the solid on solid impact with a normal DG/CZM interface case.



S

(b) Impact of element 0 with element 1 at $t = 0.082\ s$ for the solid on solid impact with a normal DG/CZM interface case. This impact is happening towards the end of a series of interactions which increased the kinetic energy of element 0 without allowing it to release the stored elastic energy created by the removal of contact intersection.

Figure 6.18: Screenshots showing the sequence of events leading to element 0 being extremely compressed and with high node velocities. These two combined factors lead to the creation of a high amount of numerical energy in the system, which then makes the simulation to violate the energy conservation principle.

The events explained in this paragraph can be all recognized in figure 6.19, where the system energy is plotted against time. Given the high amount of events mentioned, the main ones are shown on the picture with annotations. The first two node-to-face hits show no increase in the system energy. Then, the opening of the impactor's DG/CZM interface increases the system energy to around 5 times the initial value, with a step-wise increment visible for $t \sim 0.05\ s$. The start of the compression of element 0 is visible as the discontinuity in the plots of kinetic and elastic energy and is indicated as "spring hit" in the figure. Lastly, the release of the stored numerical energy is visible as the last three step-wise increments in the kinetic energy. Each increment corresponds to a hit between element 0 and one of the other elements in the simulation, which at this stage are all independent from each other. With each hit, element 0 is accelerated, and thus a higher kinetic energy increase is created with each successive impact.

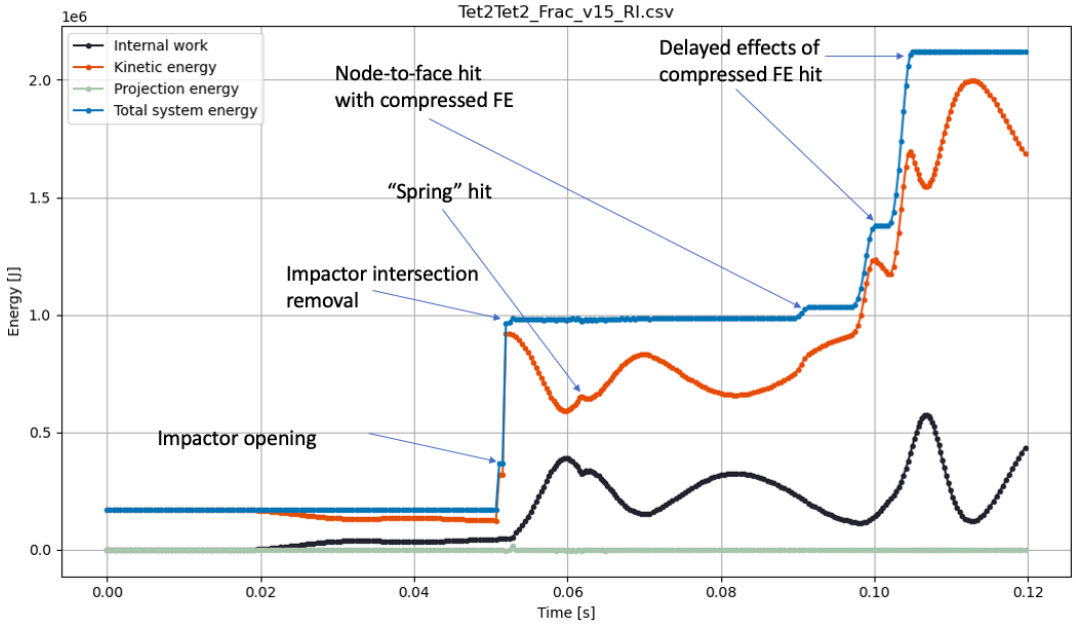
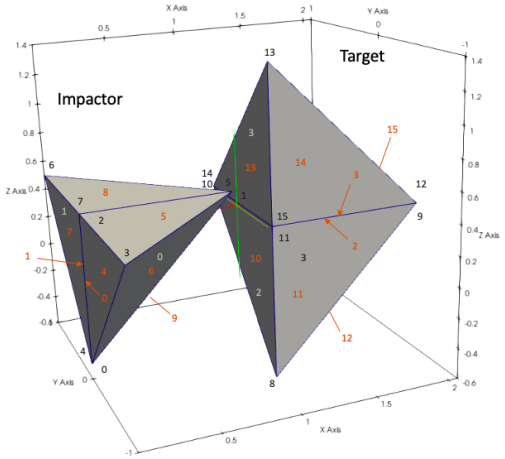


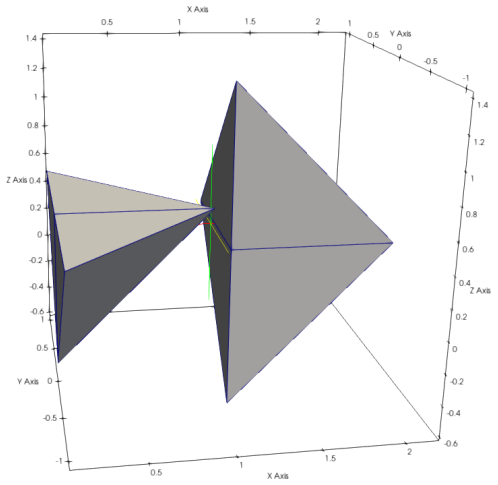
Figure 6.19: Verification of energy conservation over time for the impact between two solids, with the impactor oriented with its DG/CZM interface normal to its initial velocity. In blue is represented the summation of internal work (in black), kinetic energy (in orange) and projection energy (in green). Each dot represents a simulation output value. Three major non-physical increases are visible for the total system energy. The left-most corresponds to the removal of the intersection for the impactor DG/CZM interface. This happens with a number of interactions between faces 0 and 1, which explains the increase in kinetic energy. Indeed, each successive interaction uses the previous post-impact velocities as input values, and thus a non-physical injection of numerical energy in the system is observed. After the interaction between faces 0 and 1, element 0 is highly compressed as the intersection removal oscillations end at $t \sim 0.062\ s$. At $t \sim 0.092\ s$ the compressed element 0 hits element 2, leading to the release of its stored elastic energy and to the next two step-wise increases in the system's kinetic energy.

6.5.4. Node-to-face impact between two solids, impactor's interface parallel to initial velocity

The last solid on solid impact simulation is intended to to assess how the algorithm performs when the DG/CZM interface is oriented parallel to the initial velocity. This means that both the elements of the impactor will hit the target's face at the same time. The initial geometry and the first impact are shown respectively in figure 6.20a and figure 6.20b. The material properties are the same as those found in table 6.2a and table 6.2b, and the simulation parameters are shown in table 6.3.



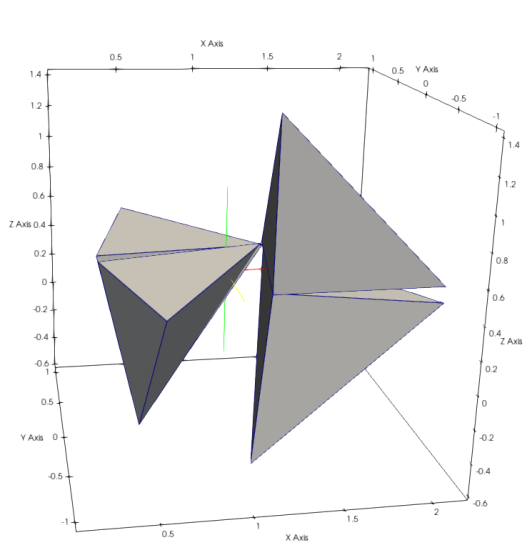
(a) Initial geometry for the solid on solid impact with a parallel DG/CZM interface case at $t = 0\ s$. On the left is visible the impactor and on the right the target. In dark blue are given the node IDs, in orange the contact elements IDs and in light green the FEs IDs.



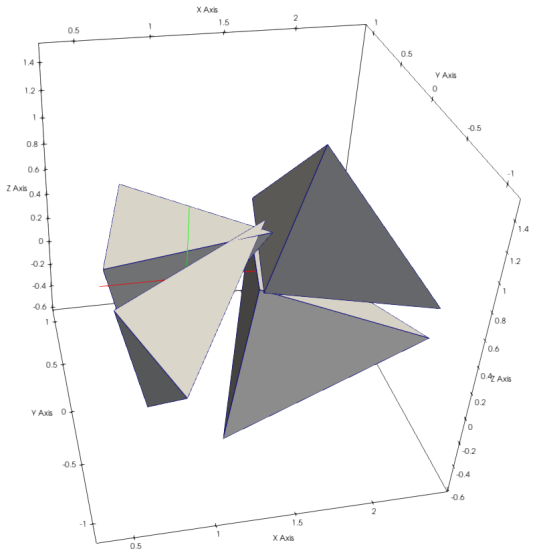
(b) First contact for the solid on solid impact with a normal DG/CZM interface case at $t = 0.004\ s$. The impact is set to be a node-to-face interaction between node 1 and 5 and triangle 13.

Figure 6.20: Initial geometry and first contact for the solid on solid edge impact case. At the center of figure 6.20a the reference system is visible, with the X-axis in red pointing right, the Y-axis pointing left in yellow and the Z-axis pointing up in green.

The interface of the target is the first one to open at $t = 0.035\ s$. There is no considerable accumulated intersection at the instant of the opening, as visible in figure 6.21a. Nonetheless, the interactions between face 2 and 3 are solved multiple times, until $t = 0.038\ s$, leading to the injection of numerical energy in the system, as visible with the first step-wise increment in kinetic energy in figure 6.22. Then, at $t = 0.066\ s$, the impactor's DG/CZM interface opens. As visible in figure 6.21b, the intersection is substantial. This leads to the computation of very large post-impact velocities in the subsequent time steps. These velocities start a cascading feedback loop which lead to the simulation diverging over the next two time steps, as visible on th right of figure 6.22.



(a) Opening of the target's DG/CZM interface at $t = 0.035\ s$ for the solid on solid edge impact case. No large intersection is visible in the target's DG/CZM interface, while the impactor's interface has only just started to be opened.



(b) Opening of the impactor's DG/CZM interface at $t = 0.066\ s$ for the solid on solid edge impact case. The intersection visible between the element tips on the right of the impactor is due to the DG/CZM interface evaluating the intersection magnitude at quadrature points which are not superposed to the vertices.

Figure 6.21: Paraview screenshots showing the instants when the last DG/CZM quadrature points reaches the critical separation and the inter-element interfaces open completely. The large intersection accumulated during the damaging and opening of the DG/CZM interface cohesive element between elements 0 and 1 is visible in figure 6.21b.

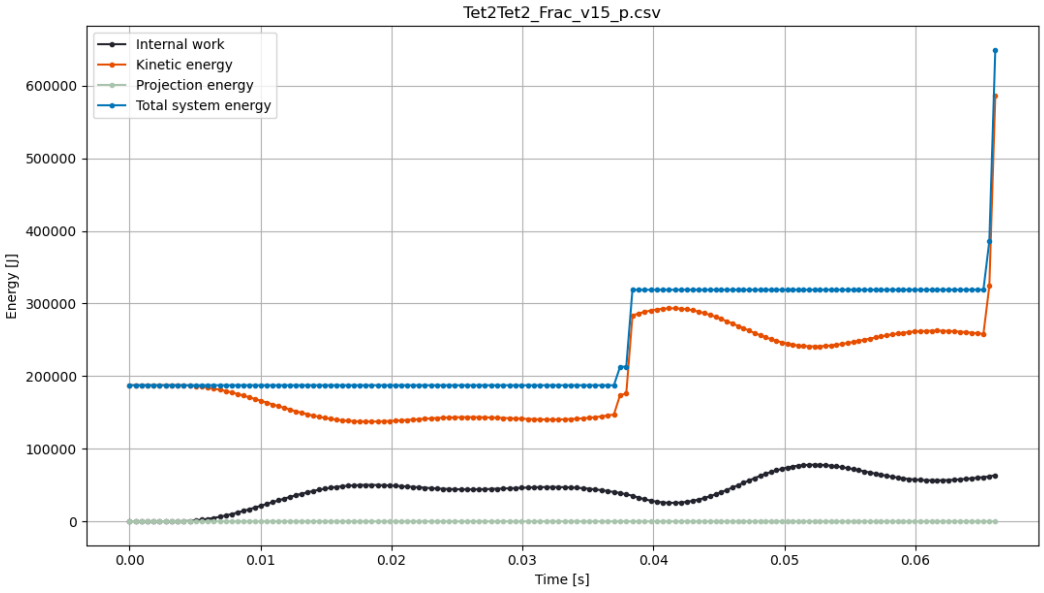


Figure 6.22: Plot showing the evolution of the system energy over time. In blue is represented the summation of internal work (in black), kinetic energy (in orange) and projection energy (in green). Each dot represents a simulation output value. Two major non-physical increases are visible for the total system energy. The one at $t \sim 0.037\ s$ corresponds to the removal of the cumulated intersection of the impactor DG/CZM interface. This happens with a number of interaction between faces 2 and 3, which explains the increase in kinetic energy. Then, on the far right the simulation is diverging due to the opening of the impactor's DG/CZM interface. A substantial cumulated intersection leads to the computation of large post-impact velocities. This starts a cascading feedback loop with each successive interaction of face 0 and 1 leading to an increase of velocity, eventually leading to a simulation crash due to excessive element deformation.

6.6. Observations about the numerical results

The results for the DG/CZM benchmarks are not satisfactory. Indeed, even if using very low number of elements, simple geometries and impact velocities far from the hypervelocity ranges, the methodology showed issues in energy conservation and non-physical behaviors. On the good side, the sources of these issues have been identified and some of the solutions could stem from the ideas discussed in subsection 4.4.3 and section 5.6.

Indeed, the non-physical results in subsections 6.5.2 and 6.5.3 is due to the detection of multiple edge-to-edge contact with a normal relative velocity between the impacting edge and the impacted triangle. This possibility was foreseen and some measures have been taken to avoid, like the criteria of subsection 5.4.1 and section 6.4. These did not prove sufficient, but the shift to a bulk element based mesh and to an extended sibling set formulation suggested in section 5.6 should make much easier to avoid the detection of these interactions. Additionally, modifying the edge-to-edge contact formulation as suggested in subsection 4.4.3 should avoid the calculation of skewed post-impact velocities.

However, the reason behind the divergence of the simulation subsection 6.5.4 is different: the DG/CZM interface elements measure interpenetrations at quadrature points which are not superposed to the nodes utilized by the DCR algorithm. Thus, a normal separation $\Delta_n \sim 0$ for a DG/CZM interface could correspond to a gap function $g \neq 0$ for the corresponding DCR faces. Possible solution would be to either act on the DG/CZM interfaces, in order to limit or even remove the possibility of interpenetrations, or to try and apply the DCR formulation to the same quadrature points acted upon by the DG/CZM formulation.

7

Conclusions

The focus of this thesis work has been to couple the Discontinuous Galerkin/ Cohesive Zone Method (DG/CZM) finite element formulation to the Decomposition Contact Response (DCR) algorithm, in order to lay the foundations for a novel simulation framework for hypervelocity impact.
At first, a literature study has been performed to gather information about the physics behind hypervelocity impact and about their characteristics phenomena. Then, the modelling approaches most commonly used for hypervelocity impact have been compared, in order to identify their strengths and weaknesses.
From this comparison, three sets of features have been highlighted as the most relevant for an hypervelocity impact simulation framework based on their mutual couplings. The first set is composed by contact, fracture and stress waves propagation, the second by hydrodynamic stress formulation and equation of state, and the third by a thermo-mechanical and coupled material model. Additionally, it has been established that no simulation technique is able to model both mid and low energy states (i.e., impacts with a relative velocity of $1\ km/s \leq v \leq 8\ km/s$) within one physics-based framework and is fit for an extension to full-scale satellite fragmentation simulation. Finite elements proved to be the best option for lower energy states and for scaling, while discrete elements give the best results for mid energy states.
For the velocity range of interest, the most fundamental feature to be included in a simulation framework is a fragmentation model (i.e., the combination of contact and fracture) that does not hinder stress wave propagation. Thus, this was made the focus of this thesis work. To model fracture and damage progression we chose the the DG/CZM finite element formulation, due to its proven ability of modelling dynamic fracture and to properly depict stress wave propagation. To complete the fragmentation model, we chose as a contact algorithm DCR, which showed good energy conservation properties.
Following an analysis of the DG/CZM and DCR algorithms, the steps necessary to adapt DCR to the DG finite element formulation have been identified. These included modifying the contact mesh structure, developing the novel concept of DG sibling node set and implementing measures against the detection of spurious contacts. While implementing these steps, some of the limitations found during the analysis of the previous implementation have also been tackled and removed. These steps have been qualitatively tested with simulations and have given overall satisfying results. However, some energy conservation issues have been observed. These issues have been explained with the interaction between the algorithms utilized to identify and flag the contact interactions to be solved in each time step and their supporting data structures.
Lastly, the DCR algorithm has been further modified to account for the development of fracture patterns in the mesh. The main measure introduced was to make the sibling node sets function of time to account for fracture onset and progression. The results in this case have been less positive, with energy conservation issues becoming more predominant and often hindering the completion of the simulation. This is mainly due to the mismatch between the evaluation of intersections of the DG/CZM interfaces and of the DCR algorithm and to the inability of the DCR to model normal impact involving the edges of the contact elements.
Solutions for the issues highlighted by the analysis of the analytical formulation and by the numerical simulations have been introduced and explained, but have not been implemented due to the time

constraints of a master graduation project.

7.1. Recommendations for future work

The results obtained at the end of this thesis work do not ensure physically sound results when running impact simulations using the DG/CZM formulation, even for lower initial impact velocities ($v \leq 25 m/s$). Nonetheless, useful information have been obtained on the limits of the DCR algorithm, of its current implementation within the SFC library and the conceptual outline of the steps needed for completing the development of a DCR-based contact algorithm for DG/CZM has been laid down. In fact, the DCR algorithm has shown limitations in its edge-to-edge impact formulation, discussed in section 4.1 and in the data structures it uses in the contact mesh. These structures limit its ability to deal with the complex mesh morphology of the DG/CZM formulation and have been discussed in section 4.2.
Specifically, inconsistencies in the definition of the edge-to-edge constraint functions have been discovered, which lead to a different evaluation of the gap vector compared to the node-to-face interactions and to the lack of a direct search of edge-to-edge intersections. Moreover, the approach used to evaluate the gap vector for the edge-to-edge interactions assumes an impact tangential to the impacted triangle plane, which can lead to the calculations of skewed post-impact velocities. These issues lead to the unsatisfactory results shown in the numerical simulations of sections 5.5 and 6.5.
The proposed solution are explained in depth in sections 4.3, 5.6 and 6.6 and are summarized in this paragraph.

7.1.1. Completing the implementation of the fragmentation model

To complete the implementation of the fragmentation model, the first step would be to change the mesh architecture. This would translate to changing the base unit of the contact mesh from surface to bulk elements. The current implementation of DCR uses a surface based contact mesh because in a continuous finite element setting only the outer boundary of solids can be involved in contact interactions. With the introduction of a discontinuous mesh and damage progression, any of the faces of each finite element could be involved in a contact interactions. Thus, using a volume based contact mesh would allow to use the topological and discretization information of the DG/CZM finite element mesh, which is more complete than the currently implemented DCR mesh. Additionally, the mesh formulation should use the concept of simplicial complexes. These would allow to simplify the storage and handling of the hierarchy and connectivity between different bulk elements and their faces, edges and vertices. Within the simplex domain, edges and bulk elements should have their dedicated classes, as triangles and vertices do in the current SFC architecture. This could for example simplify the implementation of an intersection search for edge-to-edge interaction.
Then, once implemented the new software architecture, one could change the intersections search to be a loop on bulk elements and to look for volume intersections, instead of surface-based ones. In this way, each interaction could be evaluated once and solved with the most fitting routine, being either node-to-face or edge-to-edge. This would make the code more efficient and would avoid solving the same physical contact interaction multiple times with different routines, as happened in subsection 5.5.2. Additionally, spurious contacts could be handled in a simpler way by extending the sibling set concept to edges and faces.
Lastly, the edge-to-edge formulation should be modified with a different constraint function, a different gap vector computation method and the relative velocities between impacting and impacted edges should be used to account for the direction of the impact. This would allow to avoid skewed post-impact velocities and to account for elongation/shortening of both involved edges. Considering the deformation of both impacted edges is suggested to remove the bias related to the symmetry of edge-to-edge impacts. Indeed, given that both edges participate to the contact interaction, it is not immediate to distinguish between leader and follower entity. Given how common edge-to-edge impact with relative velocities normal to the impacted triangle surface are, great emphasis should be placed on making sure that the new edge-to-edge analytical formulation can effectively treat this occurrences.

7.1.2. Completing the implementation of the hypervelocity impact framework

Once a working version of the DCR-based DG/CZM contact algorithm is implemented and its robustness and effectiveness have been demonstrated, additional features can be added in the direction of a fully fledged hypervelocity simulation framework.

The contact formulation should be expanded to include thermo-mechanical coupling. Indeed, this aspect is influencing contact in extreme conditions, such as those influenced in hypervelocity impact, in multiple ways. For example, it could be expanded with a coupling between temperature and friction coefficients, with heat generation due to friction dissipation, or with internal heat generation due to the removal of intersections and the following compression of bulk elements. An in-depth discussion on thermo-mechanically coupled finite element analysis involving contact can be found in [67].
At the same time, to obtain a DG/CZM framework applicable to hypervelocity impact it would be necessary to implement a hydrodynamic stress model, an equation of state, thermo-mechanic coupling and a material model accounting for thermal effects, strain rate saturation and plasticity. Indeed, a hydrodynamic stress formulation should be implemented together with an equation of state, like the volumetric bulk EOS or the Mie-Grüneisen EOS mentioned in subsection 2.1.2. Then, a thermo-mechanical coupling model should be added to compute the temperature field based on heat flow and on the heat generation created by volumetric compression. Afterwards, a material model suitable to hypervelocity impacts (for example, Johnson-Cook's or Steinberg-Guinan's, both mentioned in subsection 2.1.3) should be added to the simulation.
Lastly, all the above mentioned features and the complete fragmentation model should be implemented for parallel computations to improve the computational efficiency of the developed simulation framework.

7.1.3. Analyzing the interface instability observed for impacts on a discontinuous mesh

The analysis of the system energy proved to be a valuable tool in assessing the performance of the algorithm and in finding possible sources of errors and bugs. In section 5.5 oscillations were observed in the total system energy, which have been explained with the adjustments performed by the DG/CZM boundary integrals due to the use of a "rigid" cohesive law, which did not allow for fracture to start. This hypothesis could not be validated directly due to the lack of a suitable output variable. Indeed, Summit can output kinetic energy, internal and external work but not the work performed by the DG/CZM interfaces. Adding this functionality could help in the development and further expansion of a DCR-based DG/CZM contact algorithm.

Bibliography

[1] K. track. *Growing threat of space junk*. 2023. URL: https://www.keeptrack.space/deep-dive/growing-threat-of-space-junk/ (visited on 12/04/2023).

[2] E. Oneweb. *Our Network*. 2023. URL: https://oneweb.net/our-network (visited on 12/27/2023).

[3] SpaceX. *How Starlink works*. 2023. URL: https://www.starlink.com/technology (visited on 12/27/2023).

[4] e. a. M. Clormann X. Oikonomidou. "Fostering Collaborative Concepts in Space Debris Mitigation AIRBUS-UPM European Industrial Doctorate in mathematical-Marie Curie Initial Training Network-European Industrial Doctorate FP7-PEOPLE-2013-ITN methods applied to aircraft design View project Debris Risk Assessment and Mitigation Analysis (DRAMA) View project FOSTERING COLLABORATIVE CONCEPTS IN SPACE DEBRIS MITIGATION" ().

[5] Astromaterials Research and Exploration Science. *Orbital Debris Program Office*. 2022. URL: https://web.archive.org/web/20220902133204/https://www.orbitaldebris.jsc.nasa.gov/faq/ (visited on 09/02/2022).

[6] H. M. "Hypervelocity impact simulation using smoothed particle hydrodynamics". *TU Delft repository* (Dec. 2022).

[7] W. G. Colvin T.J. Karcz J. "Cost and Benefit Analysis of Orbital Debris Remediation". *Nasa report* (Mar. 2023).

[8] S. A. e. a. D.L. Oltrogge. "A comprehensive assessment of collision likelihood in Geosynchronous Earth Orbit". *Acta Astronautica* 147 (2018), pp. 316–345. DOI: https://doi.org/10.1016/j.actaastro.2018.03.017.

[9] C. S. Nuttall A. "A thermodynamic analysis of hypervelocity impacts on metals". *International Journal of Impact Engineering* (June 2020). DOI: 10.1016/j.ijimpeng.2020.103645.

[10] M. Rais-Rohani. "On the structural design of a mobile lunar habitat". *NASA/CR internal report* (2005). DOI: 10.13140/RG.2.1.3120.4962.

[11] B. D. Danny Braid. *Artemis homepage*. 2023. URL: https://www.nasa.gov/humans-in-space/artemis/ (visited on 12/21/2023).

[12] F. L. Whipple. "Meteorites and space travel." 52 (Jan. 1947). DOI: 10.1086/106009.

[13] R. Radovitzky, A. Seagraves, M. Tupek, and L. Noels. "A scalable 3D fracture and fragmentation algorithm based on a hybrid, discontinuous Galerkin, cohesive element method". *Computer Methods in Applied Mechanics and Engineering* 200 (2011), pp. 326–344. DOI: 10.1016/j.cma.2010.08.014.

[14] F. Cirak and M. West. "Decomposition contact response (DCR) for explicit finite element dynamics". *International Journal for Numerical Methods in Engineering* 64 (2005), pp. 1078–1110. DOI: 10.1002/nme.1400.

[15] H. A. Signetti S. "Transition regime between high-velocity and hypervelocity impact in metals – A review of the relevant phenomena for material modeling in ballistic impact studies". *International Journal of Impact Engineering* 167 (Sept. 2022). DOI: 10.1016/j.ijimpeng.2022.104213.

[16] K. H. Hopkins H.G. *Mechanics of hypervelocity impact of solids*. ARDE, 1960.

[17] J. W. et al. "Impact strength of materials" (1972).

[18] A. Piekutowski. "Formation and description of debris clouds produced by hypervelocity impacts". *NASA Contractor Report* (Feb. 1996).

[19] M. J. e. a. Cowardin H. "Optical characterization of debrisat fragments in support of orbital debris environmental models". *Journal of the astronatuical sciences* 68 (2021).

[20] C.-p. B.G. "THE SHAPE EFFECT OF NON-SPHERICAL PROJECTILES IN HYPERVELOCITY IMPACTS". *International journal of impact engineering* 26 (2022).

[21] K. R. e. a. Poniaev S.A. "Hypervelocity impact of mm-size plastic projectile on thin aluminum plate". *Acta Astronautica* 12 (Nov. 2017). DOI: 10.1016/j.actaastro.2016.11.011.

[22] e. a. Wu C. He Q. "Debris cloud structure and hazardous fragments distribution under hypervelocity yaw impact". *Defence Technology* (Aug. 2022). DOI: 10.1016/j.dt.2022.09.010.

[23] G. F. Liu X. and C. S. "Numerical Simulation on Characteristics of Debris Clouds Produced by Conical Projectiles Hypervelocity Impact on Thin Plates". *International Journal of Hybrid Information Technology* 8 (June 2015). DOI: 10.14257/ijhit.2015.8.6.08.

[24] M. J. Carrasquilla M.J: "Shape Effect Analysis of Aluminum Projectile Impact on Whipple Shields". *Procedia Engineering* 204 (Apr. 2017). DOI: 10.1016/j.proeng.2017.09.751.

[25] A. R. "Hypervelocity Impact on Satellite Sandwich Structures: Development of a Simulation Model and Investigation of Projectile Shape and Honeycomb Core Effects". *PHD dissertation* (June 2021).

[26] J. Zukas. *Introduction to Hydrocodes*. Baltimore, USA: Computational Mechanics Association, 2004.

[27] H. R.A. "Shock compression technique for developing multiphase equations of state". *Los Alamos Science* 28 (2003).

[28] G. R. Duvall G.E. "Phase transitions under shock-wave loading". *Reviews of Modern Physics* 49 (July 1977).

[29] K. Nagayama. *Handbook of shockwaves*. volume 1. Fukuoka, Japan: Academic Press, 2001.

[30] J. Johnson. "Single-particle model of a solid: the Mie-Grüneisein equation". *American Journal of Physics* 36.10 (1968), pp. 917–919. DOI: 10.1119/1.1974315.

[31] J. H. Tillotson. "Metallic equations of state for hypervelocity impacts". *Internal documents* (1962).

[32] H. J. e. a. Faergestad R.M. "Coupled finite element-discrete element method (FEM/DEM) for modelling hypervelocity impacts". *Acta Astronautica* (May 2023). DOI: 10.1016/j.actaastro.2022.11.026.

[33] Johnson. "History and application of hydrocodes in Hypervelocity impact". *International Journal of Impact Engineering* 5 (1987).

[34] M. G. D.J. Steinberg S.G. Cochran. "A constitutive model for metals applicable at high-strain rate". *Journal of applied physics* 51 (Nov. 1980). DOI: doi.org/10.1063/1.327799.

[35] A. O. R.L. Bjork. "The role of melting and vaporization in hypervelocity impact". *Rand corporation memorandum* (May 1965).

[36] L. Y. An F.; Zhang Y.; Liao S.; Wu C. "Study on Numerical Simulation Methods for Hypervelocity Impact on Large-Scale Complex Spacecraft Structures". *Aerospace* 55 (Nov. 2021). DOI: 10.3390/aerospace9010012.

[37] P. A. Spear D.G. "Modelling and simulation techniques use in high strain rate projectile impact". *Air Force Institute of Technology, faculty publications* 9 (Jan. 2021). DOI: 10.3390/math9030274.

[38] L. X. Z. X. L. T. J. G. "Element fracture for hypervelocity impacts simulation". *Advances in Space research* 55 (Feb. 2015). DOI: dx.doi.org/10.1016/j.asr.2015.01.040.

[39] Z. X. L. T. Q. X. "Orthotropic node-separation finite element method for composite laminate in hypervelocity impact simulation". *Acta Astronautica* (June 2017). DOI: 10.1016/j.actaastro.2017.07.049.

[40] G. R. Monaghan J.J. "Shock simulation by the particle methods SPH". *Journal of computational physics* 52 (Sept. 1983).

[41] J. G. H. H. Zhang X. "Finite element reconstruction approach for on-orbit spacecraft breakup dynamics simulation and fragment analysis". *Advances in space research* 51 (Sept. 2013). DOI: 10.1016/j.asr.2012.09.023.

[42] S. M. Watson E. "Discrete Particle Method for Simulating Hypervelocity Impact Phenomena". *Materials* 10 (Apr. 2017). DOI: 10.3390/ma10040379.

[43] T. S. McGlaun J.M. "CTH: a three-dimensional shock wave physics code". *Kaman Sciences corporation* (1990).

[44] S. N. Laboratory. *Sandia National Laboratory CTH*. 2023. URL: https://www.sandia.gov/cth/.

[45] L. R.J: "The influence of phase changes on debris-cloud interactions with protected structures". *International journal of impact engineering* 17 (1995).

[46] C. J. He Q.G. Chen X. "Finite element-smoothed particle hydrodynamics adaptive method in simulating debris cloud". *Acta astronautica* (May 2020). DOI: 10.1016/j.actaastro.2020.05.056.

[47] J. G.R. "Linking of Lagrangian particle methods to standard finite element methods for high velocity impact computations". *Nuclear engineering and design* (2020).

[48] C. J. Hirt C.W. Amsden A.A. "An Arbitrary Lagrangian-Eulerian Computing Method for All Flow Speeds". *Journal of computational physics* 14 (Nov. 1974).

[49] G. M.S. "Recent trends in ALE formulation and its applications in solid mechanics". *Computer Methods in Applied Mechanics and Engineering* (Oct. 2004). DOI: 10.1016/j.cma.2004.02.019.

[50] C. G. e. a. Nassiri A. "Adapting a Multi-Material ALE with AMR Method for Physics of High-Speed Material Interactions". *Materials and Design* 88 (Sept. 2015). DOI: 10.1016/j.matdes.2015.09.005.

[51] T. E. e. a. Yip P. "Adapting a Multi-Material ALE with AMR Method for Physics of High-Speed Material Interactions". *AIAA SciTech forum* (Jan. 2022). DOI: 10.2514/6.2022-0675.

[52] A. Seagraves and R. Radovitzky. "Large-scale 3D modeling of projectile impact damage in brittle plates". *Journal of the Mechanics and Physics of Solids* 83 (2015), pp. 48–71. DOI: 10.1016/j.jmps.2015.06.001.

[53] H. Espinosa and P. Zavattieri. "A grain level model for the study of failure initiation and evolution in polycrystalline brittle materials. Part I: Theory and numerical implementation". *Mechanics of Materials* 35 (2003), pp. 333–364.

[54] P. Klein, J. Foulk, E. Chen, S. Wimmer, and H. Gao. "Physics-based modeling of brittle fracture, cohesive formulations and the applications of meshfree methods". *Theoretical and Applied Fracture Mechanics* 37 (2001), pp. 99–166.

[55] F. Bassi and S. Rebay. "A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier-Stokes equations". *Journal of Computational Physics* 131 (1997), pp. 267–279. DOI: 10.1006/jcph.1996.5572.

[56] L. Noels and R. Radovitzky. "An explicit discontinuous Galerkin method for non-linear solid dynamics. Formulation, parallel implementation and scalability properties." *International Journal for Numerical Methods in Engineering* 74.9 (2007), pp. 1393–1420. DOI: 10.1002/nme.2213.

[57] M. Ortiz and A. Pandolfi. "Finite-deformation irreversible cohesive elements for three-dimensional crack-propagation analysis". *International Journal for Numerical Methods in Engineering* 44 (1999), pp. 1267–1282.

[58] G. Lindfield and J. Penny. "Chapter 5 - Solution of Differential Equations". *Numerical Methods (Fourth Edition)*. Ed. by G. Lindfield and J. Penny. Fourth Edition. Academic Press, 2019, pp. 239–299. DOI: https://doi.org/10.1016/B978-0-12-812256-3.00014-2.

[59] V. L. Popov. "Coulomb's Law of Friction". *Contact Mechanics and Friction: Physical Principles and Applications*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2017, pp. 151–172. DOI: 10.1007/978-3-662-53081-8_10.

[60] R. Radovitzky. *Summit*. 2019.

[61] Bianca Giovanardi's research group. *TU Delft's Gitlab page for Summit lite software*. Online: https://gitlab.tudelft.nl/aces/summit-lite. 2022.

[62] R. Graglia, D. Wilton, and A. Peterson. "High order interpolatory vector bases for computational electromagnetics". *Antennas and Propagation, IEEE Transactions on* 45 (Apr. 1997), pp. 329–342. DOI: 10.1109/8.558649.

[63] G. Haikal and K. Hjelmstad. "A finite element formulation of non-smooth contact based on oriented volumes for quadrilateral and hexahedral elements". *Computer Methods in Applied Mechanics and Engineering* 196.45 (2007), pp. 4690–4711. DOI: https://doi.org/10.1016/j.cma.2007.06.002.

[64] J. Matousek. "Using The Borsuk-Ulam Theorem". 2003.

[65] M. A. Puso and T. A. Laursen. "A mortar segment-to-segment contact method for large deformation solid mechanics". *Computer Methods in Applied Mechanics and Engineering* 193.6 (2004), pp. 601–629. DOI: https://doi.org/10.1016/j.cma.2003.10.010.

[66] R. Ogden. *Non-linear elastic deformations*. New York: Dover, 1997.

[67] D. Pantuso, K.-J. Bathe, and P. A. Bouzinov. "A finite element procedure for the analysis of thermo-mechanical solids in contact". *Computers & Structures* 75.6 (2000), pp. 551–573. DOI: https://doi.org/10.1016/S0045-7949(99)00212-6.

A

Template License