POLITECNICO DI TORINO

Corso di Laurea Magistrale
in Ingegneria Aerospaziale

Tesi di Laurea Magistrale

Experimental Verification of Three-Dimensional Mixed-Mode Crack Propagation Analysis

Relatore
Prof. Stefano Zucca

Candidato
Jury Rodella

Relatori esterni
Dr. Guido Dhondt
Prof. Markus Zimmermann
M.Sc. Simon Pfingstl

Anno Accademico 2019/2020

Experimental Verification of Three-Dimensional Mixed-Mode Crack Propagation Analysis

Master's Thesis

Scientific Thesis for Acquiring the
Master of Science Degree
at the Department of Mechanical Engineering
of the Technical University of Munich.

Thesis Advisor: Laboratory for Product Development and Lightweight Design
Prof. Dr. Markus Zimmermann

Supervisor: Laboratory for Product Development and Lightweight Design
M.Sc. Simon Pfingstl

Submitted by: Jury Rodella
Bahnhofstr. 55b, 85375, Neufahrn
Matriculation number: 03727559
jury.rodella@gmail.com

Submitted: 02.10.2020

Declaration

I assure that I have written this work autonomously and with the aid of no other than the sources and additives indicated.

Place date Signature

Project Definition (1/2)

Initial Situation

In the Aerospace field, the phenomenon of fatigue is one of the biggest challenges which have to be considered during the design phase. Material flaws, pre-cracks and crack initiations due to cyclic loading may lead to crack propagation in aircraft components. For aero-engine parts, because of high temperatures and time-dependent loads, this occurrence induces often a catastrophic failure. During the product development, a Damage Tolerant Design must be made, within an accurate prediction of the phenomenon. Because of the intricate geometries and loadings, the result is frequently a mixed-mode crack propagation and advanced tools for its estimation are required. Cracktracer3D is an MTU software able to study the mixed-mode cyclic crack propagation, based on the calculation of the Stress Intensity Factor from the FEM stress field analysis. In some cases, the complexity of the model, the impact of several parameters and computational limitations, create numerical results which may be imprecise and different from the experimental ones.

Goals

Under the assumption of crack nucleation, the subsequent crack propagation must be investigated. The thesis aims to analyse the most complex scenario, namely, the mixed-mode crack propagation. An in-depth study of the experiments is conducted to comprehend the crack behaviour and related non-idealities. Through several tests, the predicted results are compared and examined to verify the validity of Cracktracer3D. A deep understanding of the theory behind the software is essential to achieve the best code-settings. Two main categories of specimens are studied: Single Edge Cracked Four-Point Bending Specimen with slanted crack and Single Edge Cracked Specimen with traction-torsion sinusoidal loads in different phases. The findings are precisely compared and arising discrepancies are used as a starting point for a different and enhanced crack propagation approach. This new method has been implemented in a Fortran subroutine of Cracktracer3D, aiming to match the experimental mixed-mode propagation results.

Project Definition (2/2)

Contents of this Thesis:

- Introduction
  - Initial situation
  - Objectives
- Fracture Mechanics
  - Design in the Aerospace field
  - General crack propagation principles
  - Mixed-mode crack propagation
- Cracktracer3D description
  - General introduction
  - Working principle
- Specimens analyzed
  - 4PB Specimen
  - Tension-Torsion Specimen
- Verification: results and comparison
  - Numerical results
  - Experimental results
  - Comparison
- A new mixed-mode crack propagation approach
  - Description
  - Implementation
  - Verification
- Conclusions

An accurate elaboration, a comprehensible and complete documentation of all steps and applied methods, and a good collaboration with industrial partners are of particular importance.

Project Note

Master's Thesis	
Supervisor	Simon Pfingstl
Time period	01.04.2020 - 30.09.2020

The dissertation project of Mr. Rodella Jury set the context for the work presented. My supervisor Mr. Pfingstl Simon mentored me during the compilation of the work and gave continuous input. We exchanged and coordinated approaches and results weekly.

Publication
I consent to the laboratory and its staff members using content from my thesis for publications, project reports, lectures, seminars, dissertations and postdoctoral lecture qualifications.

Signature of student: ____

Signature of supervisor: ____

Contents

1 Introduction

Nowadays, reliability and efficiency constitute the design key for many technical fields, including aerospace engineering. For aircraft structures and components, failure must be necessarily avoided, ensuring on the same time high performance. Several factors play an important role during the aircraft operating life such as unsteady flight dynamics, stress concentrations, presence of notches and holes, material flaws and environmental factors like corrosion or temperature swings.

The engines represent one of the most critical structural aspects of an aircraft. Current aero-engines have to provide wide thrust ranges for lower fuel consumption under high thermal and mechanical loads, which may lead to creep-fatigue damage at some components. Furthermore, each phenomenon could combine with different mechanisms due to environmental attack such as corrosion/fatigue or oxidation/erosion (Meher-Homji & Gabriles, 1998, p.4). Although engine components are strictly subjected to careful investigation through the design phase, many issues may still take place during the operating life. For instance, a bird strike is an event that gets into the engine external parts and is not predictable. However, to certify the engine in case of foreign-object damage, a previous test has to be performed, since, depending on the impact severity, this could either break a component or create microstructural damages with residual stresses and consequent crack growth (Peters & Ritchie, 2000, p.2).

The reduction of safety margins, within the optimization of aero-engines, causes the need to ensure the required component life in the presence of cyclic loads, namely, crack propagation problems. Thereby, the necessity to predict the number of loading cycles to failure, as well as the crack propagation direction into the material, has increased over the last years, taking along new numerical methods and tools (Dhondt, 2014, p.1). However, as soon as we move toward realistic cases, the complexity to achieve reliable results becomes higher, e.g. engine blades, where the crack propagates mainly in three dimensions, originating mixed-mode crack propagation. This phenomenon poses one of the toughest adversities in fracture mechanics, although it is the most frequent way of propagation. Because of centrifugal and aerodynamic loading conditions, the prevalent locations for critical stress fields, leading to crack initiation and further propagation, are the blade attachments in typical gas turbine engines (Barlow & Chandra, 2005, p.1).

To study the crack propagation phenomenon, at MTU Aero Engines, the in-house software Cracktracer3D is used. It is based on a Finite Element Method solution of the structure and simulates the mixed-mode cyclic crack propagation in a complete self-acting way. It works iteratively, inserting into the uncracked structure the current crack shape and thereafter, it meshes the body, solves the stress field with the FEM and calculates the new crack progress by means of the stress intensity factor around the crack tip. The software is structured in three parts: preprocessor, FEM-solver and postprocessor. Inside the preprocessor, the user can set the initial crack in an arbitrary position and select the crack propagation domain inside the meshed component. Once the input data are processed, the current cracked structure model is sent to the free software CalculiX, for three-dimensional finite element stress solution (Dhondt & Wittig, 2020). The postprocessor analyses the stress intensity factors distribution along the crack front and finds the crack propagation direction and length, upgrading the crack geometry for the next iteration input. This loop runs up to a specific request of the user (Dhondt, 2014).

The fatigue component analysis can be investigated following either numerical or experimental approaches. Computer-aided engineering aims to foretell the behaviour by numerical analysis, supporting the design phase and saving money and time for the companies. Due to high costs, data reliability and complicated implementation, resorting to experimental tests is most of the time the inefficient way to proceed. Nevertheless, the software itself has to be validated in order to prove its accuracy and it can be achieved only with experimental findings (Riddell, Ingraffea, & Wawrzynek, 1997, p.13-15). In this thesis, a validation procedure of Cracktracer3D is proposed. By reason of complex geometry and significant costs, testing a real engine component would not be feasible to implement. However, with simple specimens, it has been possible to replicate the trickiest situation, including mixed-mode crack propagation or the fatigue behaviour under sophisticated loading missions. The validation described in this work is focused on these last two cases. In addition, since there is not a unique way to choose the crack propagation direction, the current approach used by Cracktracer3D is analysed and compared with a new criterion.

2 Fracture Mechanics

In this chapter, the basics of fracture mechanics are described. The following coverage provides the reader with the necessary information to comprehend mixed-mode crack propagation. In section 2.1 the topic introduction is given by referring to the aerospace field. In section 2.2 and section 2.3 the linear-elastic fracture mechanics principles and the calculation of the J-integral are shown. At the end of the chapter, in section 2.4, the cyclic crack propagation problem is introduced with its approximation by analytical models.

2.1 An Introductory Overview

2.1.1 The Role in the Aerospace Field

Fracture mechanics concerns the processes of strain and fracture of solids containing cracks, notches and material flaws. In the linear theory of elasticity, the presence of these fissures leads to infinite stresses at their tip (Savruk & Kazberuk, 2009, p.1).

Typically, to predict the behaviour of uncracked solids, the theory of elasticity is used, eventually obtaining the set of constitutive equations. However, when solids have imperfections and cracks, the field equations cannot be entirely found by the theory of elasticity because the singularity of the stress near the crack tip is not taken into account (Perez, 2017).

Fracture mechanics studies the fracture toughness as an interaction between the flaw size and the applied stress (Anderson, 2005). In 1913, Inglis started to analyse the growth of an elliptical hole in a plate, trying to degenerate it into a crack. However, the stress problem at the crack tip was not solved. This was the starting point for Griffith's theory, who developed an energy method giving an outstanding result. Later on, this theory was used by Irwin and Orowan to explain the failure of metal cracked structures. By means of Westergaard's method, Irwin obtained the first expression of the crack tip stress in the elastic field. He also was the first to define the three modes of crack propagation and the determination of the stress intensity factors, K_I, K_{II} and K_{III}. In 1959, Paris demonstrated the relation between the crack growth rate and the stress intensity factor. Fracture mechanics became more and more important through the years, bringing along many solutions of the problem, from Wigglesworth, Koiter, Bueckner and Isida, to Newman with one of the first applications for numerical methods. Afterwards, the problem of plasticity in the reversed cyclic plastic zone was also analysed by H.H. Johnson and many other complex problems such as the crack paths and changes in direction or crack instability. In 1969, fracture mechanics showed its importance in the aircraft field after the crash accident of a U.S. Air Force F-111. From that moment on, to ensure the safety of aircraft, a fatigue proof of the components became a must, leading to the safe and reliable modern-day vehicles (Paris, 2014).

The phenomenon of metal fatigue takes place when a structure is subjected to cyclic loading. Small cracks start near stress concentration areas as soon as the stress amplitude exceeds a threshold value. Once the propagation starts, at the beginning the progress is very slowly, hence, it is difficult to detect, but the crack propagation rate increases over the fatigue life and the cracks become visible. Failure is reached once the crack is at a certain critical size. The tensile and yield strength for the static load is much higher than stress levels that lead to fatigue failure. During the design of a new aircraft, many requirements have to be satisfied. The engineering field has become even more sophisticated over the years and these requirements represent difficult challenges. Sometimes, the idea of a new design process, new materials or modern fabrication processes may be developed, and when there is a lack of experience, some deficiencies and flaws arise. Since the structural design is mainly grounded in empirical rules obtained from previous experience, as soon as a new concept is introduced, a new design process has to be developed. Usually, due to the large number of factors to consider, the design is achieved by a large number of iterations that start from simple assumptions and progressively get more accurate (Hardrath, 1971, p.2).

The icons of high reliability and sophisticated development are aircraft engines. A gas turbine engine is mainly composed of three principal parts: compressor, combustor and turbine. One of the most critical parts may be the low-pressure compressor blades since the environment in which they operate is very unfavourable as well as the load conditions. Indeed, the blades have a combination of loads like high-cycle fatigue, low-cycle fatigue, centrifugal tensile stress, aerodynamic stress and vibrations. Besides that, the position of the compressor may also be an issue because of the possibility of strike with foreign objects and atmospheric corrosion. All these factors combined, cause structural fatigue problems. Statistics show that their main structural failure mode, in about 25% of failure cases, is because of high-cycle fatigue (Zhang, Yang, & Hu, 2018, p.2). Figure 2.1 shows an example of compressor blade failure with a fracture in the airfoil blade root and the damaged blades.

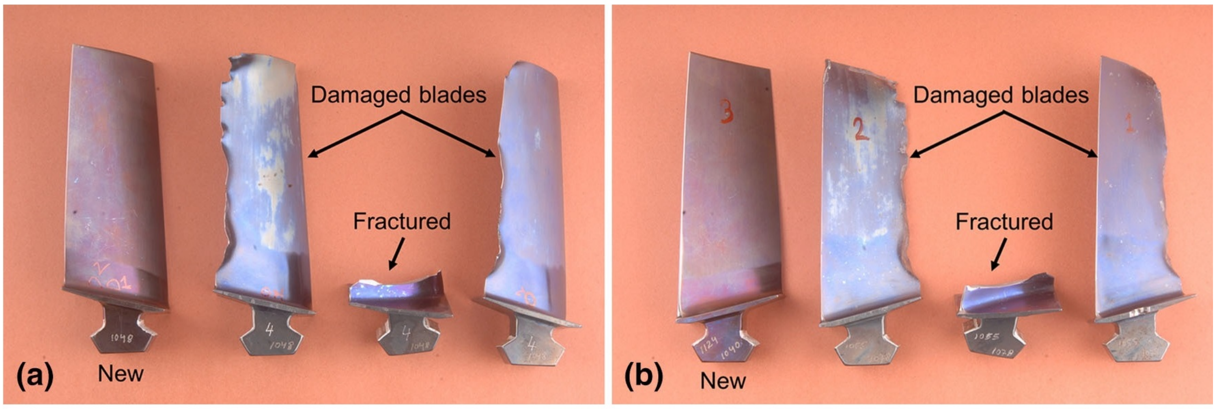

Figure 2.1: HPCR blades failure in aero-engine (Sujata & Bhaumik, 2015)

It is interesting to compare the failure modes between aircraft components and generic engineering components. In Table 2.1 the percentage of failures is reported with the correspondent cause, when looking at aircraft components, it is clear that the main failure mode occurs due to the fatigue in 55% of cases. Corrosion is also important, however, the percentage stays at 16%. This difference between corrosion and fatigue is because of two reasons:

- Fatigue is sometimes visible with bare eye and arises with crack propagation and subsequent destruction of the components, hence it is easier to observe.
- Corrosion is a slower process, thus, there are more possibilities to repair or change the

flawed components, reducing the percentage of failure.

Even though the fatigue behaviour of many materials is well known, the fact that fatigue failures still occur is a demonstration of the complexity of this phenomenon. Nowadays, aircraft components are subjected to non-destructive inspections in order to detect possible flaws after manufacturing. Unfortunately, surface defects may occur during the service life, making their detection impossible after the production, e.g. corrosion (Findlay & Harrison, 2002).

Table 2.1: Frequency of failure modes (Findlay & Harrison, 2002)

	Percentage of Failures	
	Engineering Components	Aircraft Components
Corrosion	29	16
Fatigue	25	55
Brittle fracture	16	-
Overload	11	14
High temperature corrosion	7	2
SCC/Corrosion fatigue/HE	6	7
Creep	3	-
Wear/abrasion/erosion	3	6

2.1.2 Fatigue Design Approaches

To guarantee the non-appearance of failures during service life, for aerospace structures some fatigue design approaches are used. Following the time evolution, the safe-life was the first fatigue philosophy applied, followed by the fail-safe and eventually the damage-tolerance approach. The latter is currently mainly used, ensuring reliability, structural resistance and low weight.

Given the load mission, safe-life ensures a specific fatigue life for a component with no inspections. It is the reason why this approach is still utilized for aircraft parts like the landing gear. Fail-safe is used for those components that can be inspected like the fuselage, where it must be defined what kind of flaws are feasible and which ones have to be fixed. Damage-tolerant follows a different design: it allows damages but you have to predict their time evolution to guarantee safety and avoid failure (Tavares & De Castro, 2017, p.2).

In other words, the safe-life method imposes to substitute the component after you reach prearranged life and this is done by dividing its average life by a safety factor. Obviously, the mean life has to be determined through many tests and the safety factor needs to be accurate, considering all possible events. Commonly, a risk analysis is done with a study of the probability of events to appear. In this way, using a statistical approach, the influence of many design parameters can be considered to achieve the best safety factor (Lazzeri, 2002). On the other hand, the damage-tolerance method considers the existence of initial cracks in critical positions and estimates the relative component life. For aircraft, the loads applied on a part are constantly recorded by sensors and this is used to redefine its service life after the scheduled inspection. Indeed, during the inspection, there may be two cases: either it does not have cracks, or cracks and flaws are detected inside the component. For the first case, the new component life is updated according to the recent fatigue load spectrum recorded, whereas, in case of cracks detection, their size is measured and used as an initial crack to predict the remaining life within the current fatigue load spectrum. Moreover, this method allows to decide the right inspection period, as well as to ensure a proper check of critical aircraft components (A. F. Liu, 2005).

2.2 Linear-Elastic Fracture Mechanics (LEFM)

2.2.1 Loading Modes

The nature of a crack allows to define three fundamental modalities of loading at the crack tip that differ between each other in the movement of the crack surfaces which individuate the crack front. In Figure 2.2 this is represented as a local crack element for each mode with the respective crack surface displacement. More specifically, the modes are defined as follows:

- Mode I, known as opening mode. The crack surfaces move orthogonally and symmetrically to the crack plane under normal stress.
- Mode II, known as sliding mode. The crack surfaces slide antisymmetrically on the crack plane under in-plane shear stress.
- Mode III, known as tearing mode. The crack surfaces move antisymmetrically on the crack plane under out-of-plane shear stress.

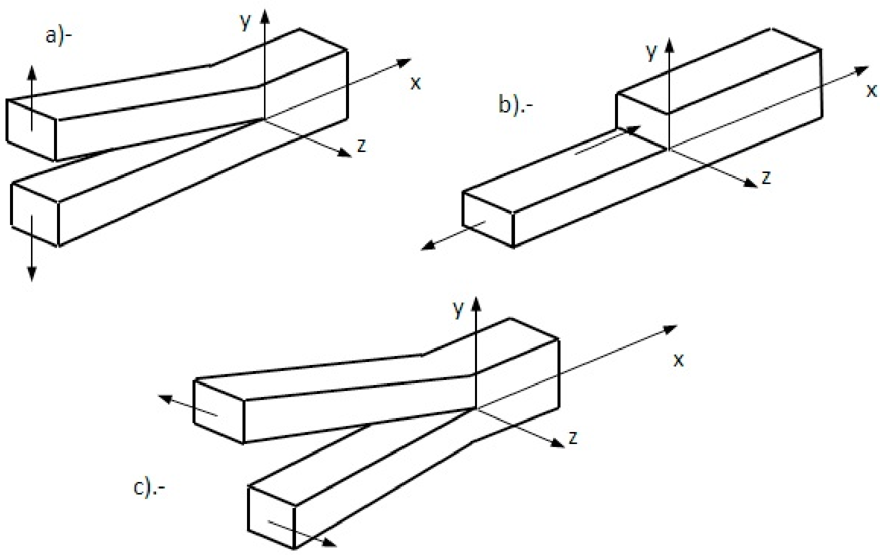

Figure 2.2: Loading modes: a) Mode I, b) Mode II and c) Mode III (Chambel, Martins, & Reis, 2016)

The combination of these three modes leads to the so-called mixed-mode crack propagation, which depicts the most frequent case in real problems, due to non-regular crack geometries or complex loading modes (Gdoutos, 2020).

2.2.2 Stress Field and Stress Intensity Factor (SIF)

To study crack fatigue behaviour, it is essential to know the stress field distribution around it a priori. It turns out that for the following treatment, the best approach is to use a polar coordinate system near the crack front as reported in Figure 2.3.

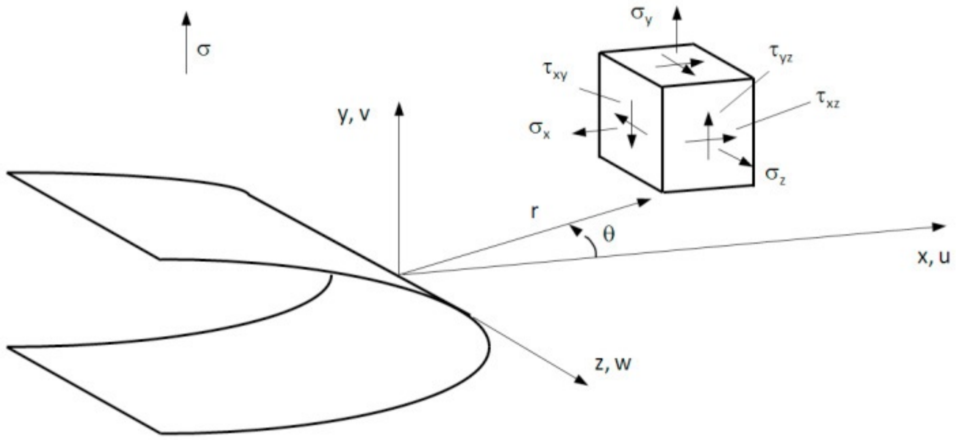

Figure 2.3: Polar coordinate system (r,θ,z) and stress orientation around the crack tip (Chambel et al., 2016)

Considering this scheme, it is analytically proved that for a solid cracked body under the assumption of isotropic linear-elastic material, the stress tensor can be determined as reported in Equation 2.1 (Anderson, 2005).

$$\sigma_{ij} = \left(\frac{k}{\sqrt{r}}\right) f_{ij}(\theta) + \sum_{m=0}^{\infty} A_m r^{\frac{m}{2}} g_{ij}^{(m)}(\theta) \tag{2.1}$$

The σ_{ij} represents the i-jth stress tensor component at the point identified by r and θ (polar coordinates). The first term, namely the dominant one, contains the constant k and f_{ij}, which is a dimensionless function of θ. The summation introduces the higher-order terms that are mostly neglectable or finite, thus, it is clear that the stress field shows a singular behaviour when $r = 0$ due to the proportionality of the main term to $\frac{1}{\sqrt{r}}$.

The stress field near the crack tip can be also quantified by means of the stress intensity factor, which is a constant that takes into account the shape and position of the crack, as well as the strength and the mode of loading. The stress intensity factor is defined as $K = k\sqrt{2\pi}$ and as previously discussed in the subsection 2.2.1, this lead to K_I,K_{II} and K_{III}, respectively to its fundamental loading mode.

$$\begin{aligned}
\sigma_{ij}^{(I)} &\sim \frac{K_I}{\sqrt{2\pi r}} f_{ij}^{(I)}(\theta) + \mathscr{O}(1) \\
\sigma_{ij}^{(II)} &\sim \frac{K_{II}}{\sqrt{2\pi r}} f_{ij}^{(II)}(\theta) + \mathscr{O}(1) \\
\sigma_{ij}^{(III)} &\sim \frac{K_{III}}{\sqrt{2\pi r}} f_{ij}^{(III)}(\theta) + \mathscr{O}(1)
\end{aligned} \tag{2.2}$$

In the case of mixed-mode, the final value of σ_{ij} is given by the sum of the components of active modes. Moreover, detailed expression of the non-zero tensor components for mode I, II and III, has the following expressions as stated by (Anderson, 2005):

- Mode I:

$$\begin{aligned} \sigma_{xx} &= \frac{K_I}{\sqrt{2\pi r}} \cos\left(\frac{\theta}{2}\right)\left[1-\sin\left(\frac{\theta}{2}\right)\sin\left(\frac{3\theta}{2}\right)\right] \\ \sigma_{yy} &= \frac{K_I}{\sqrt{2\pi r}} \cos\left(\frac{\theta}{2}\right)\left[1+\sin\left(\frac{\theta}{2}\right)\sin\left(\frac{3\theta}{2}\right)\right] \\ \tau_{xy} &= \frac{K_I}{\sqrt{2\pi r}} \cos\left(\frac{\theta}{2}\right)\sin\left(\frac{\theta}{2}\right)\cos\left(\frac{3\theta}{2}\right) \\ \sigma_{zz} &= \begin{cases} 0 & \text{(Plane stress)} \\ \nu\left(\sigma_{xx}+\sigma_{yy}\right) & \text{(Plane strain)} \end{cases} \end{aligned} \tag{2.3}$$

- Mode II:

$$\begin{aligned} \sigma_{xx} &= -\frac{K_{II}}{\sqrt{2\pi r}} \sin\left(\frac{\theta}{2}\right)\left[2+\cos\left(\frac{\theta}{2}\right)\cos\left(\frac{3\theta}{2}\right)\right] \\ \sigma_{yy} &= \frac{K_{II}}{\sqrt{2\pi r}} \sin\left(\frac{\theta}{2}\right)\cos\left(\frac{\theta}{2}\right)\cos\left(\frac{3\theta}{2}\right) \\ \tau_{xy} &= \frac{K_{II}}{\sqrt{2\pi r}} \cos\left(\frac{\theta}{2}\right)\left[1-\sin\left(\frac{\theta}{2}\right)\sin\left(\frac{3\theta}{2}\right)\right] \\ \sigma_{zz} &= \begin{cases} 0 & \text{(Plane stress)} \\ \nu\left(\sigma_{xx}+\sigma_{yy}\right) & \text{(Plane strain)} \end{cases} \end{aligned} \tag{2.4}$$

- Mode III:

$$\begin{aligned} \tau_{xz} &= -\frac{K_{III}}{\sqrt{2\pi r}} \sin\left(\frac{\theta}{2}\right) \\ \tau_{yz} &= \frac{K_{III}}{\sqrt{2\pi r}} \cos\left(\frac{\theta}{2}\right) \end{aligned} \tag{2.5}$$

where ν is the Poisson's ratio. However, in real materials, infinite stress at the crack tip is not allowed by nature. In fact, at the crack tip, the materials have always plastic behaviour restricted to the so-called plastic zone. It can be affirmed that the main limitation of linear-elastic fracture mechanics is that the plastic zone is considered neglectable with respect to crack size (Caputo, Lamanna, Lanzillo, & Soprano, 2013).

2.3 J-Integral

As reported previously, linear-elastic fracture mechanics does not consider the plastic zone at the crack tip, showing its greatest limitation. In order to include also the elastoplastic deformation inside a fracture criterion, the J-integral has been introduced. It represents a fracture parameter which takes into account the non-linearity of the materials. The value of the J-integral comes from an energetic study of the crack: J can be seen as the energy release rate in a nonlinear elastic body with a crack. Through the use of a line integral around the crack front, it is possible to find a value of this energy release rate as shown in Equation 2.6, where the quantity W is the strain energy density, $\vec{T}$ the tension vector, ds the differential element of the contour Γ and $\vec{\mu}$ the displacement vector (Perez, 2017).

$$J = \int_{\Gamma} \left(W dy - \vec{T} \frac{\partial \vec{\mu}}{\partial x} ds \right) \tag{2.6}$$

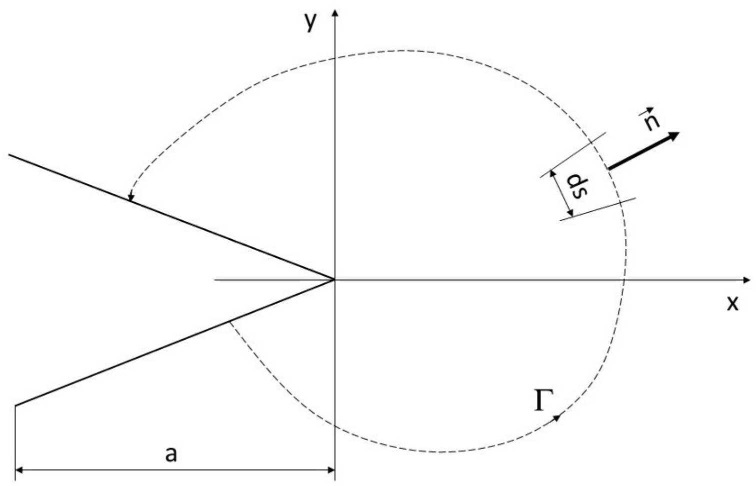

Figure 2.4: J-integral path around the crack tip

In case of linear elastic material and mode I, the J-integral assumes the simple form (Caputo et al., 2013):

$$J = \frac{K_I^2}{E'} \quad \text{where} \quad E' = \begin{cases} E & \text{(plane stress)} \\ \frac{E}{1-\nu^2} & \text{(plane strain)} \end{cases} \tag{2.7}$$

It is necessary to remind that the J-integral derives from an elastic study, not necessarily linear, but is used for plastic material behaviour. Actually, it can be utilized as a failure criterion instead of the critical stress intensity factor fracture criterion. The latter says that under mode I, the fracture condition is reached when the stress intensity factor is equal to its critical value:

$$K_{\mathrm{I}} = K_{\mathrm{c}} \tag{2.8}$$

The critical value K_c depends on the material and can be found experimentally. Its value varies with respect to the specimen thickness, giving origin to three different cases:

- small thickness, hence plane stress condition.
- medium thickness, hence stress transition.
- elevated thickness, hence plane strain condition.

In the last one, the critical stress intensity factor, K_{Ic}, takes the name of fracture toughness since it can be seen as a measure of the fatigue material resistance. When its value is high, the fracture due to crack propagation is harder to achieve. Based on that, a similar criterion can be determined for the J-integral. Under the assumption of plain strain and loading mode I, its critical value becomes:

$$J_{\mathrm{Ic}} = \frac{1-\nu^2}{E} K_{\mathrm{I}c}^2 \tag{2.9}$$

Equation 2.9 represents the material property which identifies the fracture condition in terms of energy release rate (Gdoutos, 2020).

2.4 Crack Propagation Under Cyclic Loading

As formerly described in section 2.1, aircraft work always within cyclic load conditions and this creates crack nucleations inside the structures. Successively, the crack starts to propagate up to a certain point where the propagation is too long and it reaches the fracture of the component. Fatigue design aims to find the component service life and this is done by the study of load-mission over the time. Therefore, it is essential to know the crack propagation rate $\frac{da}{dN}$, which defines the "speed" of propagation with a dependency on the range of stress intensity factor ΔK (Rege & Lemu, 2017, p.2). A typical evolution of $\frac{da}{dN}$ with respect to ΔK is shown in Figure 2.5

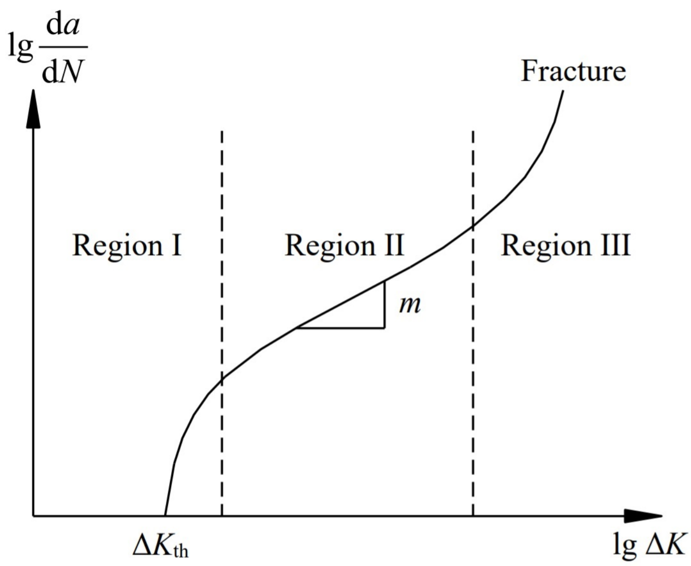

Figure 2.5: Fatigue crack growth curve (Rege & Lemu, 2017, p.2)

Usually, the crack growth rate curve can be divided into three regions:

- Region I: the crack propagates from the threshold value of stress intensity factor range ΔK on, with a very slow rate. Here the crack initiation takes place.
- Region II: the rate is linear (in double logarithmic scale) and can be predicted by the Paris law (Equation 2.10). For simplicity, this region is often expanded to the region I.
- Region III: here the component reaches the fracture condition. This region is also known as unsteady crack propagation range and is characterized by a rapid increase of $\frac{da}{dN}$.

The huge amount of variables which defines the crack propagation problem makes the crack growth rate difficult to predict. The first empirical model used to describe this phenomenon was the Paris law, shown in the Equation 2.10, where C and m are material constants. However, this law can be used just for the region II (Wolf, Revankar, & Riznic, 2009).

$$\frac{\mathrm{d}a}{\mathrm{d}N} = C\Delta K^m \tag{2.10}$$

There are other laws able to describe also the region I and III, e.g. the Forman law in Equation 2.11. It considers the critical value K_c and the crack closure effect by the presence of the stress ratio $R = \frac{K_{min}}{K_{max}}$ (Chernyatin, Matvienko, & Razumovsky, 2018).

$$\frac{\mathrm{d}a}{\mathrm{d}N} = \frac{C(\Delta K)^m}{(1-R)K_c - \Delta K} \tag{2.11}$$

Unfortunately, the Forman law does not take into account the crack initiation region. On the other hand, other laws do it, e.g. the NASGRO law (Equation 2.12). This equation is one of the most accurate crack growth models since it covers all the three regions and the crack closure effect. C and m are the same material constants coming from the Paris law, whereas p and q are the exponents describing the curve in the crack initiation and unsteady propagation regions, rispectively. Please note the presence of the threshold value ΔK_{th}, as well as the ratio between the maximum stress intensity factor and the critical one $\frac{K_{\mathrm{max}}}{K_{\mathrm{c}}}$ (Wang et al., 2018, p.5).

$$\frac{\mathrm{d}a}{\mathrm{d}N} = C\left(\frac{1-f}{1-R}\Delta K\right)^m \frac{\left(1-\frac{\Delta K_{\mathrm{th}}}{\Delta K}\right)^p}{\left(1-\frac{K_{\mathrm{max}}}{K_{\mathrm{c}}}\right)^q} \tag{2.12}$$

The term f is the crack opening function. It depends on the material, the load conditions and the stress ratio R. For most of the materials, the crack closure effect is one of the requirements to respect for a crack propagation law. Indeed, Figure 2.6 shows that decreasing R, the crack closure becomes predominant, delaying the crack growth rate (W. Liu, Yang, Mu, Liu, & Yu, 2011, p.3).

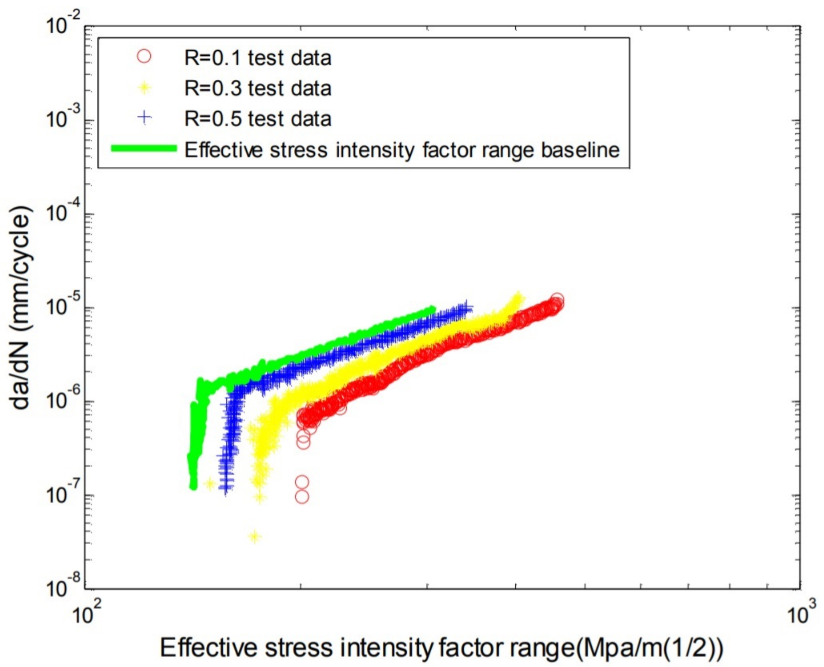

Figure 2.6: Crack growth rates with different stress ratios R (W. Liu et al., 2011, p.3)

Another law which describes the crack closure effect is the Walker law shown in the Equation 2.13, where C and m are the Paris material parameters in case of $R=0$ (Toribio, Matos Franco, González, & Escuadra, 2015, p.4).

$$\frac{\mathrm{d}a}{\mathrm{d}N} = C\left[\frac{\Delta K}{(1-R)^{(1-w)}}\right]^{\mathrm{m}} \tag{2.13}$$

The effective stress intensity factor range, defined as $\Delta K_{\mathrm{eff}} = K_{\mathrm{max}} - K_{\mathrm{op}}$ takes into account the crack growth rate change in case of overloads, where K_{op} is the stress intensity factor when the crack starts to open (Mcevil & Sotomi, 2002). This value is assumed to be $\Delta K_{\mathrm{eff}} = K_{\mathrm{max}}(1-R)^{w}$ by the Walker formulation and is taken as starting point for the study of the R-dependency in the crack propagation law proposed by MTU (Equation 2.14) (Dhondt, Rupp, & Hackenberg, 2015).

$$\frac{da}{dN} = \left[\left(\frac{da}{dN}\right)_{ref}\left(\frac{\Delta K}{\Delta K_{ref}}\right)^{m}\right]\frac{f_R \cdot f_{th}}{f_C} \tag{2.14}$$

This is a so-called multiplicative formulation, where the Paris range (between the square brackets) is multiplied by the corrective factors f_R,f_{th} and f_C. The first one, f_R, considers the previously cited R-dependency and assumes the following form:

$$f_R = \frac{1}{(1-R)^m}\left[1-\frac{1-\left(1-\frac{R}{g}\right)^{w}}{1-\left(1-\frac{1}{g}\right)^{w}}\right]^{m} \tag{2.15}$$

This is valid just for $R < 1$, since when $R > 1$ the crack is completely in the pressure range and does not propagate. $R = 1$ leads to a singularity of f_R, hence is not considered. The parameter g is a function of the temperature T, whereas w, the Walker exponent, is assumed to be constant when $R < 0$ and a function of T when $0 \leq R < 1$.

The crack initiation region is contained in the term f_{th}. It can be expressed with the Equation 2.16:

$$f_{th} = \begin{cases} 1-\exp\left[\varepsilon\left(1-\frac{\Delta K}{\Delta K_{th}(R)}\right)\right] & \text{for } \Delta K > \Delta K_{th} \\ 0 & \text{for } \Delta K \leq \Delta K_{th} \end{cases} \tag{2.16}$$

The parameter ε represents the curvature of the function.

Intuitively, f_C is the correction factor for the critical region. Its equation has the same form of Equation 2.16 and the parameter δ is its relative curvature factor.

$$f_C = 1-\exp\left[\delta\left(\frac{K_{\mathrm{max}}}{K_C}-1\right)\right] \text{ for } \quad K_{\mathrm{max}} < K_C \tag{2.17}$$

Therefore, it is possible to observe that this crack propagation law has the three regions (shown previously in Figure 2.5) distinctly placed in its equation and each corrective factor is independent from each other. This represents its biggest feature and it shows an outstanding prediction according to the experimental results (Dhondt et al., 2015).

3 Cracktracer3D

The software used to study numerically the crack propagation problems of this thesis is described in this chapter. In section 3.1, Cracktracer3D is briefly introduced with a comparison with other different crack propagation software. section 3.2 focuses on its preprocessor, Fe-solver and postprocessor. Each section shows all the steps executed by the software to achieve the crack propagation prediction. The last section outlines its input and output.

3.1 Software Introduction

In section 2.1 the need for aerospace companies to predict the crack propagation behaviour under cyclic loads has been widely discussed. To study the phenomenon of fatigue, due to its high complexity, a tool that works with numerical predictions is required. For fracture mechanics, many tools are currently on the market and they all aim to find the right crack propagation direction. To achieve that, there are different ways to approach the problem. Many of the them work by means of the Finite Element Method, e.g ZENCRACK (*ZENCRACK*, 2018), FRANC3D (Ingraffea, Wawrzynek, Carter, & Ibrahim, 2020) or ADAPCRACK3D (Schöllmann, Fulland, & Richard, 2003). However, other techniques are also used, such as the Boundary Element Method (BEM), implemented in BEASY (*BEASY*, 2020) or the Extended Finite Element Method (XFEM), implemented in Abaqus (Kim, Lee, Kim, & Kim, 2019). Using the FEM, the programs start from the uncracked structure mesh. Then, step-by-step, the crack evolution is inserted inside the model and its current stress intensity factors are computed. In the case of the Boundary Element Method, the solution is formulated with the boundary conditions and just the boundary of the model is meshed (Dhondt, 2014). Last but not least, the Extended Finite Element Method, which works with a different approach: the crack propagation path is found without re-meshing the model (Bhattacharya, Singh, & Mishra, 2013).

The idea of calculating the crack propagation into aircraft engine components in a fully automatic way was already born many years ago (Dhondt & Mångård, 2009) (see also (Mångård & Dhondt, 2007)). Over the years, at MTU has been developed Cracktracer3D, an in-house tool that operates iteratively using the Finite Element Method to calculate the stress intensity factors and the relative crack propagation in case of cyclic load conditions. It studies the crack evolution under low-cycle fatigue (LCF) and high-cycle fatigue (HCF). The main difference between the previously cited software is that the Cracktracer3D mesh of the crack propagation domain in the model is not dependent on the uncracked mesh. Moreover, the uncracked mesh has no constraints for the type of element: it can be meshed with hexahedral element, as well as with tetrahedral elements (Dhondt, 2014).

Figure 3.1 shows the working process of Cracktracer3D. It is composed of three main parts: the preprocessor, the FE-solver and the postprocessor (in section 3.2 they are described in depth). The preprocessor receives the user input for the problem to study. The initial input consists of the FE input deck of the uncracked structure, the initial crack geometry, crack propagation data and the definition of the crack propagation domain. All input and output is well described in section 3.3. Once the input has been processed by the preprocessor, it meshes the crack propagation domain with the crack inserted into the structure and sends the new FE input deck to CalculiX, the FE-solver. The stress field for the cracked structure gets solved and the results are taken by the postprocessor. This latter has to analyse them, determining the stress intensity factors along the crack front and calculating the crack propagation direction and progress length. The crack geometry is then updated and it becomes the new input for the next iteration, restarting from the preprocessor, up to the fulfilment of a stopping criterion (Dhondt, 2014). This can be the maximum number of iterations, the maximum number of loading cycles, the stress intensity factor being lower than the threshold value or equaling its critical value (see section 2.4).

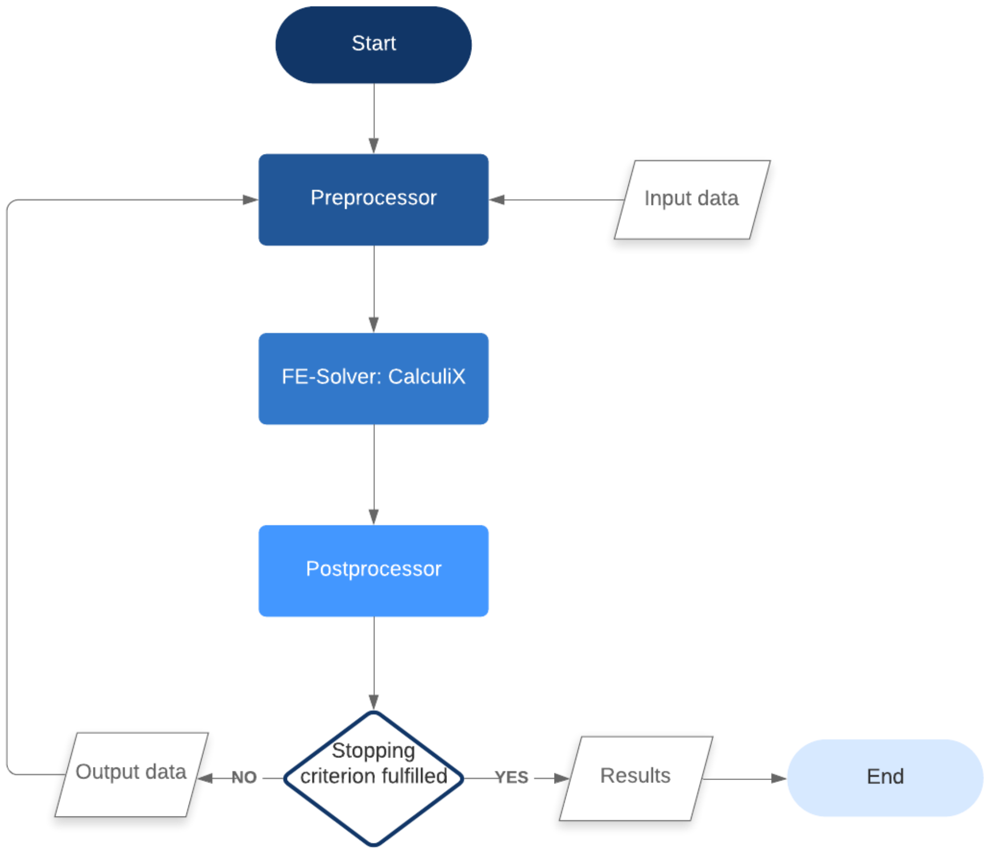

Figure 3.1: Organigram of Cracktracer3D

3.2 Working Principle

3.2.1 Preprocessor

The preprocessor has the main function of taking the input given by the user and creating the FE-model with the current crack inserted into the structure. The working principle can be easily understood looking at Figure 3.2, where it is possible to observe the uncracked structure on the left and the final FE-model with the initial crack on the right.

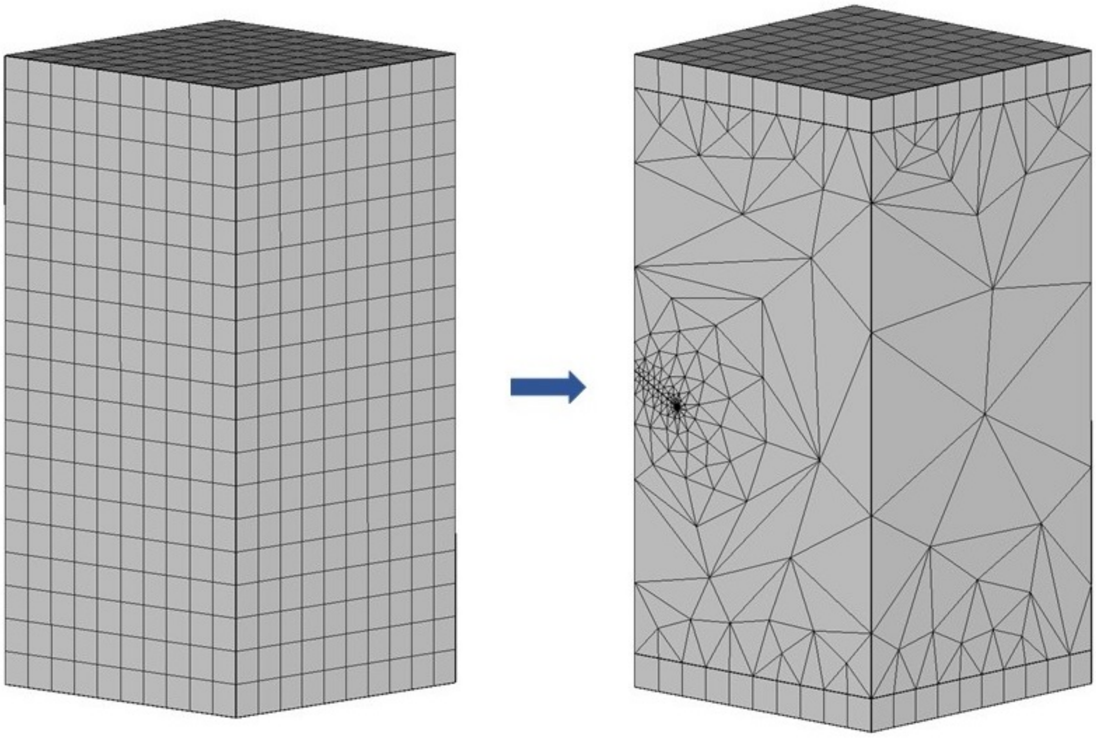

Figure 3.2: On the left the uncracked structure, on the right the remeshed structure with the initial crack

At the crack tip, the $\frac{1}{\sqrt{r}}$ singularity (see section 2.2) has to be modeled by the mesh. To do that, Cracktracer3D creates a tube of 20-node hexahedral elements with reduced integration points along the crack front. These elements are collapsed when they are adjacent to the crack tip and as shown in (Dhondt, 1993, p.20), they recreate accurately the tip strain singularity. They are also called quarter-point elements since the nodes in the middle are placed on a quarter-point position.

However, the tube may represent an issue in case of a change in curvature of the crack propagation path. When the crack curvature is very high, there could be the risk of the intersection with the tube, in case this is kept with a constant radius. Indeed, the preprocessor operates adapting the tube radius for each increment, in case it is needed. Moreover, another problem occurs in case of intersection with the free surfaces of the component. When these surfaces and the crack front are not orthogonal, there would be a generation of degenerated elements. To avoid that, the preprocessor keeps the elements of the tube always locally perpendicular to the front, so that the mesh remains regular. Unfortunately, this reduces the accuracy of the stress intensity factors next to the free surfaces(Dhondt, 2014, p.4). The tube generation for the case of circular corner crack is shown in Figure 3.3.

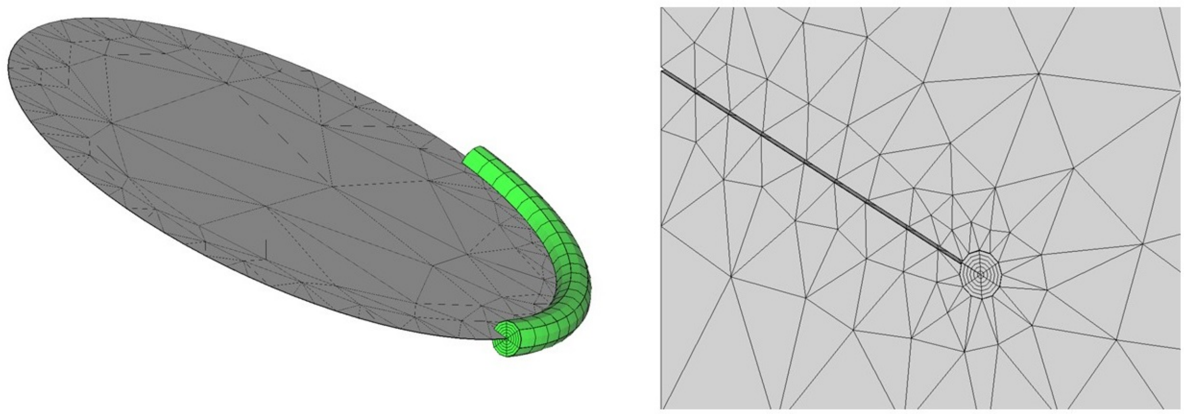

Figure 3.3: Crack front tube and insertion into the structure

Once the tube has been created, all the rest of the crack propagation domain is meshed with tetrahedral elements by NETGEN (*NETGEN*, 2019). A pure hexahedral mesh around the crack tip and all over the crack propagation domain has also been tested (Dhondt, 2005). However, a combined approach with hexahedral and tetrahedral mesh, as currently implemented in Cracktracer3D, shows better results (Mångård & Dhondt, 2009, p.9).

The tetrahedral mesh is connected to the hexahedral elements of the tube and the boundary elements of the domain by multiple point constraints. Once this is done, the preprocessor interpolates the temperature and the residual stresses defined for the uncracked structure. Since the new mesh is different, this step is essential.

3.2.2 FE-Solver

In the previous section it was described how the FE-model is built. For the FEM analysis, Cracktracer3D calls CalculiX, which in turn is composed of two parts: the preprocessor and postprocessor CalculiX GraphiX (Wittig, 2020), and the solver CalculiX CrunchiX (Dhondt, 2020). Once the input deck for the cracked structure is processed, CalculiX solves its stress field. The solution, written in a dedicated file, is the input for the postprocessor of Cracktracer3D.

The FEM software has the main routine written in C and all its subroutines written in FORTRAN and C. It can be used not only for structural problems but it solves many other numerical analyses as thermodynamic or fluid-dynamic calculations. More information can be found on its web site (Dhondt & Wittig, 2020).

In Figure 3.4 the stress component σ_{zz} distribution can be observed. In this example (see section 3.3 for more details), it can be noticed how the maximum stress is concentrated around the crack tip. The stress solution has to be very accurate in this region since the postprocessor will use it to calculate the crack propagation progress and its direction.

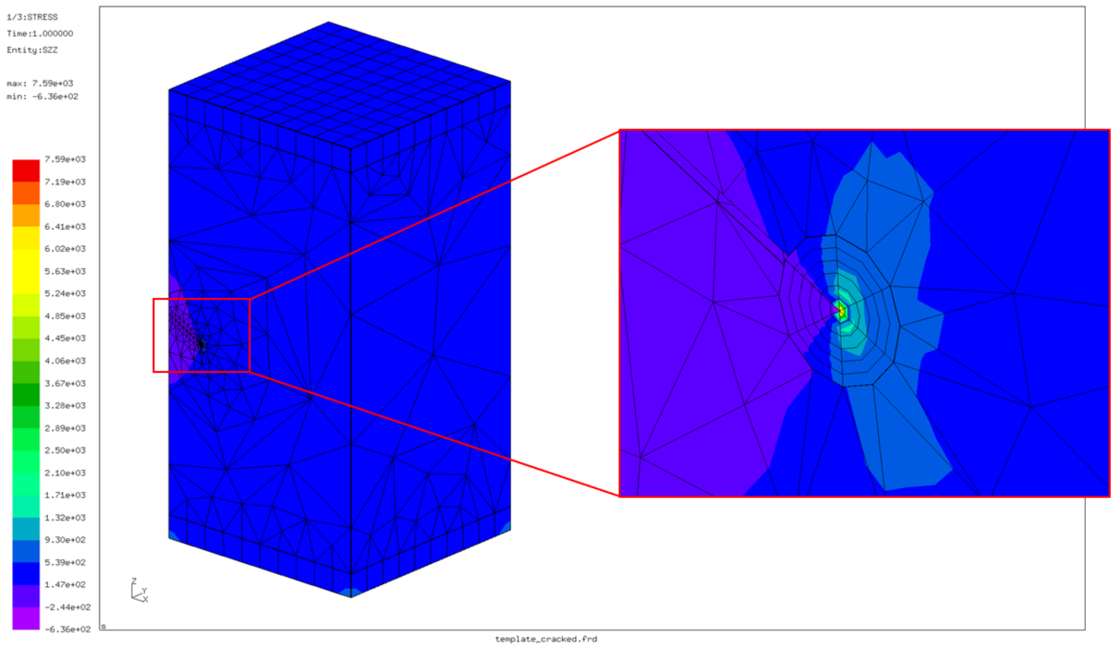

Figure 3.4: σ_{zz} distribution with a close-up view at the crack tip

3.2.3 Postprocessor

At first, the postprocessor uses the stress field solution to find the stress intensity factors along the crack front. To achieve that, it uses a numerical method called quarter-point elements stress method (QPES). There are many numerical ways to compute the stress intensity factors: some of them use the energy release rate e.g. the virtual crack extension method or the virtual crack closure technique; others use the displacement field e.g. the displacement extrapolation method or the J-integral e.g. the interaction integral method (IINT). One of the pros of the QPES is that it calculates the stress intensity factors from the stress field, which is easier to implement than an energy method. On the other hand, the stress intensity factors may be less accurate compared with the ones from an energy method (Dorca, 2018). The Figure 3.5 shows the integration points around the crack tip. There, the stress field is known thanks to the tube mesh (see subsection 3.2.1).

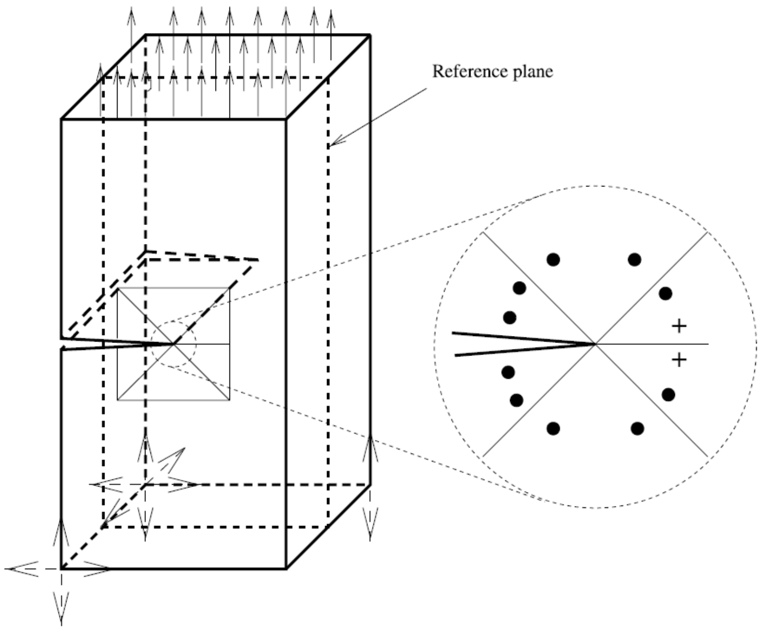

Figure 3.5: Integration points (Dhondt, 2002)

From section 2.2 the linear elastic asymptotic stress field can be found in the form shown in Equation 3.1.

$$\sigma = \frac{1}{\sqrt{r}} f\left(K_I, K_{II}, K_{III}, \varphi\right) \tag{3.1}$$

Since the position of the integration points (r and φ) and its stress tensor are known, Equation 3.1 can be solved to find the three unknown K_I, K_{II} and K_{III}. The equations available are six, one for each stress tensor component. Then, the system is overdetermined. Since the component parallel to the crack front is different for plane stress and plane strain, this is not considered in the system. Subsequently, the three stress intensity factors are found solving a system of five equations with the least-squares method. The mean values of the stress intensity factors between the integration points marked with a "+" in Figure 3.5 represent the stress intensity factors used for the crack propagation. This procedure is executed for all the elements along the crack front yielding the stress intensity factors distribution (Dhondt, 2014). A more detailed description of the QPES method is stated in (Dhondt, 2002) and a comparison with the IINT method is shown in (Dhondt, 2001).

The next step done by the postprocessor is the determination of the crack propagation direction and rate. For the direction, the stress intensity factors distribution is used to determine the orientation of the crack propagation plane, defined by the geometrical angles φ_0 and ψ_0 as represented in Figure 3.6.

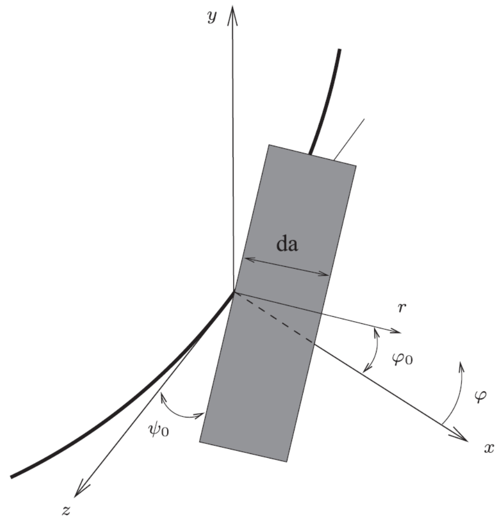

Figure 3.6: Crack propagation plane and crack tip coordinate system (Dhondt, 2014)

From Equation 3.1, by multiplying it with $\sqrt{r}$, the self-similar stress field σ^* can be obtained, which not coincidentally has the dimension of MPa$\sqrt{\mathrm{m}}$ as the stress intensity factor. The self-similar stress field depends only on φ since the dependency from r has been previously cancelled. This means that for a given value of φ, the six independent components of σ^* are known and the three local self-similar principal stresses can be determined. The criterion adopted by Cractracer3D to find the crack propagation direction is based on the assumption that the crack will propagate in a plane perpendicular to the largest self-similar principal stress and described by the angle φ_0 (Dhondt, 2002). However, the crack propagation plane needs to pass through the crack front, hence the angle φ must be equal to φ_0. Indeed, the postprocessor varies φ, obtaining different values for σ^* and φ_0, until the condition $\varphi = \varphi_0$ is fulfilled. The corresponding self-similar principal stress and its magnitude represents the equivalent stress intensity factor K_{eq}, which also leads to the value of the twist angle ψ_0 (Dhondt, 2014, p.8). Once the direction is known, the postprocessor calculates the crack growth rate for each node on the crack front by means of the previously cited crack growth models in section 2.4.

Now, the crack propagation progress can be added to the current crack front, updating the new input for the preprocessor and restarting the iterative process. However, from a computational point of view, this would to too expensive in time because of the huge number of iterations to perform. This number of iterations would match exactly with the number of loading cycles. Due to the fact that between consecutive loading cycles the stress intensity factors do not change so much, the postprocessor uses a maximum crack propagation increment as a distance within which the K-distribution is constant. This reference value represents 20% of the crack front tube radius. Hence, a relationship between the number of loading cycles and the crack front tube radius can be determined as shown in Equation 3.2, where $\left(\frac{da}{dN}\right)_{max}$ is the maximum crack growth rate among the crack front nodes.

$$N_i = \frac{0,2 \cdot r_{tube}}{\left(\frac{da}{dN}\right)_{max}} \tag{3.2}$$

The crack length increment for the k-th node on the crack front is:

$$\Delta a_{i,k} = \max\left\{N_i \cdot \left(\frac{da}{dN}\right)_k ; 0,1 \cdot r_{tube}\right\} \tag{3.3}$$

This ensures the crack front smoothing even in the case where some of the nodes have a propagation equal to zero. If the stopping criterion is not reached, the new crack front is sent to the preprocessor and a new iteration starts (Schrade, 2011). For the example in Figure 3.7, the maximum number of iteration equal to 50 has been chosen as a stopping criterion. The figure shows the updated geometry by the preprocessor for iteration 51.

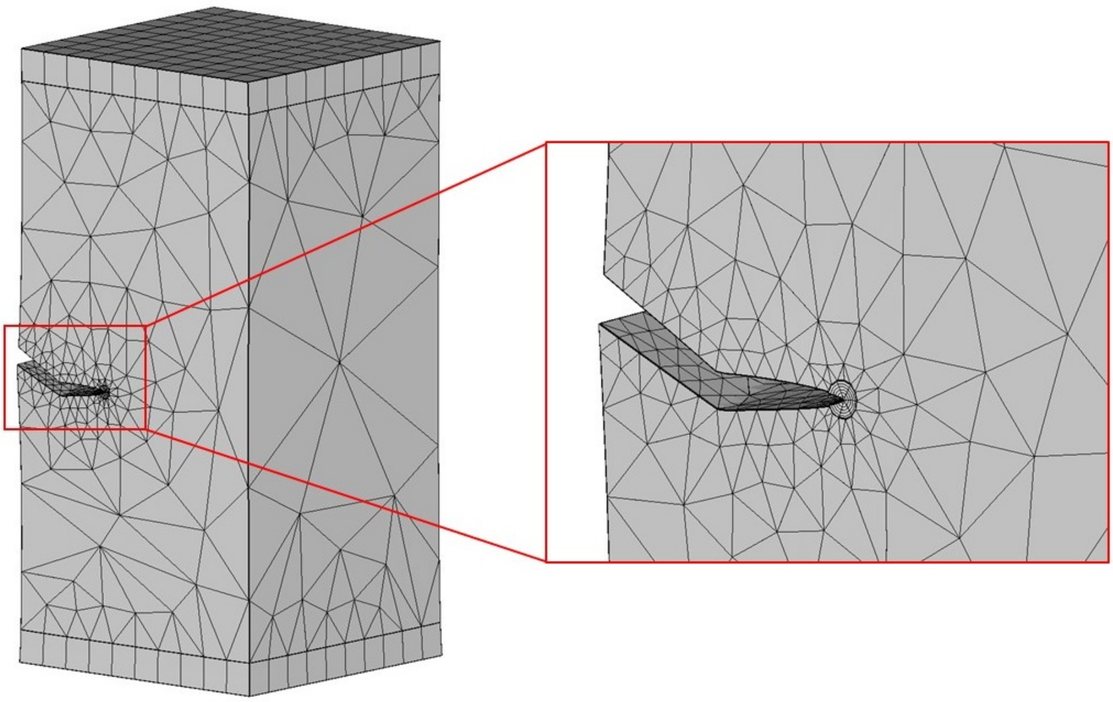

Figure 3.7: Crack propagation after 50 iterations

3.3 Input and Output

In Cracktracer3D, the user has to run the program giving four input files:

- The FE input deck (Abaqus format) of the uncracked structure. It matches the CalculiX input deck form. In fact, inside this file it is possible to find the mesh definition, the elastic and thermal proprieties and the boundary and load conditions.
- The crack propagation domain. This is a set of elements of the uncracked structure which contains the initial crack. As explained in subsection 3.2.1, this part of the model will be re-meshed with tetrahedral elements, taking into account the presence of the crack. The choice of this domain is done to save computational time, instead of re-meshing all the model.
- The initial crack geometry. In this file the crack meshed (Abaqus format) with triangular elements can be found. In the case of elliptical and rectangular cracks, it is possible to define the geometry with just a few geometrical parameters.
- The crack propagation data. This file contains material information, e.g. all the parameters to use for the crack propagation laws and program settings, e.g. the number of elements along the crack front tube or its maximum radius.

In Figure 3.8 the input for the case of a circular corner crack with a tensile load, is clearly shown.

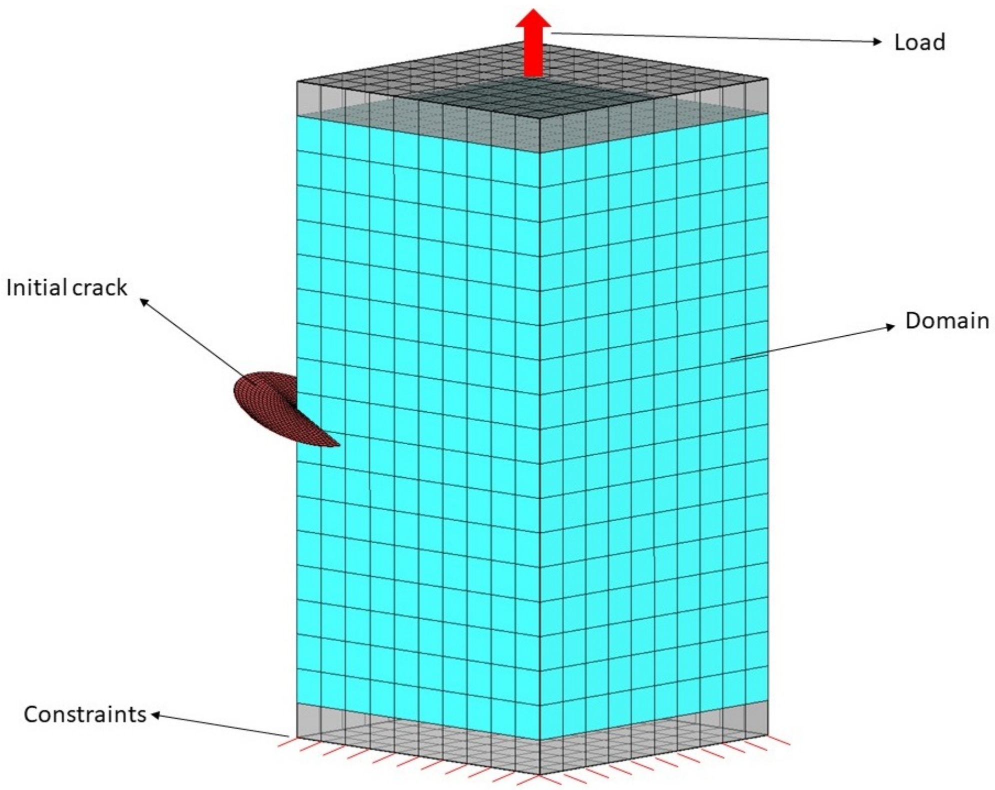

Figure 3.8: Uncracked structure with input definition

A lot of output is generated by the preprocessor, the FE-solver and the postprocessor. Most is used for debugging, in case an error is encountered. A useful output given by the postprocessor is the crack geometry: this allows to study whether the crack propagates toward critical regions and as said in subsection 3.2.3, this is an essential input for the next iteration. Over the crack surface, it is also possible to show with a fringe plot some crack proprieties such as K_I, K_{II}, K_{III} and ΔK_{eq}, or other loading cycle proprieties as the stress ratio R and the crack growth rate distribution. Moreover, the crack propagation can be also described through some graphs which list the number of loading cycles for each iteration, as well as the maximum crack length and the maximum crack growth rate with respect to N. For the example reported in Figure 3.8, the crack surface after 50 iteration is shown in Figure 3.9 and its ΔK_{eq} distribution in Figure 3.10.

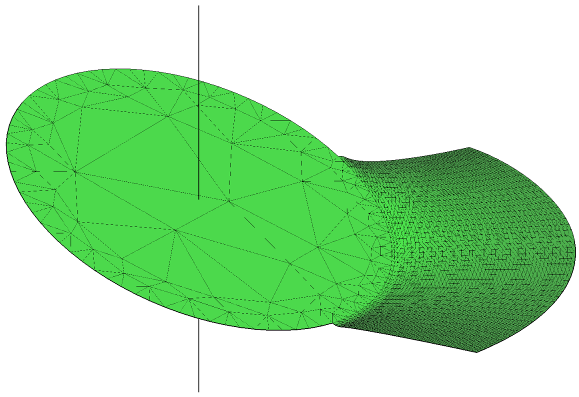

Figure 3.9: Crack surface geometry after 50 increments

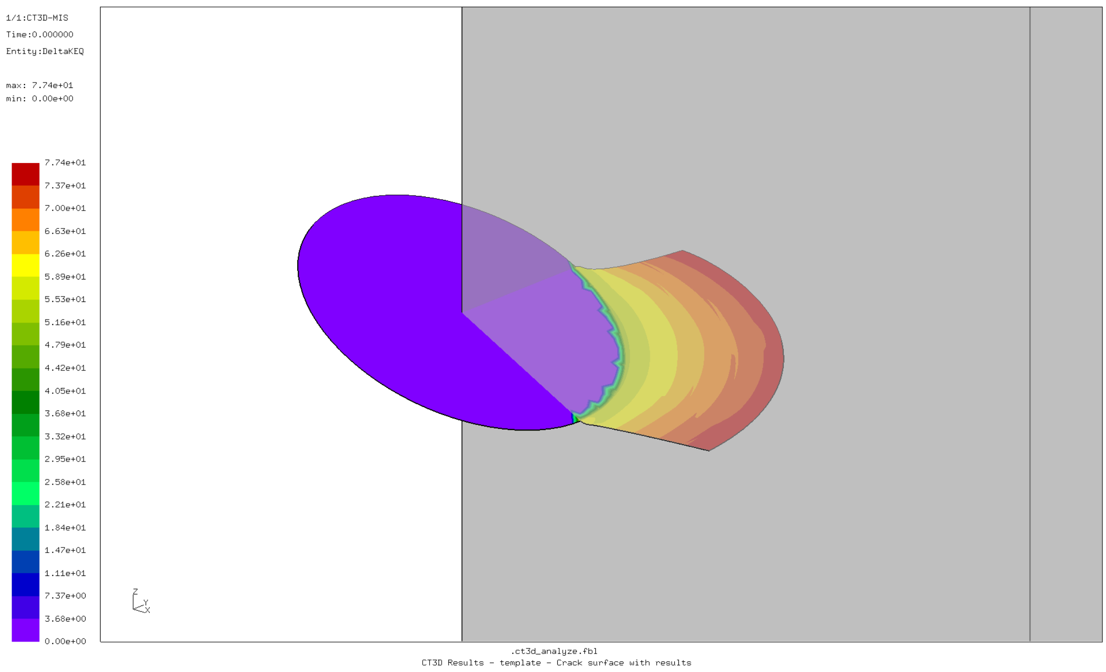

Figure 3.10: ΔK_{eq} distribution on the crack surface

According to the crack evolution, it is clear that due to the slanted initial position, the propagation starts in mixed-mode, but afterwards it twists and tends to propagate orthogonal to the tensile load, increasing the ΔK_{eq} and the crack growth rate. Figure 3.11 shows the maximum crack length versus to the number of loading cycles N.

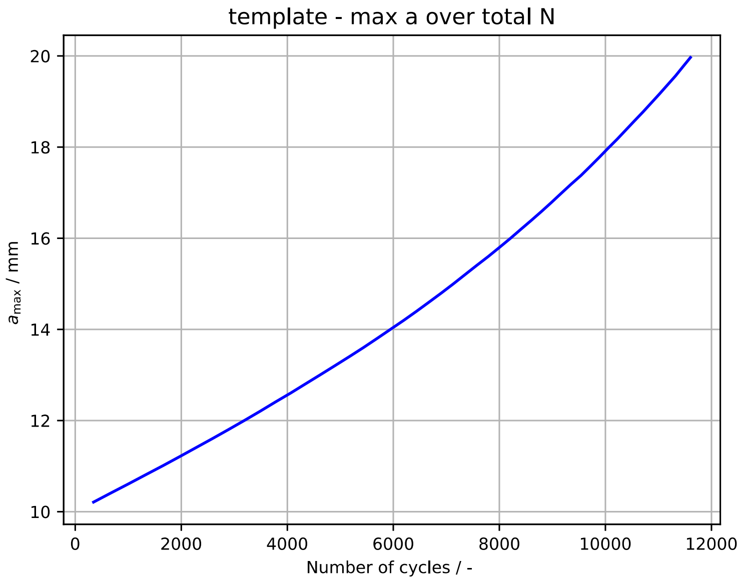

Figure 3.11: Maximum crack length a_{max} over the number of cycles N

4 Cracktracer3D Validation and Modification

In this chapter, the specimens analysed for the validation of Cracktracer3D are described. More specifically subsection 4.1.1 is related to the 4-point bending specimen and subsection 4.1.2 to the tension-torsion specimen. Moreover, a new crack propagation criterion is introduced in section 4.2 for a comparison with the current one already implemented into the software. In section 4.3 the tool used for the validation is outlined.

4.1 Specimens Analysed

In chapter 3 Cracktracer3D, the MTU in-house software for crack propagation, has introduced. As every simulation software, to ensure the reliability of the predictions, experimental tests have to be performed for a comparison of the results. Sometimes, for simple analysis, a prior validation can be made with an analytical solution. However, for crack propagation problems, these solutions are found subject to many approximations and they always consider ideal cases, which of course do not represent the reality. Cracktracer3D aims to predict the crack propagation in aero-engine components, where the experience shows a mixed-mode behaviour (see chapter 2). In the literature, there are some models and corrections for it, e.g. (Haefele & Lee, 1995), but the number of applications is very limited. Hence, the unique way to proceed is to resort to experiments. For this thesis, two different types of specimens have been used. The goal is to study different combinations of load and temperature to validate the mixed-mode crack propagation prediction of Cracktracer3D. Specifically, they are described below.

4.1.1 4-Point Bending Specimen

The 4-point bending specimens are well known in the field of experimental fracture mechanics. This type of specimen allows studying the crack propagation with a bending load of the structure. In the literature, it can be classified as a single edge notched bend specimen (SENB). 16 specimens with different load conditions and temperatures have been tested in Stockholm, at the KTH Royal Institute of Technology. In Figure 4.1 the specimen geometry can be observed. To study the mixed-mode crack propagation, the notch is placed with 45° respect to the symmetry line. The dimensions are relatively small: the cross-section has a nominal width of 5 *mm* and a height of 15 *mm*, the global length measures 70 *mm*. The notch is placed in the middle, with 3 *mm* of height and a tip angle of 80°. The four characteristic points of this specimen are the two supports and the two loading points. In the reality, they are the points of contact between the body and a roller. Hence, for the test, four rollers are used, where two of them are the supports placed below the specimen (in Figure 4.1 the red triangular prisms). The remaining two are the application points of the force, which due to a loading symmetry, count as half of the global force. The force is applied exactly in the middle by a hydraulic machine and homogenously distributed on two ceramic rollers (in Figure 4.1 the red arrows pointing downward).

The most important geometrical parameter is the distance along the longitudinal axis be-

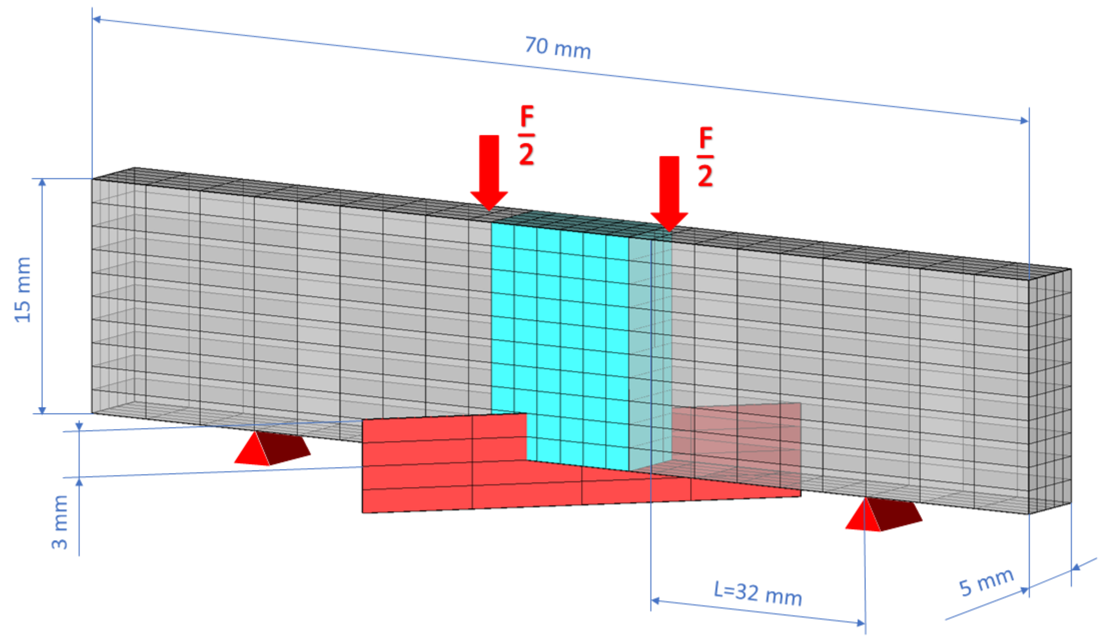

Figure 4.1: 4PB specimen model

tween the point of loading and the support. This is always kept constant for each test and determines the magnitude of the maximum normal stress at the crack tip, and consequently the stress intensity factors. By using the Navier equation for the beam bending theory, this stress component for the uncracked structure can be calculated with the Equation 4.1:

$$\sigma_{max} = \frac{6FL}{4BW^2} \tag{4.1}$$

The parameter L, equal to 32 *mm*, is the distance between the outer and inner rollers. B and W are respectively the width and the height of the cross-section and F is the force applied in the middle.

In Figure 4.1, the 4-point bending specimen model implemented in Cractracer3D is shown. The crack propagation domain for the input file can be seen in light blue and the location of supports and loads in red. It can be noticed that the notch is not modeled, but in its place the crack geometry is inserted. This approximation does not affect too much the results. The preprocessor reproduces the notch effect with the first iteration while meshing the domain. Al the rest of the component is meshed with hexahedral elements.

The tests have been performed with fixed temperature and fixed peak load ($R = 0.1$):

- The temperature, changed between +50°C and +350°C.
- The peak load magnitude, changed between 3.02 kN and 4.78 kN.

Just for a couple of specimens, the experiment has been repeated twice. The combination of these two parameters is summarized in Table 5.1. The material is titanium alloy Ti6246, characterized by a good fatigue resistance. This type of material is used for lightweight applications requiring a high strength.

Before the final test, each specimen has been precracked to ensure that during the experiment crack propagation occurs and no extra number of loading cycles for the crack initiation are counted. The precracking phase consists of a cyclic min-max load with $R = 0.1$, $\Delta K = 14 MPam^{0.5}$ at $T = +24°C$. This was stopped after the initial crack of 0.4 mm was reached. The final test for the crack propagation study involves a fixed loading sequence as shown in Figure 4.2, where just the amplitude is varied. At first, there is a ramp from 10% of the peak load toward its full value. Then, this is kept constant and then it decreases again to 10%, remaining constant for 1 second before restarting the sequence. Each step described lasts 4 second.

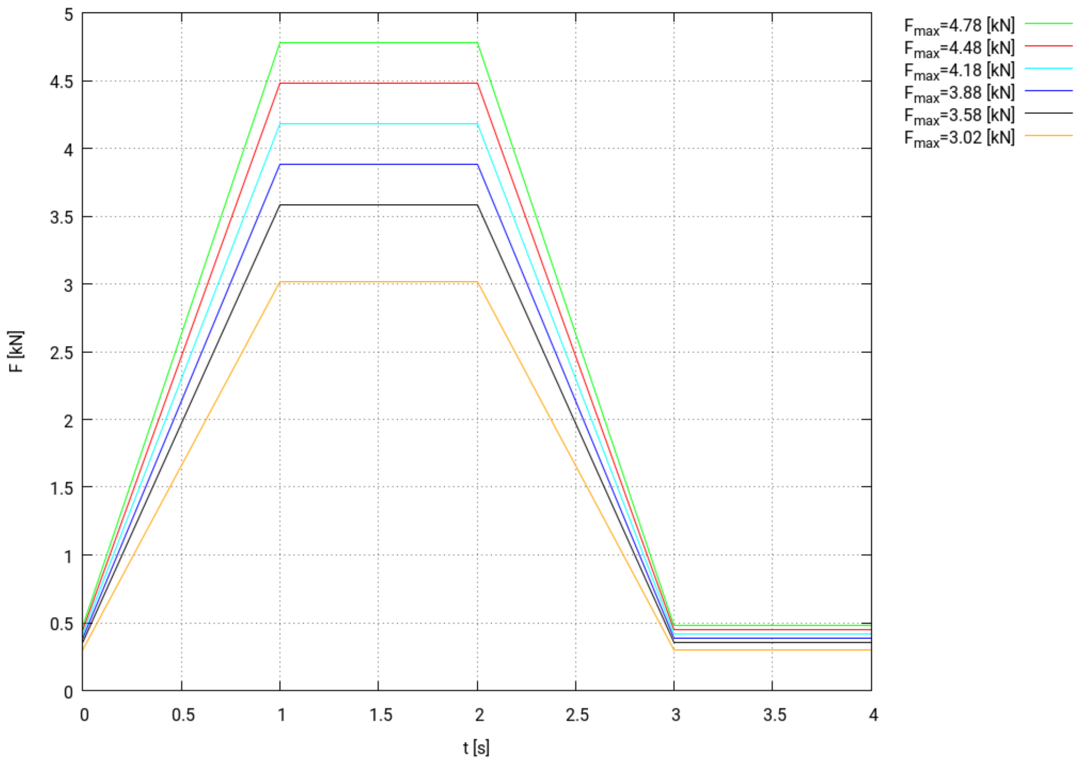

Figure 4.2: 4PB specimen load conditions

To be able to measure the crack propagation length over the number of loading cycles, the marker load technique has been used. The latter produces some marks on the crack surface by changing the load sequence for a specific instant. In this case, the loading sequence has been changed to a sinus loading cycle with a frequency equal to 5 Hz and $R = 0.7$. For some specimens, a constant time interval has been fixed to create these marks, whereas, for others, the markig times have been chosen trying to keep the crack increment constant between them. An ideal representation of these marks is shown in Figure 4.3, where the red lines are the so-called beach marks and represent the crack front after a specific number of cycles. To compute the crack growth rate the MTU law, already discussed in Equation 2.14, has been chosen.

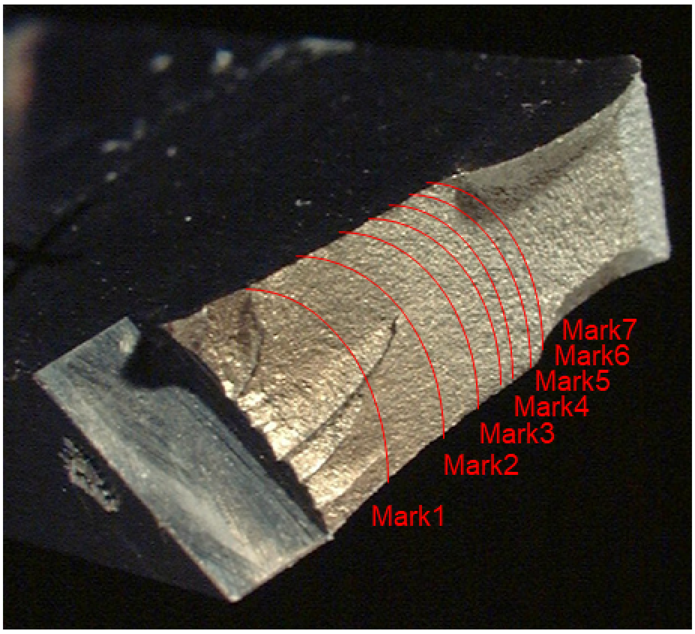

Figure 4.3: Ideal representation of the marker load technique on the 4PB specimen crack surface

4.1.2 Tension-Torsion Specimen

The tension-torsion specimens allow to study the crack propagation with a combination of sinus tensile and torsional loads, hence mixed-mode crack propagation. For this specific application, the experiments aim to show the crack behaviour in a load mission, varying the phase between the two loads. The specimens can be classified as single-edge notched specimens. 36 specimens have been tested at the Universität Rostock (Köster, Benz, Heyer, & Sander, 2020). Their geometry is shown in Figure 4.4, where is possible to observe a similarity with the 4-point bending specimen. However, in this case, the dimensions are much bigger. The cross-section has a width of 10 *mm* and a height of 50 *mm*, whereas the length along the longitudinal axis is 230 *mm*. The notch is included in the model and it is identified by a tip angle of 90°, a height of 8 *mm* and a width of 3 *mm*. The body can be divided into 3 parts: left side, where the loads are applied; the central part, i.e. the crack propagation domain; right side, where the body is fixed. To set up the calculation in Cracktracer3D, the two sides are modeled using a rigid body approximation. More specifically, the right part is wedged. On the left side, along its longitudinal axis, the tensile force and the torsional torque around it are applied. The tensile load is used to generate mode I at the initial crack tip, fracturing the structure symmetrically in opening mode, whereas the torsional torque induces shear stresses, with subsequent loading mode II and III. An initial crack of 0.3 *mm* as been considered.

It is possible to calculate the maximum nominal stresses caused by the loads. For the tensile force, the normal stress can be easily found with Equation 4.2:

$$\sigma_N = \frac{F}{A} \tag{4.2}$$

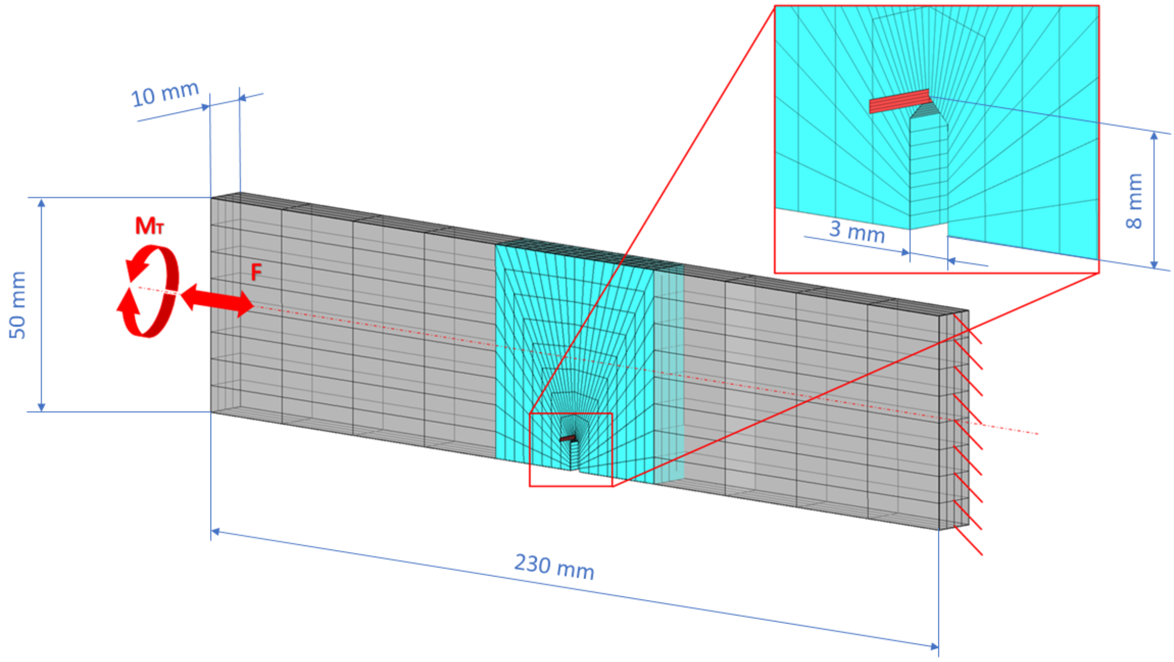

Figure 4.4: Tension-torsion specimen model

where A is the cross-section area.

Following the same approach of the beam theory, the τ_N is calculated with the equation Equation 4.3:

$$\tau_N = \frac{M_T}{kbh^2} \tag{4.3}$$

where M_T is the torsional moment magnitude, b and h the cross sectional area dimensions, and k the torsional parameter which depends on the ratio $\frac{b}{h}$. For the given dimensions, k is equal to 0.29. In Equation 4.4 the dependency of the ratio $\frac{\tau_N}{\sigma_N}$ on M_T and F is shown.

$$\frac{\tau_N}{\sigma_N} = 0.345\frac{M_T}{F} \tag{4.4}$$

This ratio can have three different values for the tests: 1.31, 1 and 0.76. The amplitude of the tensile force is kept constant for each experiment (F=27500 N). Hence, the three different amplitudes of the torsional moment are respectively: 104500 Nmm, 80000 Nmm and 60500 Nmm.

In Figure 4.5 all the loading configurations used for the specimens are shown. For the axial force, just the stress ratio R_{axial} can vary between 0 and -1. For the torsional torque, it is kept always equal to -1. In Table 5.4 the combinations of τ_N and σ_N are reported. For some load cases, the test has been repeated more than one time.

The material of the specimens is steel 34CrNiMo6, characterized by high strength. Its parameters for the fatigue analysis are shown in Table 4.1 considering a probability of survival equal to 50%. For these specimens, to compute the crack growth rate the NASGRO law has been used (see section 2.4). However, the interest of these tests is in studying just the crack

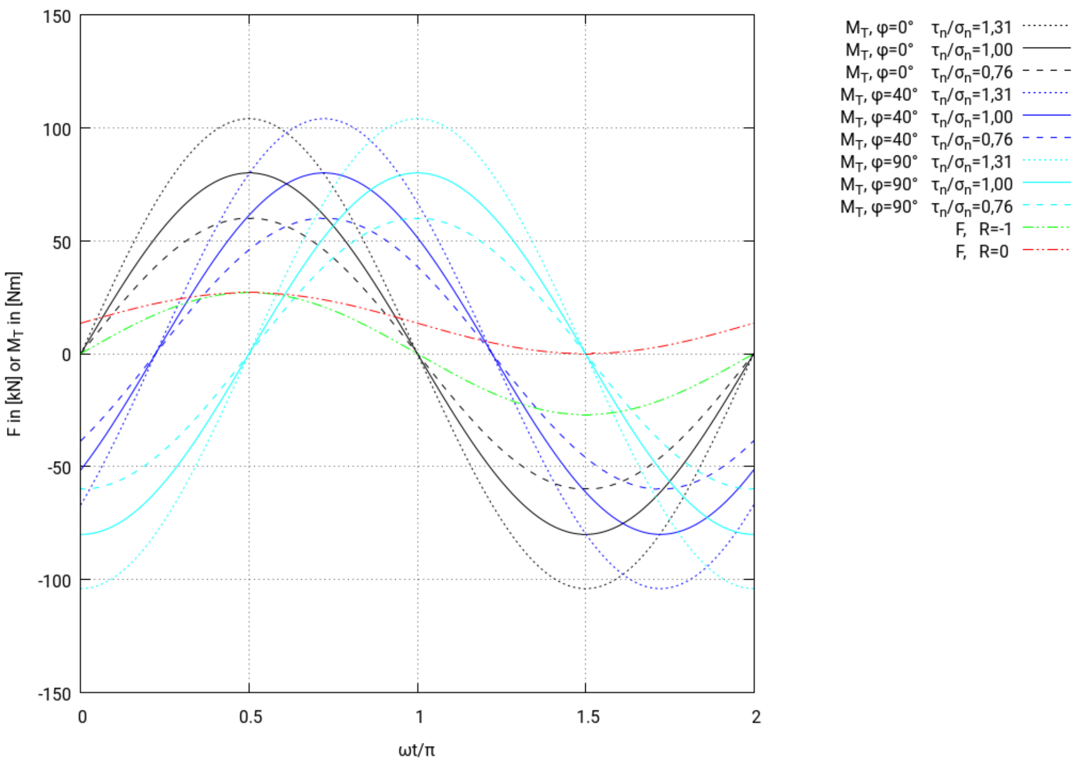

Figure 4.5: Tension-torsion specimen load conditions

propagation direction and not its evolution over the number of loading cycles. For this reason, the marker load technique has not been utilized during the load sequence. All the tests were performed in room temperature.

Table 4.1: Material parameters for 34CrNiMo6 ($P_S = 50\%$) (Hannemann, Köster, & Sander, 2017)

E $[GPa]$	A $[\%]$	R_m $[MPa]$
210	9	1200

C_{FM}	n	p	q	$\Delta K_{th,1}$ $[MPam^{1/2}]$	K_{IC} $[MPam^{1/2}]$	C^+_{th}	C^-_{th}	α_{CF}	$\frac{\sigma_{max}}{\sigma_F}$
$9.38\ 10^{-7}$	1.89	2.39	0.43	1.14	145	3.89	0.05	1.9	0.3

4.2 A New Crack Propagation Criterion

4.2.1 Current Approach

As previously stated in subsection 4.2.2, the tension-torsion specimens are utilized to study the mixed-mode crack propagation in case of missions. The change of mixed-mode conditions over time is typical for aero-engines during a complete flight (start-idle-climb-cruise-descent-thrust reverse-idle-shut down). Therefore, in Cracktracer3D there is a subroutine which takes this effect into account following the approach described in subsection 3.2.3. To understand the working principle, the crack growth rate law is considered as shown in Equation 4.6:

$$\frac{da}{dN} = \left(\frac{da}{dN}\right)_{ref} \left(\frac{\Delta K_{eq}}{\Delta K_{ref}}\right)^{m} \tag{4.5}$$

The Paris constant C can be written as:

$$C = \left(\frac{da}{dN}\right)_{ref} \left(\frac{1}{\Delta K_{ref}}\right)^{m} \tag{4.6}$$

The temperature has to be considered during the time as well, since ΔK_{ref} and m are a function of T. The postprocessor looks for the instant t_{max} which generates the maximum crack propagation growth. Taking as an example Figure 4.6, after having read the mission as a composition of 0-max cycles, the maximum is found. For that load condition, considering its maximum principal stress, the crack propagation direction is calculated and taken as the dominant one. For the other load steps, the relative principal stress is calculated considering the closest principal plane to the dominant one (Dhondt, 2011). This may result in a decrease of some values in the curve (dashed line in Figure 4.6), the highest peak, however, does not change.

Once the principal planes, hence the equivalent stress intensity factor, are calculated, the cycle extraction is performed on the curve in Figure 4.6 following the rainflow counting algorithm (Downing & Socie, 1982). In this way, each extracted cycle has a maximum and a minimum, and $\Delta K_{eq} = K_{eq}^{max} - K_{eq}^{min}$ is characterized by the corresponding temperatures $T_{K_{eq}^{max}}$ and $T_{K_{eq}^{min}}$. To find the crack growth rate, Equation 2.14 is applied. The crack propagation rate of the cycle (K_{eq}^{max},K_{eq}^{min}) is the maximum of it propagation at $T_{K_{eq}^{max}}$ and at $T_{K_{eq}^{min}}$ (for some materials the propagation rate decreases for increasing temperatures in certain temperature ranges). The crack propagation direction is always kept equal to the dominant one. The crack growth rate at the end of the cycle extraction is given by the sum of the crack growth rate of each extracted cycle (Dhondt, 2011).

For the tension-torsion specimens, as well as the 4-point bending specimens, dynamic phenomena are not considered since the load conditions represent a low-cycle fatigue problem. Therefore, the load mission is simulated with a sequence of static loading steps. More specifically, each load mission is divided into 13 steps as shown in Figure 4.7.

The static steps are placed into the input deck of the uncracked structure FE model. Cracktracer3D calls CalculiX to solve the stress field for each step and the output is the starting point of the procedure described above.

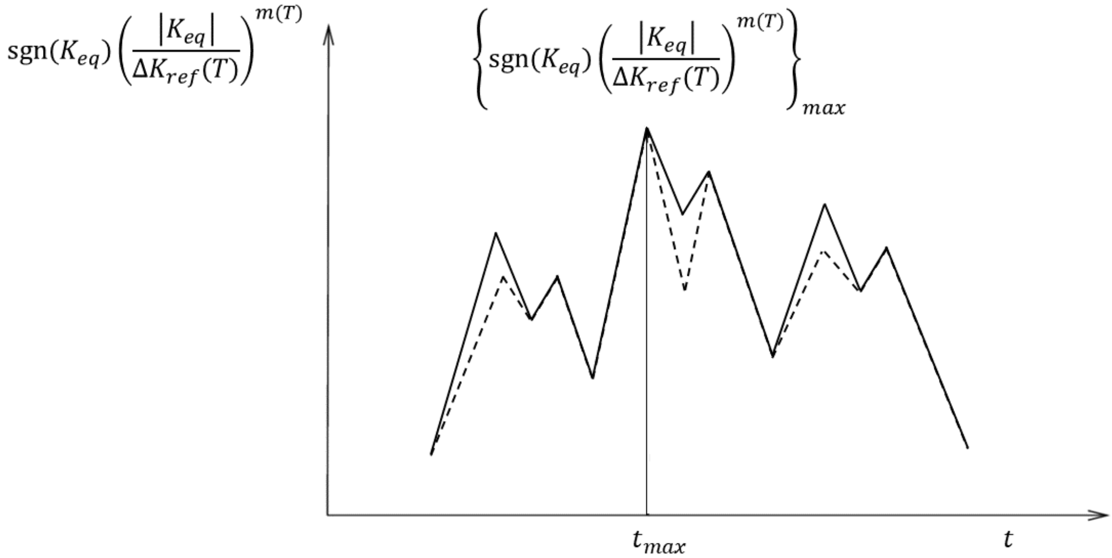

Figure 4.6: Dominant crack propagation step for the crack growth rate

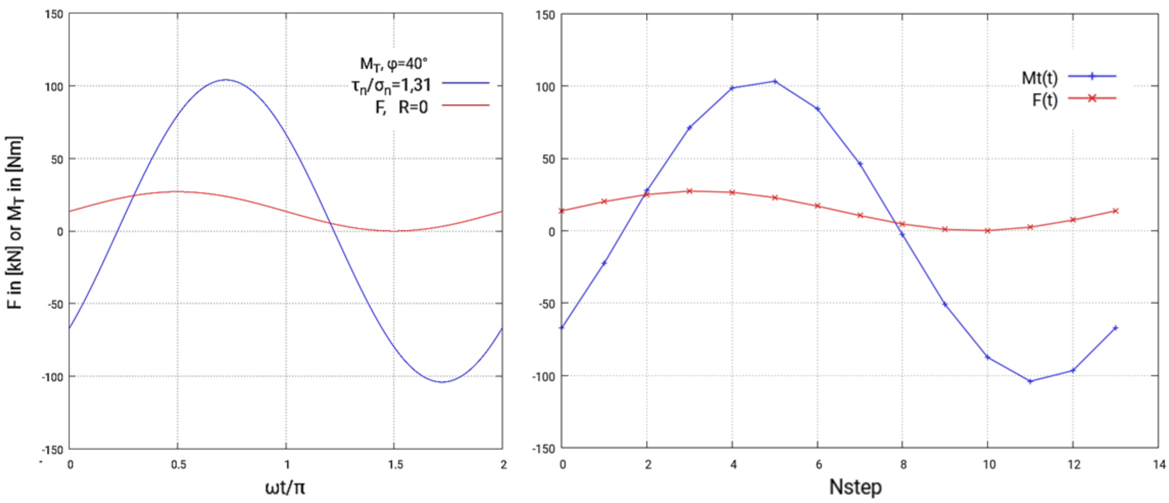

Figure 4.7: Load discretization in static steps

This method is based on assumptions to simplify the implementation of the problem. However this does not necessarily mean that it is incorrect, however, verification with the experiments is needed.

4.2.2 New Implementation

An alternative procedure to choose the crack propagation direction is shown in this subsection. In the literature many other existing methods can be found following different approaches. One of these is the maximum tangential stress criterion (MTS), which considers the crack propagation in the direction where there is the maximal tangential stress. Another one, based on an energetic approach, is the strain energy density criterion(SED). In this case, it is assumed that the crack will propagate in the direction with the minimum strain energy density. Considering the vector crack tip displacement as the driving force for the crack propagation, the crack tip displacement criterion (CTD) can be another possibility to achieve this target (Malíková, Veselý, & Seitl, 2016). However, a final consideration in the case of time-dependent loads has to be implemented. To prove and compare the validity of the current implemented method (described in subsection 4.2.1) a new idea has been formulated.

Instead of employing just the crack propagation direction of the dominant step, the new criterion aims to consider the contributions of all the other steps. In order to do that, the deflection angle identifying the crack propagation direction (see Figure 3.6), is taken as a weighted average of the crack deflection angles of each loading step with the related crack growth rate. Its definition can be seen in Equation 4.7.

$$\varphi = \frac{\sum_{i=1}^{Nstep} \varphi_i \left(\frac{da}{dN}\right)_i}{\sum_{i=1}^{Nstep} \left(\frac{da}{dN}\right)_i} \tag{4.7}$$

φ is the deflection angle which characterizes the crack propagation direction for a specific iteration. The index i identifies the $i-th$ loading step.

4.3 The Tool CT3D _Validator

4.3.1 Working Principle

To compare experimental and numerical results a new tool has been developed. CT3D_Validator aims to calculate some values which identify the grade of accuracy of Cracktracer3D with respect to the reality. More specifically, in order to measure the difference for the crack propagation direction, the program calculates the volume between the real crack surface and the numerical one. This volume is afterwards divided by the area of the Cracktracer3D crack surface. This provides the user with a length that defines a global deviation between the two directions. This length is actually measured along a direction chosen before by the user. In fact, when CT3D_Validator runs the calculation needs at first the direction for the measurement of the volume. Equation 4.8 shows how the global deviation is defined.

$$dev_{x,y,z,n} = \frac{\sum_{i=1}^{N} V_i}{\sum_{i=1}^{N} A_i} = \frac{\sum_{i=1}^{N} |d_i| A_i}{\sum_{i=1}^{N} A_i} \tag{4.8}$$

Four possible directions can be selected by the user: parallel to the three cartesian axes or the local normal direction on the i-th element of the numerical surface. N is the number of elements (triangles) of the Cracktracer3D surface. A_i is the triangle area of the i-th element and d_i is the value of the distance between its centre of gravity and the experimental surface, measured along the selected direction. This value is written as an absolute value, to mark that it does not matter whether the experimental crack is behind or in front of the numerical surface. The volume must be positive, otherwise, in some particular cases, a deviation equal to zero can be measured even though the experimental and numerical surfaces do not coincide. To give a more clear idea, a graphical representation of this value is shown in Figure 4.8.

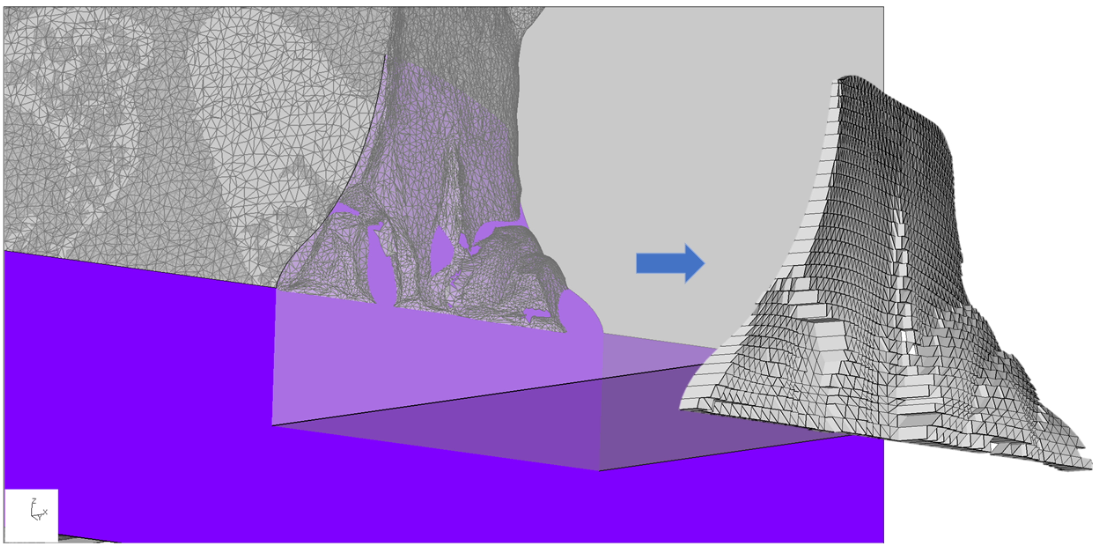

Figure 4.8: Discretized volume between the experimental crack surface and the numerical one (in blue)

Furthermore, this deviation is also calculated locally at the first and last front (by using the triangles adjacent to the first and last front). This can be useful in case the user wants to compare the differences just at the beginning of the propagation or at the end. Once these values are calculated, the accuracy for the crack propagation direction can be evaluated. However, most of the time, the real cases show many other imperfections such as the fact that the crack does not start to propagate from the notch tip or discontinuities (ridges) along the crack surface. This latter morphological phenomenon is called factory-roof effect. This mainly occurs in metallic materials because of their microstructure combined with the presence of loading mode II and III. It is characterized by many irregular patterns where microcracks propagate in mixed-mode (Pokluda, Slámečka, & Šandera, 2010). CT3D_Validator is also able to give a measure of this factory-roof effect following the same approach as done for $dev_{x,y,z,n}$. Since these imperfections involve only the experimental crack, its average crack surface is just calculated by the least-squares method. The assumption is that this surface can be at maximum a third-grade surface as defined in Equation 4.9.

$$f(x,y) = ax^3 + by^3 + cx^2y + dxy^2 + ex^2 + fy^2 + gxy + hx + jy + k \tag{4.9}$$

After the tool has found its coefficients, the volume of the factory-roof is approximately calculated with respect to the average crack plane. In the end, it is divided by the crack area (the same reference is used as before), generating a characteristic length called FR (factory-roof). The more factory-roof exists, the higher FR is. For the 4-point bending specimens, since there is also interest in comparing the crack propagation length over the number of loading cycles, this parameter is essential for clarifying some findings. Equation 4.10 shows how it is defined and in Figure 4.9 it is possible to observe the irregularities ahead and behind the crack average surface where the volume is measured.

$$FR = \frac{\sum_{i=1}^{N} \left| d_{i,experiment-average} \right| A_i}{\sum_{i=1}^{N} A_i} \tag{4.10}$$

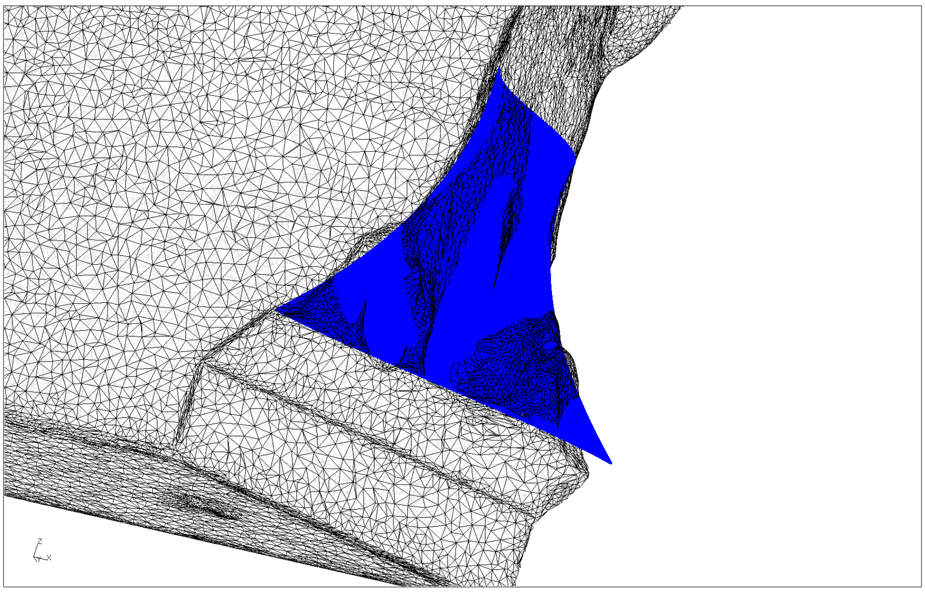

Figure 4.9: Average crack surface for a 4-point bending specimen

4.3.2 Input and Output

To get a better understanding of the tool, the input and output must be described more in detail. CT3D_Validator works with two input files. One is the crack geometry computed by Cracktracer3D, defined with a triangulated mesh (as e.g. in Figure 3.9). This geometry is the *.inc* output file of Cracktracer3D. The other one is the real crack geometry, also defined with a triangulation. Both the triangulations must be written in Abaqus format. The Figure 4.10 gives an idea of how this mesh is created. In this example, a 4-point bending specimen has been used to explain the process. The figure at the left shows the crack surface after the test. The specimen is broken in two parts and if these are complementary, then just one side can be used for the comparison. With a blue light scanning procedure, the component is analysed and the figure in the middle shows its 3D reproduction. Afterwards, this is meshed with triangles and the *.stl* file is generated. This file is converted into the Abaqus format, to be consistent with the file generated by Cracktracer3D.

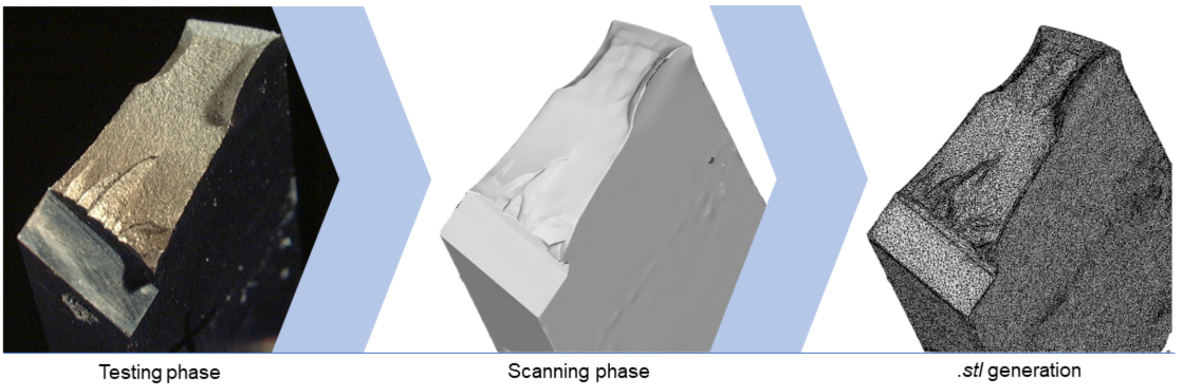

Figure 4.10: Experimental crack measurement phases, from the experimental surface to its triangulation

It is clear from the figure on the right that the experimental mesh contains many elements. This is the reason why the subroutine near3d.f from CalculiX is implemented in the tool. This allows saving time while measuring the deviation, looking for the k-nearest elements instead of checking every time at the whole set of triangles. When the curvature is very high, the number of neighbours k is increased, allowing CT3D_Validator to find the right element on the experimental triangulations. This is a fully-automatic process implemented in the main code. Unfortunately, for very irregular surfaces with protruding parts, the scanned surface may have some holes because of the scanner limitations. This is very rare, however, when the tool comes across these parts, the distance cannot be measured and the value of the distance is set to zero. This phenomenon occurs more for very small specimens, but it does not affect so much the results (for the analysed specimen this concerned always less than the 4% of the total number of elements N).

The outputs generated are mainly used for debugging and visualize graphically the results. CT3D_Validator generates six different output files:

- CT3D_Validator.log: all the steps performed by the tool are stored in this file. It contains all the information related to the input files and the numbers needed for the validation as the $dev_{x,y,z,n}$ global, at the first front and last crack front, and FR.

- lines.fdb: the .fdb files have to be read by CalculiX GraphiX in build-mode. This creates a graphical visualization of the segments representing d_i in Equation 4.8.
- prisms.fdb: this file generates the volume measured between the real crack and one predicted by Cracktracer3D, as shown in Figure 4.8.
- flfront.fdb: here just the volumes between the first and last crack front are represented.
- avesruf.fdb: a dense set of points recreates the average crack surface of the tested specimen.
- devfr.fdb: this file is very similar to the output lines.fdb. It shows the segments representing $d_{i,experiment-average}$ in Equation 4.10.

5 Results

In this chapter, the experimental results are compared with the numerical predictions made by Cracktracer3D. In section 5.1 the 4-point bending specimens results are analysed. The same is done in section 5.2 for the tension-torsion specimens. A further comparison with the new crack propagation criterion, described in the previous chapter, can be found in subsection 5.2.2.

5.1 4-Point Bending Specimen

In this section, the results for the sixteen 4-point bending specimens are shown. More specifically, a comparison between the crack propagation directions predicted by Cracktracer3D and the real ones is done in subsection 5.1.1. In subsection 5.1.2 can be seen the relation between the numerical predictions of the crack propagation length progress and the experimental findings. The results are summarized in tables and graphs, with some figures used to show critical examples.

5.1.1 Crack Propagation Direction

The crack propagation direction has been evaluated by means of the tool CT3D_Validator (see section 4.3). To compare the differences between the real and experimental results, the parameter dev_x has been measured by the program. For this specimen, the x-axis represents the longitudinal axis and it has been chosen as a reference to calculate the deviation. In Table 5.1 all the loads and temperatures combinations with the respective results are summarized. The deviation for the crack propagation direction is reported in *mm* for three different locations along the crack. The column for "Global" considers dev_x over the entire crack propagation surface. "Initial" and "final" show this value locally at the beginning of the propagation and at its end (it is taken from the numerical surface). It can be noticed that for specimen 13526, these values could not be measured since the geometry of the notch was different. For the other fifteen specimens, the initial dev_x is never equal to zero. This parameter tells if the crack started the propagation exactly at the notch tip or if it had an initial deviation. Hence, the ideal case is represented by $dev_x = 0$. Many reasons can cause deviations, such as an inaccurate notch machining with residual stresses or material flaws. In the beginning, the presence of high mode II and III loading induced for all of them a crack propagation slightly twisted from the notch tip. This can be easily seen also in Figure 5.1, were looking at the corners on the bottom of the crack surface in blu, the difference from the experiment is clear. All the simulations were run up to component failure.

The global and finial deviations depend on the crack propagation. In Table 5.1 it is interesting to look at the line for the specimens 13320 and 13414, where the load and temperature conditions are the same. The identical test has been done twice but with different results. For the initial dev_x, as previously described, this difference can be explained by the complexity of making exactly the same notch machining. Moreover, because of material differences (location and size of grains), the global and final deviations differ for the two specimens as well.

Table 5.1: Deviation of the crack propagation direction for the 4-point bending specimens analysed

Num.	Label	T [$°C$]	Peak load [kN]	Global $dev_x[mm]$	Initial $dev_x[mm]$	Final $dev_x[mm]$
1	13298	350	3.02	0.3351	0.1224	0.6098
2	13305	350	4.78	0.1694	0.1144	0.1642
3	13320	350	4.18	0.2344	0.1443	0.1784
4	13321	350	3.58	0.1708	0.1218	0.1967
5	13414	350	4.18	0.3495	0.2324	0.5196
6	13444	250	4.78	0.1831	0.1284	0.2269
7	13445	250	4.18	0.1654	0.1804	0.1750
8	13466	150	4.78	0.3618	0.2414	0.3766
9	13467	150	4.48	0.3115	0.2436	0.2827
10	13485	50	4.78	0.2102	0.1685	0.2206
11	13486	50	4.78	0.2240	0.1357	0.3653
12	13495	250	4.48	0.1725	0.1758	0.1628
13	13498	350	3.88	0.4130	0.2136	0,3339
14	13510	250	3.88	0.1985	0.1829	0.1781
15	13521	150	4.18	0.3241	0.0826	0.4896
16	13526	150	3.58	-	-	-

Especially for the final dev_x, the values are 0.1784 mm for the specimen 13320 and 0,5196 mm the specimen 13414. The same observation can be done for the specimens 13485 and 13486. Compared to the former couple, these are tested with a higher peak load at a lower temperature. However, in this case, the results are more consistent and the difference for the initial, global and final deviations is much lower.

The rest of the tests were performed just once, with different temperatures and peak loads. The vast majority was done at the highest temperature: 350$°C$. Focusing on this temperature condition, it can be observed that there is no relation with the peak load and the three deviations. This confirms that the main role is played by the material differences, not predictable by the software and not equal for each experiment.

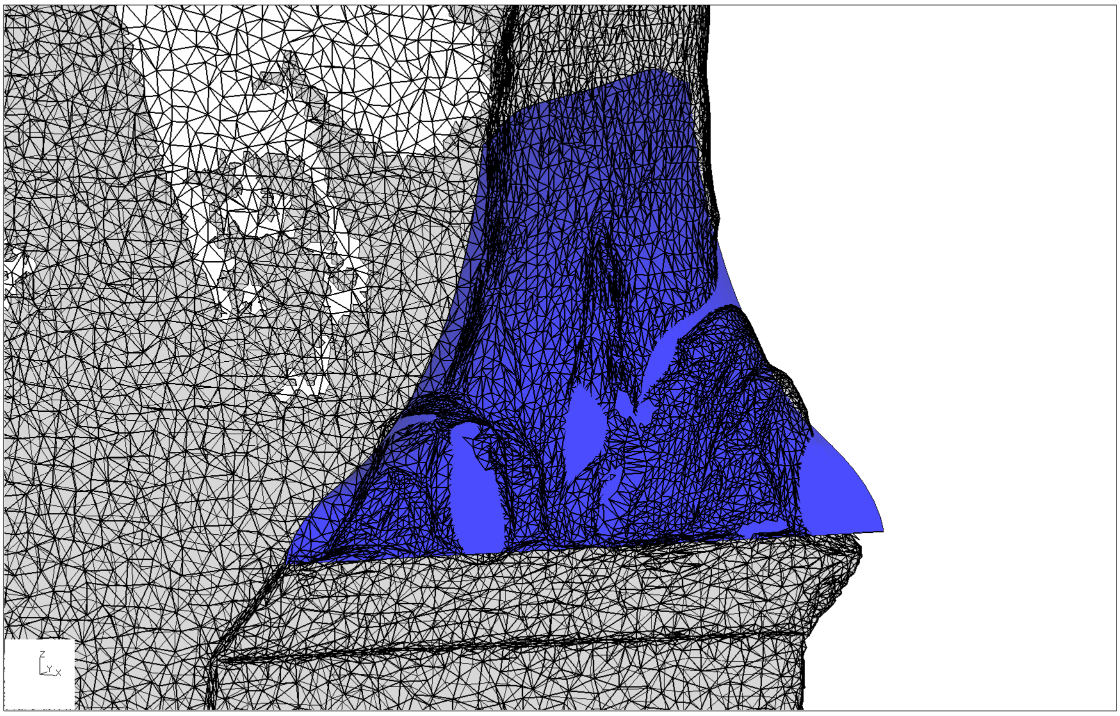

Figure 5.1: Crack surface predicted by Cracktracer3D (in blue) compared with the experimental result of specimen 13444

The graphical comparison between the experimental crack surface and the numerical one for specimen 13444 is shown in Figure 5.1. At first, it can be noticed that the result of Cracktracer3D (in blue) is a smooth surface. At the contrary, the real crack surface shows many irregularities which are called factory-roof patterns (see section 4.3). In the figure, the real specimen surface is kept slightly transparent to show the numerical crack evolution inside and outside the body. It can be seen that at the end of the propagation, the numerical surface is entirely inside the component. This is the reason of a final dev_x equal to 0.2269 mm. However, this value is relatively small and it is possible to observe that the global prediction is very accurate. The real crack has an initial twist due to loading mode II and III, until loading mode I becomes predominant and leads the crack to propagation on the plane y-z in the middle of the specimen. Despite non-idealities of the phenomenon, this crack propagation behaviour is well represented by Cracktracer3D.

To achieve a better evaluation for the accuracy of Cracktracer3D, the values for global and final dev_x in Table 5.1 can be expressed as a percentage of the crack propagation length measured on the specimen free surface. Since the crack propagates intersecting two sides of the specimen, Cracktracer3D calculates two different a_{surf}. Due to the symmetry of the problem, this value is the same for the two sides and for each specimen. It does not make sense to relate the initial deviation to the crack propagation length since it is independent on the crack behaviour. Thus, in Table 5.2 the a_{surf} for the specimens is reported and its relative percentage of the global and final deviation, making their comparison simpler.

Looking at the global behaviour, the crack propagation direction deviates in the worst case of 6.86 % by a_{surf}. The best prediction is represented by the specimen 13321 with 2.59%.

Across all fifteen specimens, the global dev_x has an average of 4.45 % and the final dev_x of 5.09 %.

Table 5.2: Deviation of the crack propagation direction in percentage of a_{surf} for the analysed 4-point bending specimens

Label	a_{surf} $[mm]$	Global $dev_x[\%]$	Final $dev_x[\%]$
13298	7.1677	4.68	8.51
13305	5.2190	3.25	3.15
13320	5.9545	3.94	3.00
13321	6.6058	2.59	2.98
13414	5.9545	5.87	8.73
13444	5.2546	3.48	4.32
13445	5.9030	2.80	2.96
13466	5.2762	6.86	7.14
13467	5.5930	5.57	5.06
13485	5.2782	3.98	4.18
13486	5.2782	4.24	6.92
13495	5.5881	3.09	2.19
13498	6.1744	6.69	5.41
13510	6.2627	3.17	2.84
13521	5.9308	6.48	8.25

5.1.2 Crack Propagation Length and number of loading cycles

The second comparison for the 4-point bending specimens is done to study the crack propagation length over the number of loading cycles and the prediction of the failure. In Table 5.3 a number of loading cycles to failure are shown from the experiments and numerical simulations. For each specimen, a ratio can be defined between these two results as shown in Equation 5.1.

$$N_{ratio} = \frac{N_{failure,experimental}}{N_{failure,Cracktracer3D}} \tag{5.1}$$

If N_{ratio} is greater than 1, it means that Cracktracer3D is more conservative. On the contrary, if N_{ratio} is lower than 1, Cracktracer3D predicts a higher number of loading cycles to failure than the real one. Intuitively, for $N_{ratio} = 1$, the prediction is perfect. Looking at the values reported in the table, N_{ratio} is always greater than 1 except for specimen 13305, where $N_{ratio} = 0.9773$. The highest deviation among the experimental and numerical results is given by specimen 13467, with a N_{ratio} equal to 3.4303. In average, counting all the fifteen specimens, N_{ratio} is equal to 2.1514.

The last column of Table 5.3 shows the parameter FR for each specimen which is a measure for the factory-roof patterns described in section 4.3. The value is given in *mm* and is measured taking as reference the longitudinal axis x. The bigger FR, the larger the factory-roof effect. In Figure 5.2 the relation between N_{ratio} and FR can be observed. A trend curve based on a logarithmic function such as $a\log(x) + b$ is represented with a dashed blue line. From the experiment, it can be noticed that when the factory-roof effect is very strong, then the N_{ratio} increases. Hence, the presence of the factory-roof delays the crack propagation progress. This can be explained by considering the factory-roof patterns as elements which create friction between the two crack surfaces. For all of the specimens, this effect occurred mainly in the first half of the propagation, where most of loading cycles took place. Since it is a component-dependent phenomenon it is not stimulated by Cracktracer3D and this causes the biggest difference of the results.

A similar problem with a different initial crack geometry has been analysed in (Kikuchi, Wada, & Suga, 2011). The contact between the surfaces reduces the crack growth rate and also the stress intensity factor range. Considering the crack propagation criterion proposed in (Richard, 2003), the presence of the factory-roof effect can be taken into account by reducing ΔK_{eq} when K_{III} is large. However, this method has been proved just for this specific type of specimen.

In subsection 4.1.1 has been already explained how the load sequence was changed during the test in order to apply the marker load technique. With these marks on the experimental crack surface, it is possible to evaluate the crack length evolution. However, for most of the specimens, those marks were not easy to detect with the microscope. Therefore, the analysis of the marks has been made just on six specimens which showed clear and measurable results. More specifically, the maximum crack propagation length has been measured from the bottom of the specimen for each front mark. The results are shown in Figure 5.3 and compared with the prediction of a_{max} made by Cracktracer3D. Each specimen is identified by a specific colour: a continuous line represents the numerical result whereas a dashed line the experimental one. The general curve trend is well replicated by Cractracer3D. However, it can be noticed that for the same specimen, the two curves differ from each other by a horizontal translation. This

Table 5.3: Difference in number of loading cycles and FR for the 4-point bending specimens analysed

Label	T [$°C$]	Peak load [kN]	N failure (experimental)	N failure (Cracktracer3D)	N_{ratio}	FR [mm]
13298	350	3.02	40110	35111	1.1424	0.0599
13305	350	4.78	8940	9148	0.9773	0.0628
13320	350	4.18	16806	13706	1.2262	0.0635
13321	350	3.58	52880	21538	2.4552	0.0561
13414	350	4.18	32700	13706	2.3858	0.0800
13444	250	4.78	20820	9676	2.1517	0.0852
13445	250	4.18	42456	14987	2.8329	0.0595
13466	150	4.78	30748	10472	2.9362	0.0823
13467	150	4.48	45256	13193	3.4303	0.1548
13485	50	4.78	23100	11673	1.9789	0.0820
13486	50	4.78	18922	11673	1.6210	0.0738
13495	250	4.48	31030	11993	2.5873	0.0972
13498	350	3.88	50270	17052	2.9480	0.1811
13510	250	3.88	33237	18989	1.7503	0.0523
13521	150	4.18	31026	16791	1.8478	0.0671
13526	150	3.58	32034	-	-	-

offset is given by N_{ratio}.

The worst prediction is done for specimen 13467, in accordance with Table 5.3. The FR value, in this case, is very high and the presence of protruding factory-roof patterns is visible looking at the experimental crack surfaces in Figure 5.4. For this case, it is also interesting to observe that the two sides of the crack surface are not complementary. The most protruding discontinuity, visible on the right side, cannot be placed complementary on the other component side. The reason is the presence of contact between the crack surfaces during the propagation, which in this case has really bent the pattern toward the outside. For this

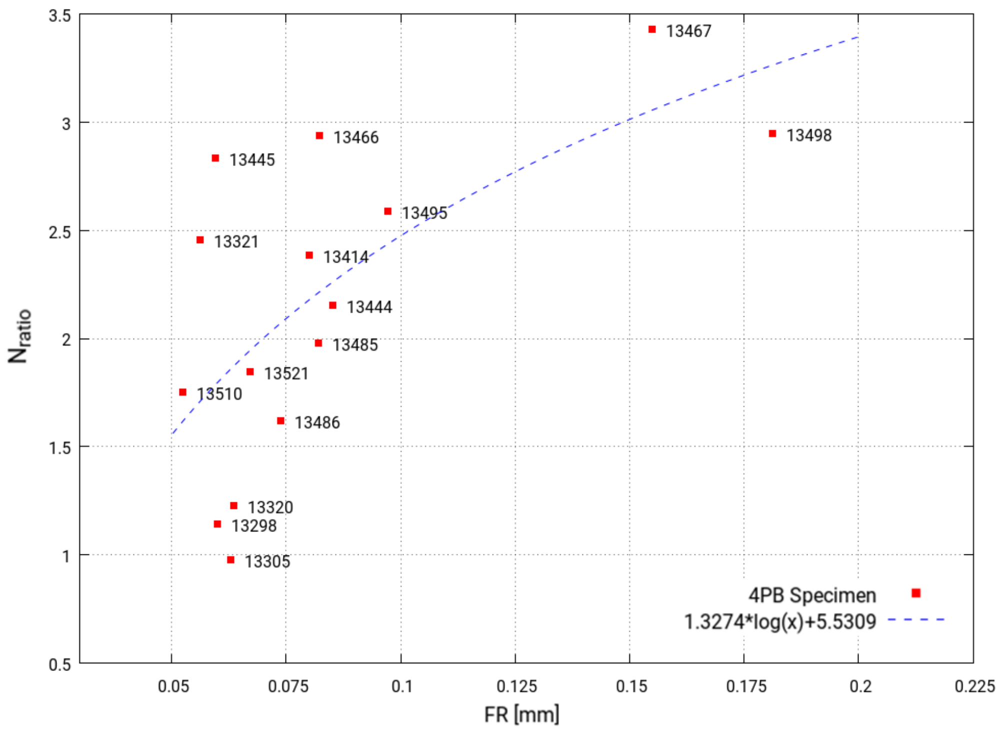

Figure 5.2: N_{ratio} dependency on FR

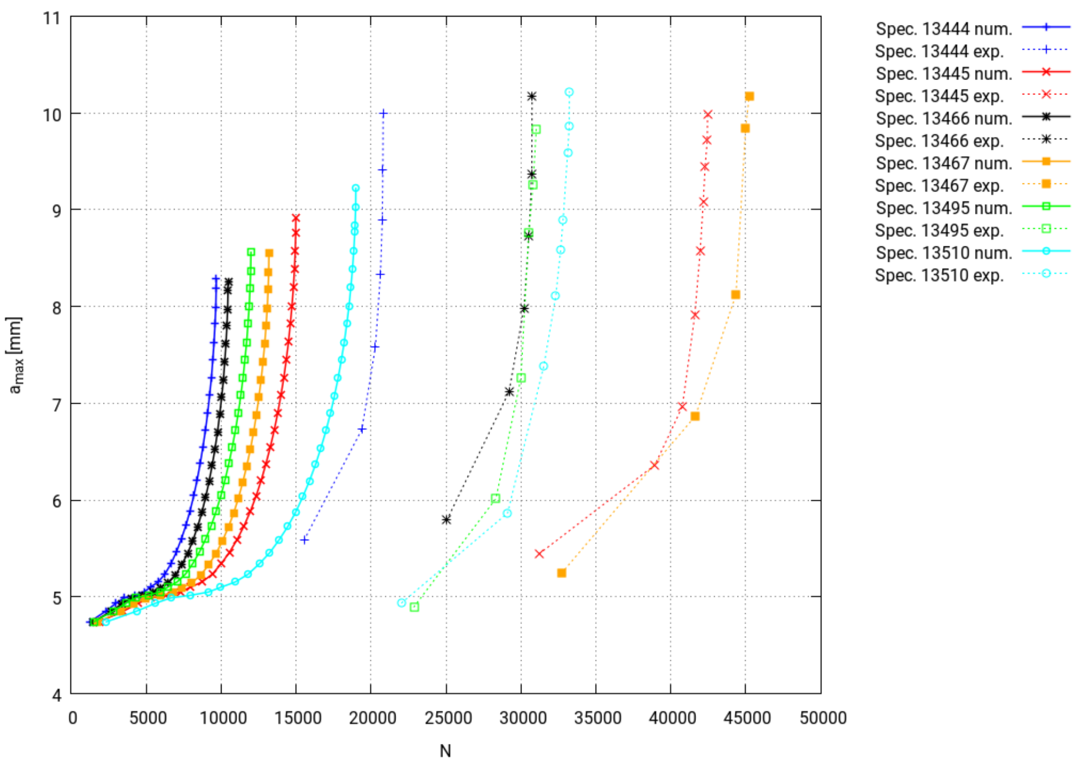

Figure 5.3: Maximum crack propagation length over the number of loading cycles. The continuous curves are the numerical results. The dashed curves are experimental ones

specimen, to measure all the three deviations and FR, the crack side has been chosen with the most precise triangulation.

Figure 5.4: Experimental crack surfaces of specimen 13467 with a strong factory roof effect

5.2 Tension-Torsion Specimen

In this section, the results for the thirtysix tension-torsion specimens tested are analysed. In subsection 5.2.1 a comparison can be found between the crack propagation directions predicted by Cracktracer3D and the real ones, whereas in subsection 5.2.2 a comparison is reported between the two crack propagation criteria shown in section 4.2. The results are summarized in tables and figures.

5.2.1 Crack Propagation Direction

To evaluate the prediction of the crack propagation direction made by Cractracer3D, the same approach has been used, as previously done in subsection 5.1.1. Then, for these specimens, the three dev_z (global, initial and final) are measured with CT3D_Validator. The current type of specimen has the z-axis as longitudinal axis. In Table 5.4 just the global and final deviations are reported since the initial one is approximately equal to 0 for all the specimens. This means that the crack propagation starts exactly at the notch tip, without any pre-twist or other imperfections. For these specimens, there are eighteen possible combinations of loads. The third and fourth column contains the value for the axial stress ratio R_{axial}, and the torsional one $R_{torsion}$, which is always kept equal to -1. The phase shift is shown in degrees whereas the load amplitudes are reported non-dimensionalized in terms of $\frac{\tau_N}{\sigma_N}$ (see subsection 4.1.2). The global and final dev_z, given in mm, are measured taking as reference the crack surface computed by Cractracer3D. The main interest for these specimens is to study the crack propagation near the notch tip, where it can be easily detected from the experiments. Therefore, for each specimen, the iterative process of Cracktracer3D has been stopped trying to keep the crack evolution in a clear crack propagation region without critical or non-ideal phenomena.

It can be noticed that in Table 5.4 the specimens 0_1_90_1 and 1_1_90_1 do not have any value for the deviations. In case of 90 degrees of phase, Cracktracer3D showed numerical instabilities, probably due to the irregular principal planes along the crack tip. However, for $\frac{\tau_N}{\sigma_N} = 1$ and $\frac{\tau_N}{\sigma_N} = 0.76$, it has been possible to run the simulations longer because of fewer irregularities, obtaining tolerable results to compare. The two critical specimens have been tested three different times and the results can be observed in Figure 5.5 and Figure 5.6. The experimental results show diverse crack surfaces with the same loading conditions. Therefore, it can be confirmed that there is also instability in reality and these two specimens represent a critical case. When the torsional moment magnitude is reduced, the crack propagation surface is easier to detect. However, small irregularities occurred in these cases as well, with continuous changes of the crack propagation direction, leaving an arbitrary choice of the crack propagation surface.

For each specimen, the value of the final dev_z is always greater than the global dev_z. This means that the crack propagation direction deviation measured at the end of the numerical crack front is less precise than the global average. Therefore, the more the numerical crack evolves, the bigger is the deviation. To compare correctly the specimens, it is necessary to normalize dev_z with the crack propagation length measured on the component surface. In Table 5.5 the computed value of a_{surf} is reported in mm and the global and final dev_z are reported as percentage of a_{surf}. Focusing on the column for global dev_z, it can be observed that the maximum value of deviation is given by the specimen 0_1_40_3, with global $dev_z = 17.26\%$.

The most precise prediction gives a global dev_z equal to 1.71% for the specimen 1_1_00_3. Considering all the measurements, the average value for global dev_z is 7.65%. Following the same logic, the best prediction for the final dev_z is: 2.75% for the same specimen as for the global one. The worst case, with 33.08% is given by the specimen 0_1_90_2. The average final deviation is equal to 15.20%. In case of traction without compression, for the in-phase and 40° out-of-phase cases, it is evident that the numerical predictions are globally more precise when the torsional moment magnitude increases. This relation is not present when compression occurs. It can be confirmed that the crack character strictly depends on the phase shift angle.

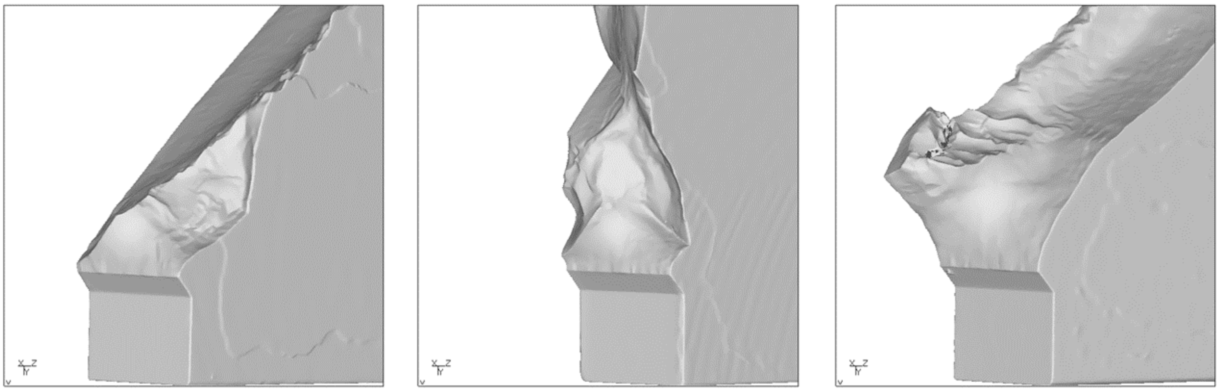

Figure 5.5: Experimental crack surface of three different specimens 0_1_90_1

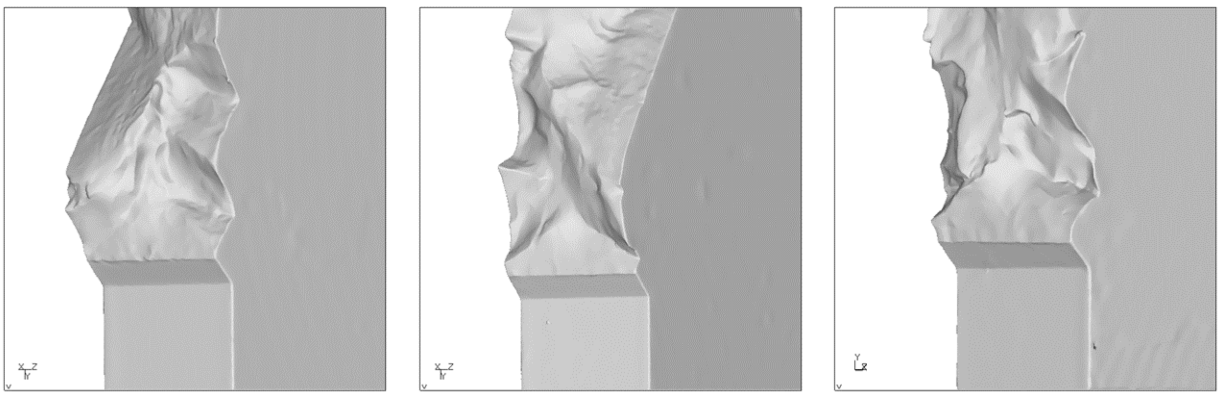

Figure 5.6: Experimental crack surface of three different specimens 1_1_90_1

Table 5.4: Deviation of the crack propagation direction for the tension-torsion specimens analysed

Num.	Label	R_{axial}	$R_{torsion}$	Phase [°]	$\frac{\tau_N}{\sigma_N}$	Global $dev_z[mm]$	Final $dev_z[mm]$
1	0_1_00_1	0	-1	0	1.31	0.0617	0.1482
2	0_1_00_2	0	-1	0	1	0.0657	0.1446
3	0_1_00_3	0	-1	0	0.76	0.0705	0.1353
4	0_1_40_1	0	-1	40	1.31	0.0979	0.2669
5	0_1_40_2	0	-1	40	1	0.1237	0.2648
6	0_1_40_3	0	-1	40	0.76	0.1726	0.2960
7	0_1_90_1	0	-1	90	1.31	-	-
8	0_1_90_2	0	-1	90	1	0.1415	0.3308
9	0_1_90_3	0	-1	90	0.76	0.1170	0.1704
10	1_1_00_1	-1	-1	0	1.31	0.0196	0.0384
11	1_1_00_2	-1	-1	0	1	0.0455	0.0921
12	1_1_00_3	-1	-1	0	0.76	0.0171	0.0275
13	1_1_40_1	-1	-1	40	1.31	0.0791	0.1846
14	1_1_40_2	-1	-1	40	1	0.0336	0.0901
15	1_1_40_3	-1	-1	40	0.76	0.0178	0.0439
16	1_1_90_1	-1	-1	90	1.31	-	-
17	1_1_90_2	-1	-1	90	1	0.1206	0.1353
18	1_1_90_3	-1	-1	90	0.76	0.0411	0.0636

Table 5.5: Deviation of the crack propagation direction in percentage of a_{surf} for the tension-torsion specimens analysed

Num.	Label	a_{surf} $[mm]$	Global $dev_z[\%]$	Final $dev_z[\%]$
1	0_1_00_1	4.6695	6.17	14.82
2	0_1_00_2	5.0724	6.57	14.46
3	0_1_00_3	5.8457	7.05	13.53
4	0_1_40_1	3.9610	9.79	26.69
5	0_1_40_2	3.9556	12.37	26.48
6	0_1_40_3	3.0564	17.26	29.60
8	0_1_90_2	0.5843	14.15	33.08
9	0_1_90_3	0.7967	11.70	17.04
10	1_1_00_1	4.9594	1.96	3.84
11	1_1_00_2	5.0346	4.55	9.21
12	1_1_00_3	5.8481	1.71	2.75
13	1_1_40_1	3.9614	7.91	18.46
14	1_1_40_2	3.9414	3.36	9.01
15	1_1_40_3	3.8992	1.78	4.39
17	1_1_90_2	0.4978	12.06	13.53
18	1_1_90_3	3.7642	4.11	6.36

5.2.2 New and Current Approach

In this section, the previously described results are compared with the new crack propagation approach proposed in subsection 4.2.2. To accomplish the right comparison, the numerical analyses were run trying to keep the same a_{surf}. In this way, the new approach can be directly compared with the current one by looking at the percentage values of global and final dev_z. In Table 5.6 these values are reported for all the specimens. To confront the two approaches, global and final dev_z has to be analysed separately. For $R_{axial} = 0$, the current approach is always more precise for 0 and 40 degrees of phase. Accordingly, even with the new method, the crack propagation direction is more precise increasing the torsional moment magnitude. The two results for the specimen 0_1_00_1 are shown in Figure 5.7. In blue is represented the crack surface predicted with the new method, whereas the result with the current one is represented in red. It is clear that the red surface is closer to the real one. Although the blu one has a higher deviation, the crack kinking angle does not differ too much. When the phase is equal to 90°, then for the specimens 0_1_90_2 and 0_1_90_3, the new method generates a better prediction. In Figure 5.8 the graphical results for the specimen 0_1_90_3 can be observed. The irregularity of the real crack surface is obvious. The two numerical results do not differ to each other so much, except for the smoothness of the surfaces. In fact, the current method shows an oscillating behaviour along the propagation whereas the new one propagates straightly. The current method calculates the crack propagation direction for the dominant step in each iteration (see subsection 4.2.1). Due to the phase of 90°, the dominant step changes for each iteration as represented in Figure 5.9. This explains the numerical behaviour and somehow, it replicates the irregular changes of the real crack propagation direction. The new method considers the crack inclination angle as an average for all the loading steps and therefore, its propagation results are smoother. The same can be noticed for the other two cases with 90° of out-of-phase loadings.

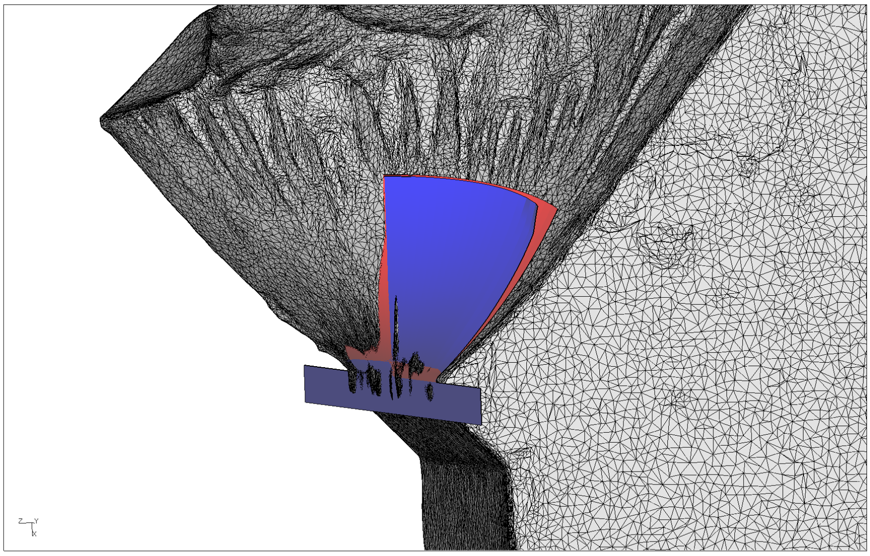

Figure 5.7: Experimental crack surface of the specimen 0_1_00_1 *with the comparison between the new approach (in blue) and the current one (in red)*

In the case of traction-compression loads ($R_{axial} = -1$), the difference between the two methods is not very significant. For the specimen, 0_1_90_3, the global dev_z is the same for both

Table 5.6: Deviation of the crack propagation direction in percentage of a_{surf} for the tension-torsion specimens analysed with the current and new crack propagation approach

			Global		Final	
Num.	Label	a_{surf} $[mm]$	$dev_z[\%]$ Current	$dev_z[\%]$ New	$dev_z[\%]$ Current	$dev_z[\%]$ New
1	0_1_00_1	4.6695	6.17	7.84	14.82	18.70
2	0_1_00_2	5.0724	6.57	8.14	14.46	18.04
3	0_1_00_3	5.8457	7.05	8.42	13.53	16.62
4	0_1_40_1	3.9610	9.79	12.57	26.69	31.35
5	0_1_40_2	3.9556	12.37	15.63	26.48	33.53
6	0_1_40_3	3.0564	17.26	20.32	29.60	36.11
8	0_1_90_2	0.5843	14.15	11.87	33.08	6.58
9	0_1_90_3	0.7967	11.70	11.40	17.04	16.04
10	1_1_00_1	4.9594	1.96	1.96	3.84	3.84
11	1_1_00_2	5.0346	4.55	4.58	9.21	9.35
12	1_1_00_3	5.8481	1.71	1.73	2.75	2.82
13	1_1_40_1	3.9614	7.91	8.73	18.46	19.94
14	1_1_40_2	3.9414	3.36	4.39	9.01	10.08
15	1_1_40_3	3.8992	1.78	1.48	4.39	2.51
17	1_1_90_2	0.4978	12.06	11.60	13.53	11.57
18	1_1_90_3	3.7642	4.11	2.94	6.36	4.17

criteria. In Figure 5.10 the graphical comparison between them is shown. A minimal difference occurs at the beginning of the crack propagation. During the entire crack progress, the two surfaces match perfectly and replicate precisely the real crack surface. Focusing on the final dev_z, the same considerations can be done.

For the new method, the best global prediction is given for specimen 1_1_40_3, where the global dev_z is equal to 1.48%. The worst result is given by the same specimen as the one for

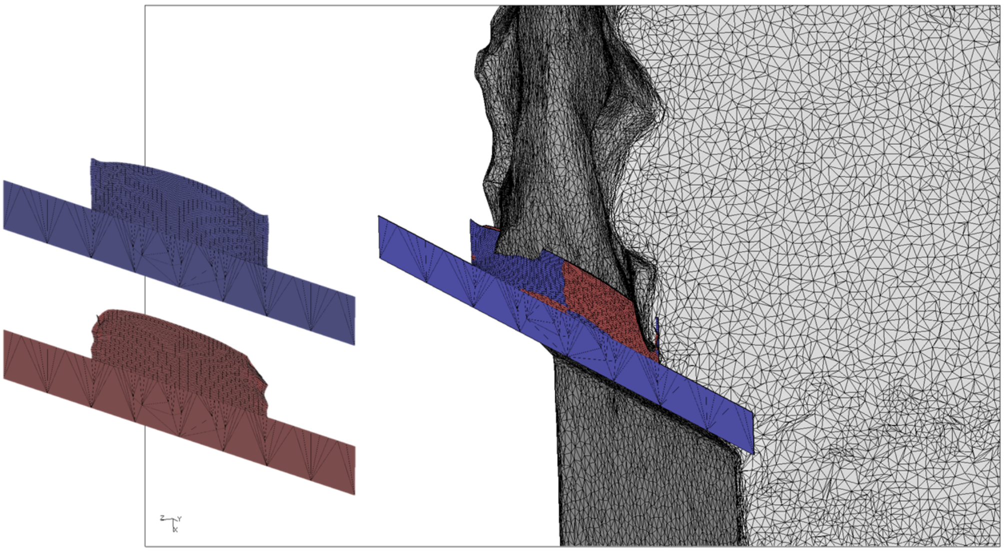

Figure 5.8: Experimental crack surface of the specimen 0_1_90_3 *with the comparison between the new approach (in blue) and the current one (in red)*

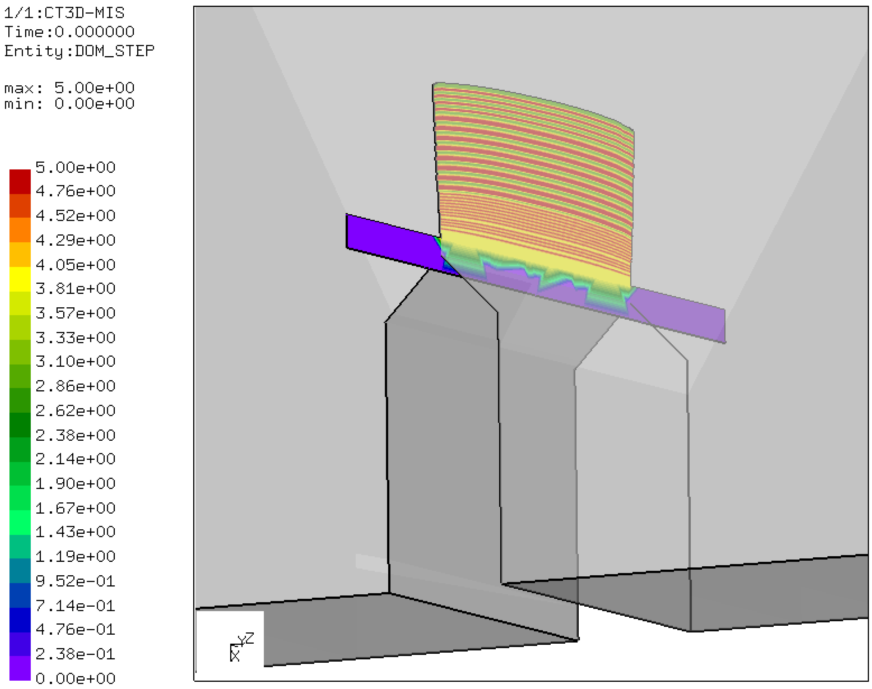

Figure 5.9: Dominant loading step for the specimen 1_1_90_3

the current method: 0_1_40_3 with global $dev_z = 20.32\%$. In average, the global deviation for the new criterion is 8.35%. For the final deviation, the same specimens as for the global deviation exhibit the best and worst prediction, with respectively $dev_z = 2.51\%$ and $dev_z = 36.11\%$. The average value is 15.08%.

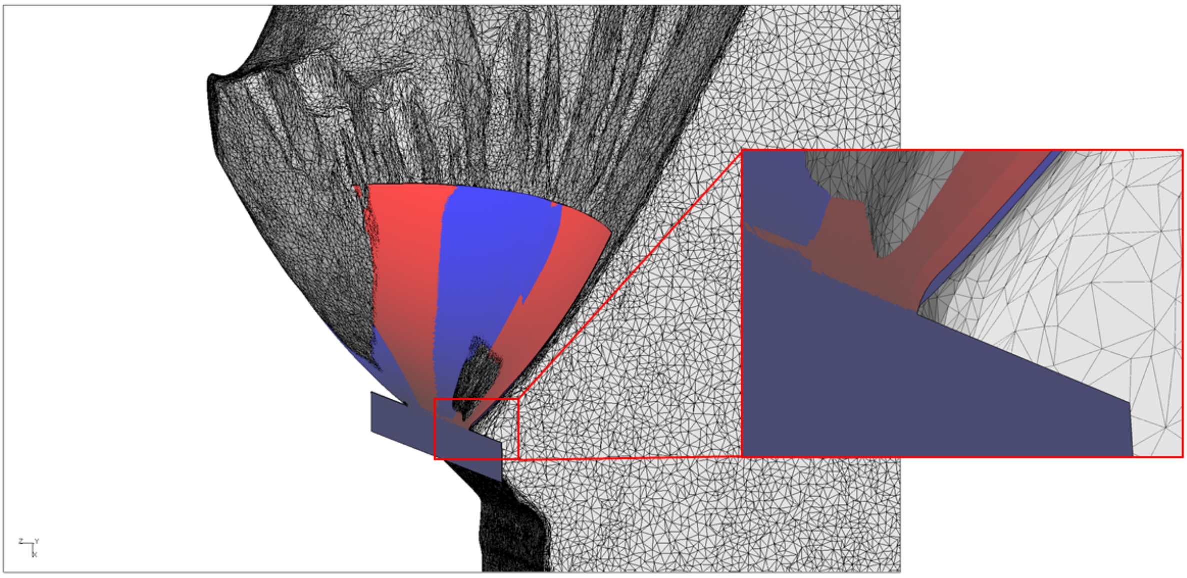

Figure 5.10: Experimental crack surface of the specimen 1_1_00_1 *with the comparison between the new approach (in blue) and the current one (in red)*

6 Discussion and Conclusion

Through experimental results, the validation for the software Cracktracer3D has been done in case of mixed-mode crack propagation. Two different types of specimen have been investigated:

- the 4-point bending specimen, to compare the crack propagation direction and the crack length evolution over the number of loading cycles.
- the tension-torsion specimen, to analyse the crack propagation prediction in case of a loading mission, with in-phase and out-of-phase loading conditions.

Based on the results for the second type of specimen, a new crack propagation criterion has been tested for mission analysis.

Focusing on the 4-point bending specimen results, it can be confirmed that Cracktracer3D replicates the same crack propagation behaviour as in reality. The deviation for the crack propagation direction is relatively small compared to the crack propagation length measured on the component free surface. The difference in percentage stays under 7% for all the fifteen tests analysed.

Studying the crack propagation evolution, Cracktracer3D showed faster propagation, remaining on the conservative sides. N_{ratio}, defined as the ratio of the real number of loading cycles to failure and the predicted one, yielded an average value of 2.1514. This means that on average, Cracktracer3D has approximately a conservative margin of 2 with respect to reality. The reason behind these results has been investigated studying the influence of the factory-roof patterns. In the experimental results, the presence of these irregularities creates friction between the crack surfaces and delays the crack propagation. Thus, a relation between N_{ratio} and FR has been established. Unfortunately, this phenomenon is not replicable in the software since discontinuous geometries cannot be modeled. However, with more experiments to take into account the factory-roof phenomenon, a statistical approach which identify a corrective factor for the propagation can be implemented in the future.

Concerning the tension-torsion specimen, another positive evaluation of Cractracer3D can be stated. Despite the challenge of complicated loading missions, the crack propagation direction has always been predicted with a deviation below 18% of the crack propagation length measured on the component side. In more than 50% of the cases, this deviation was lower than the 10%. In the case of 90 degrees of phase, the software shows some numerical instabilities, whereas at the same time the experimental results are not very reproducible.

Finally, a new crack propagation approach has been compared with the current one, already implemented in Cracktracer3D. The results showed a lower accuracy for most of the cases analysed by the new method. This fact can be used to prove the validity of the approach currently used by the software. The new method gave a more accurate prediction for the critical cases with 90 degrees of phase, due to the smoothing within the loading steps. However, for these specific examples, the experimental results left some uncertainties in detecting the real crack surfaces to compare. Either way, further experimental investigations with different loading missions should be studied to confirm the validation.

7 References

Anderson, T. L. (2005). *Fracture mechanics: Fundamentals and applications, third edition*. Taylor & Francis.

Barlow, K., & Chandra, R. (2005). Fatigue crack propagation simulation in an aircraft engine fan blade attachment. *International Journal of Fatigue*, *27*, 1661–1668.

Beasy. (2020). https://www.beasy.com/. (Accessed on 2020-09-06)

Bhattacharya, S., Singh, I. V., & Mishra, B. (2013, 10). Fatigue-life estimation of functionally graded materials using xfem. *Engineering with Computers*, *29*. doi: 10.1007/s00366-012-0261-2

Caputo, F., Lamanna, G., Lanzillo, L., & Soprano, A. (2013, 09). Numerical investigation on lefm limits under lsy conditions. *Key Engineering Materials*, *577-578*, 381-384. doi: 10.4028/www.scientific.net/KEM.577-578.381

Chambel, P., Martins, R. F., & Reis, L. (2016). Research on fatigue crack propagation in ct specimens subjected to loading modes i, ii or iii. *Procedia Structural Integrity*, *1*, 134–141.

Chernyatin, A., Matvienko, Y., & Razumovsky, I. (2018, 04). Fatigue surface crack propagation and intersecting cracks in connection with welding residual stresses. *Fatigue & Fracture of Engineering Materials & Structures*, *41*. doi: 10.1111/ffe.12808

Dhondt, G. (1993). General behaviour of collapsed 8-node 2-d and 20-node 3-d isoparametric elements. *International Journal for Numerical Methods in Engineering*, *36*(7), 1223-1243. doi: 10.1002/nme.1620360708

Dhondt, G. (2001, 05). 3-d mixed-mode k-calculations with the interaction integral method and the quarter point element stress method. *Communications in Numerical Methods in Engineering*, *17*, 303 - 307. doi: 10.1002/cnm.407

Dhondt, G. (2002, 05). Mixed-mode k-calculations in anisotropic materials. *Engineering Fracture Mechanics - ENG FRACTURE MECH*, *69*, 909-922. doi: 10.1016/S0013-7944(01)00127-8

Dhondt, G. (2005). Mixed-mode crack propagation calculations in a pure hexahedral mesh. *Structural Durability & Health Monitoring*, *1*(1), 21–34. doi: 10.3970/sdhm.2005.001.021

Dhondt, G. (2011, 09). Cyclic mixed-mode crack propagation due to time-dependent multiaxial loading in jet engines. *Key Engineering Materials*, *488-489*. doi: 10.4028/www.scientific.net/KEM.488-489.93

Dhondt, G. (2014). Application of the finite element method to mixed-mode cyclic crack propagation calculations in specimens. *International Journal of Fatigue*, *58*, 2–11.

Dhondt, G. (2020). *Calculix crunchix user's manual - version 2.17*.

Dhondt, G., & Mångård, D. (2009, 10). Automatic cyclic crack propagation calculations in aircraft engines. *Key Engineering Materials - KEY ENG MAT*, *417-418*, 733-736. (Mångård, formally Bremberg) doi: 10.4028/www.scientific.net/KEM.417-418.733

Dhondt, G., Rupp, M., & Hackenberg, H.-P. (2015, 07). A modified cyclic crack propagation description. *Engineering Fracture Mechanics*, *146*. doi: 10.1016/j.engfracmech.2015.07.020

Dhondt, G., & Wittig, K. (2020, Jul). *Calculix*. `http://www.calculix.de`. (Accessed on 2020-08-02)

Dorca, B.-P. (2018). *Three-dimensional mixed-mode crack propagation calculations based* *on a submodel technique* (Unpublished master's thesis). TU-Delft, the Netherlands.

Downing, S., & Socie, D. (1982). Simple rainflow counting algorithms. *International Journal of Fatigue*, *4*(1), 31 - 40. Retrieved from `http://www.sciencedirect.com/science/article/pii/0142112382900184` doi: https://doi.org/10.1016/0142-1123(82)90018-4

Findlay, S. J., & Harrison, N. D. (2002). Why aircraft fail. *Materials Today*, *5*(11), 18–25.

Gdoutos, E. (2020). *Fracture mechanics: An introduction*. Springer.

Haefele, P. M., & Lee, J. D. (1995). Combination of finite element analysis and analytical crack tip solution for mixed mode fracture. *Engineering Fracture Mechanics*, *50*(5), 849 - 868. Retrieved from `http://www.sciencedirect.com/science/article/pii/0013794494E0063M` doi: https://doi.org/10.1016/0013-7944(94)E0063-M

Hannemann, R., Köster, P., & Sander, M. (2017, 12). Investigations on crack propagation in wheelset axles under rotating bending and mixed mode loading. *Procedia Structural Integrity*, *5*, 861-868. doi: 10.1016/j.prostr.2017.07.104

Hardrath, F. H. (1971). Fatigue and fracture mechanics. *Journal of Aircraft*, *8*(3), 129–142.

Ingraffea, A., Wawrzynek, P., Carter, B. J., & Ibrahim, O. (2020, July). *Franc3d.* https://franc3d.in/. (Accessed on 2020-09-06)

Kikuchi, M., Wada, Y., & Suga, K. (2011, 12). Surface crack growth simulation under mixed mode cyclic loading condition. *Procedia Engineering*, *10*, 427-432. doi: 10.1016/j.proeng.2011.04.073

Kim, J.-S., Lee, H.-J., Kim, Y.-J., & Kim, Y.-B. (2019). The mesh density effect on stress intensity factor calculation using abaqus xfem. *Journal of Mechanical Science and Technology*, *33*, 4909–4916.

Köster, P., Benz, C., Heyer, H., & Sander, M. (2020, 04). In-phase and out-of-phase mixed mode loading: Investigation of fatigue crack growth in sen specimen due to tension–compression and torsional loading. *Theoretical and Applied Fracture Mechanics*, *108*, 102586. doi: 10.1016/j.tafmec.2020.102586

Lazzeri, R. (2002). A comparison between safe life, damage tolerance, and probabilistic approaches to aircraft structure fatigue design. *Aerotecnica Missili & Spazio*, *81*(2), 53–64.

Liu, A. F. (2005). *Mechanics and mechanisms of fracture: An introduction*. ASM International.

Liu, W., Yang, R., Mu, Z., Liu, S., & Yu, D. (2011, 08). Comparison of crack growth models and crack closure effect. *Advanced Materials Research*, *311-313*, 822-825. doi: 10.4028/www.scientific.net/AMR.311-313.822

Malíková, L., Veselý, V., & Seitl, S. (2016, 08). Crack propagation direction in a mixed mode geometry estimated via multi-parameter fracture criteria. In (p. 99-107). doi: 10.1016/j.ijfatigue.2016.01.010

Mcevil, A., & Sotomi, I. (2002). On the development of crack closure at high r levels. *Fatigue & Fracture of Engineering Materials & Structures*, *25*(11).

Meher-Homji, B., & Gabriles, G. (1998). Gas turbine blade failures-causes, avoidance, and troubleshooting. In *Turbomachinery symposium* (pp. 129–180).

Mångård, D., & Dhondt, G. (2007, 01). Automatic mixed-mode crack propagation based on a combined hexahedral-tetrahedral approach. *Key Engineering Materials*, *348-349*. (Mångård, formally Bremberg) doi: 10.4028/www.scientific.net/KEM.348-349.581

Mångård, D., & Dhondt, G. (2009, 05). Automatic 3-d crack propagation calculations: A pure hexahedral element approach versus a combined element approach. *International*

Journal of Fracture, *157*, 109-118. (Mångård, formally Bremberg) doi: 10.1007/s10704-009-9313-z

Netgen. (2019). https://ngsolve.org/. (Accessed on 2020-09-08)

Paris, C. P. (2014). A brief history of the crack tip stress intensity factor and its application. *Meccanica*, *49*, 759–764.

Perez, N. (2017). *Fracture mechanics*. Springer.

Peters, J., & Ritchie, R. (2000). Infuence of foreign-object damage on crack initiation and early crack growth during high-cycle fatigue of ti-6al-4v. *Engineering Fracture Mechanics*, *67*(3), 193–207.

Pokluda, J., Slámečka, K., & Šandera, P. (2010, 07). Mechanism of factory-roof formation. *Engineering Fracture Mechanics - ENG FRACTURE MECH*, *77*, 1763-1771. doi: 10.1016/j.engfracmech.2010.03.031

Rege, K., & Lemu, H. (2017, 12). A review of fatigue crack propagation modelling techniques using fem and xfem. *IOP Conference Series Materials Science and Engineering*, *276*, 012027. doi: 10.1088/1757-899X/276/1/012027

Richard, H. (2003, 09). Simulation of fatigue crack growth in real structures. *steel research international*, *74*. doi: 10.1002/srin.200300236

Riddell, W., Ingraffea, A., & Wawrzynek, P. (1997). Experimental observations and numerical predictions of three-dimensional fatigue crack propagation. *Engineering Fracture Mechanics*, *58*(4), 293–310.

Savruk, M., & Kazberuk, A. (2009). Problems of fracture mechanics of solid bodies with v-shaped notches. *Materials Science*, *45*(2), 162–180.

Schöllmann, M., Fulland, M., & Richard, H. (2003, 01). Development of a new software for adaptive crack growth simulations in 3d structures. *Engineering Fracture Mechanics*, *70*, 249-268. doi: 10.1016/S0013-7944(02)00028-0

Schrade, M. (2011). *Automatic mixed-mode crack propagation calculations with the finite element method* (Unpublished master's thesis). University of Stuttgart, Germany.

Sujata, M., & Bhaumik, S. (2015). Fatigue fracture of a compressor blade of an aeroengine: What caused this failure? *J Fail. Anal. and Preven.*, *15*, 457–463.

Tavares, S. M. O., & De Castro, P. M. S. T. (2017). An overview of fatigue in aircraft structures. *Fatigue & Fracture of Engineering Materials & Structures*, *40*(10), 1510–1529.

Toribio, J., Matos Franco, J. C., González, B., & Escuadra, J. (2015, 11). Influence of residual stress field on the fatigue crack propagation in prestressing steel wires. *Materials*, *8*, 7589-7597. doi: 10.3390/ma8115400

Wang, J., Zhang, X.-Q., Wei, W., Tong, J.-Y., Chen, B., Fang, G.-W., & Yin, Y.-D. (2018, 07). Investigation of fatigue growth behavior of an inclined crack in aluminum alloy plate. *Journal of Failure Analysis and Prevention*, *18*. doi: 10.1007/s11668-018-0503-8

Wittig, K. (2020). *Calculix user's manual - calculix graphix - version 2.17.1.*

Wolf, B., Revankar, S., & Riznic, J. (2009, 01). Crack growth model for the probabilistic assessment of inspection strategies for steam generator tubes. In (Vol. 1). doi: 10.1115/ICONE17-75618

Zencrack. (2018). `http://www.zentech.co.uk/zencrack.htm`. (Accessed on 2020-09-06)

Zhang, Z., Yang, G., & Hu, K. (2018). Prediction of fatigue crack growth in gas turbine engine blades using acoustic emission. *Sensors*, *18*(5), 1321.

8 List of Figures

9 List of Tables

Appendix A - CT3D_Validator

A1 Tool Organigram and Figures

This Appendix goes more into details of the tool CT3D_Validator. At first, in Figure A1.1, the organigram of CT3D_Validator is shown. Its main code is reported in the next chapter. The working principle of the subroutine near3d.f combined with the code algorithm is represented in Figure A1.2. Graphical results of the measurements of the volume between the experimental crack surface of a 4-point bending specimen and the one predicted by Cracktracer3D can be observed in Figure A1.3. Moreover, its so-called experimental crack average surface used to calculate the parameter FR due to the presence of factory roof effect can be seen at the and of this chapter.

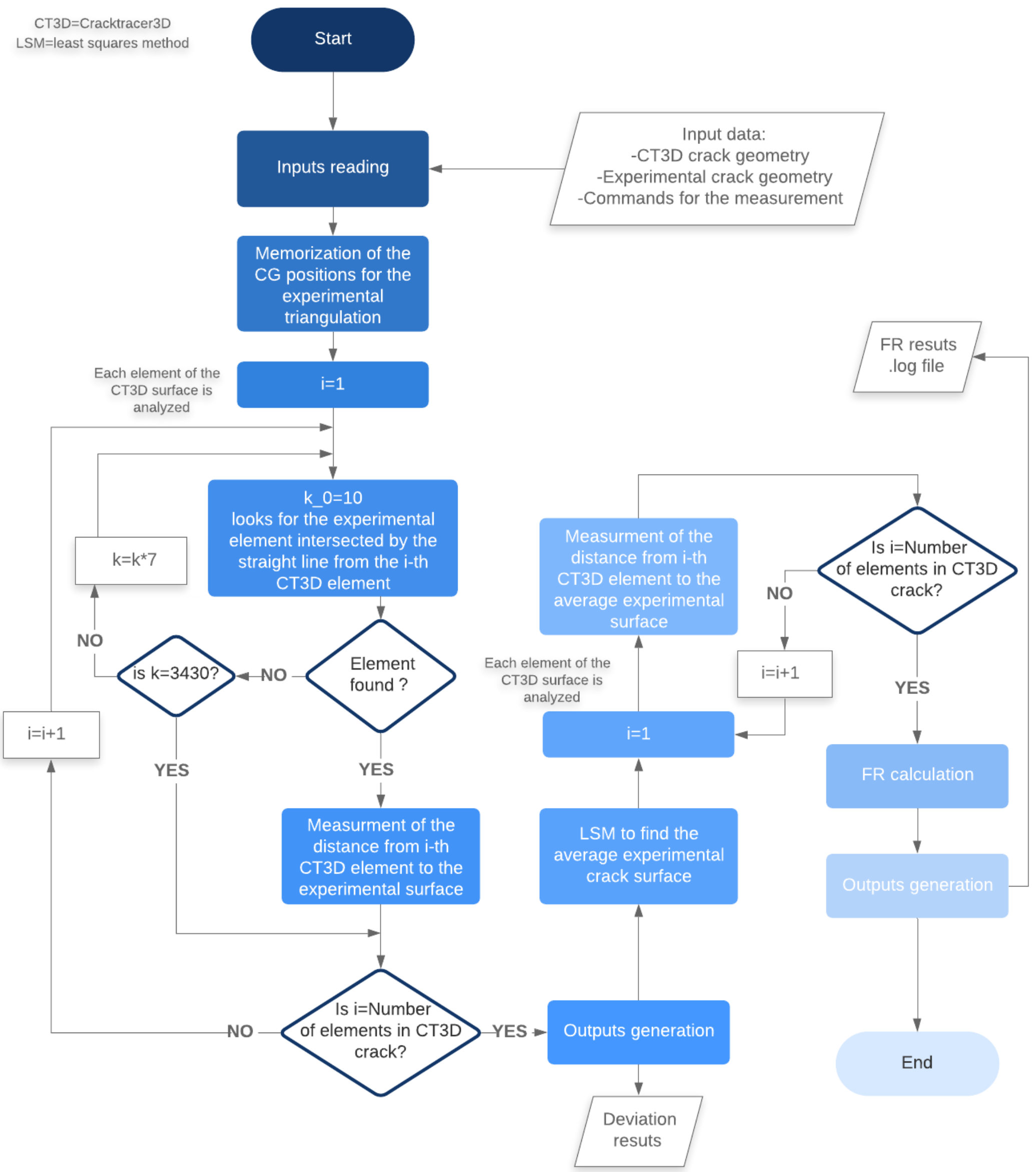

Figure A1.1: Organigram of CT3D_Validator

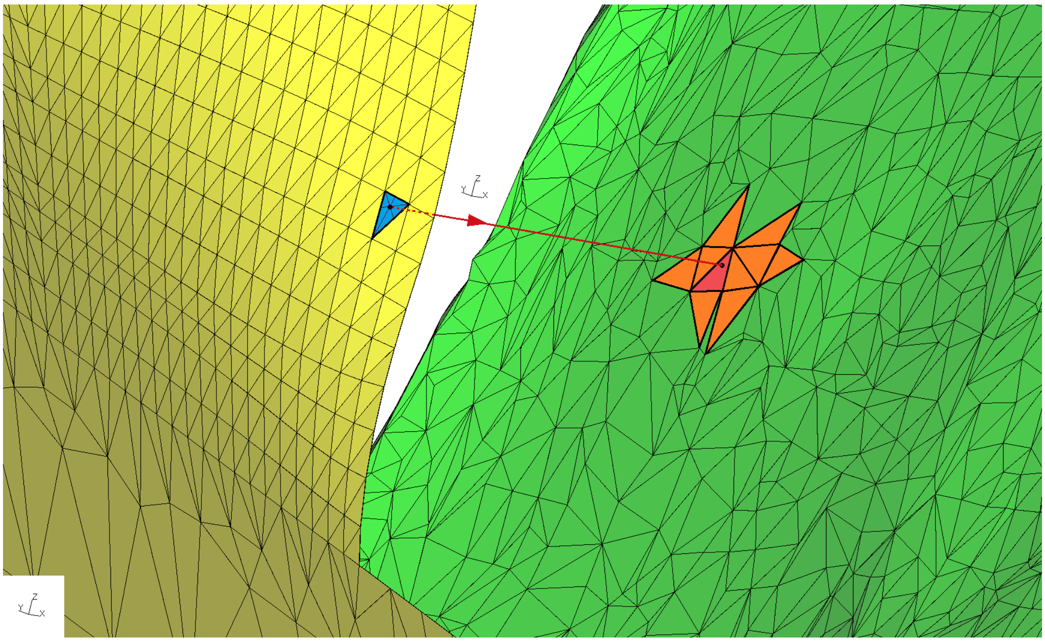

Figure A1.2: Measurement of the distance. In yellow the Cracktracer3D result, in green the experimental one. In this case the number of neighbors k is 10

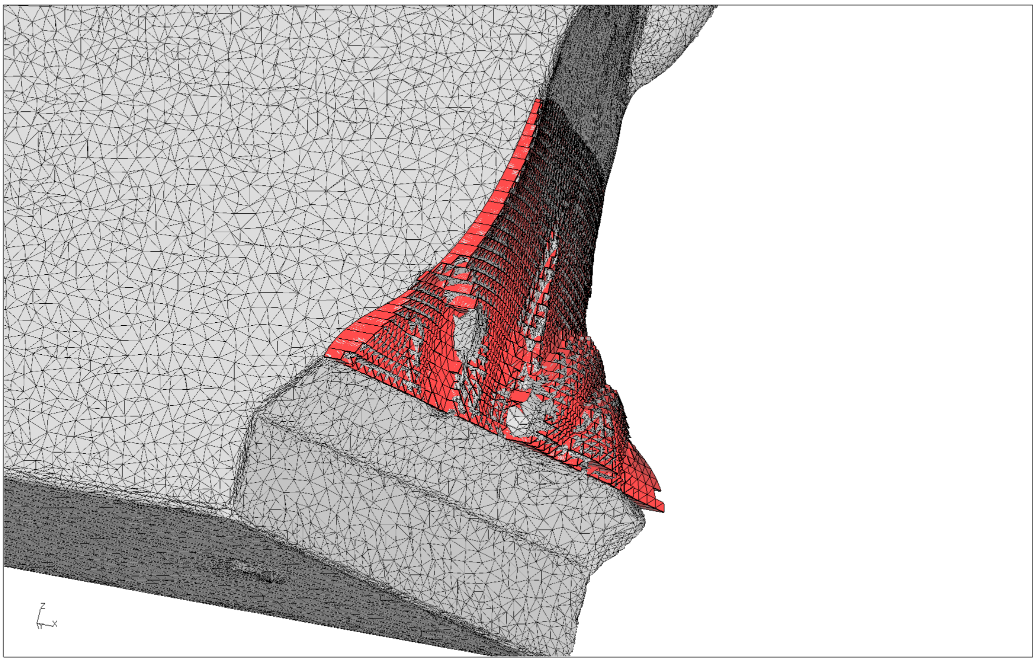

Figure A1.3: In red the volume between the Cracktracer3D crack surface and the experimental one

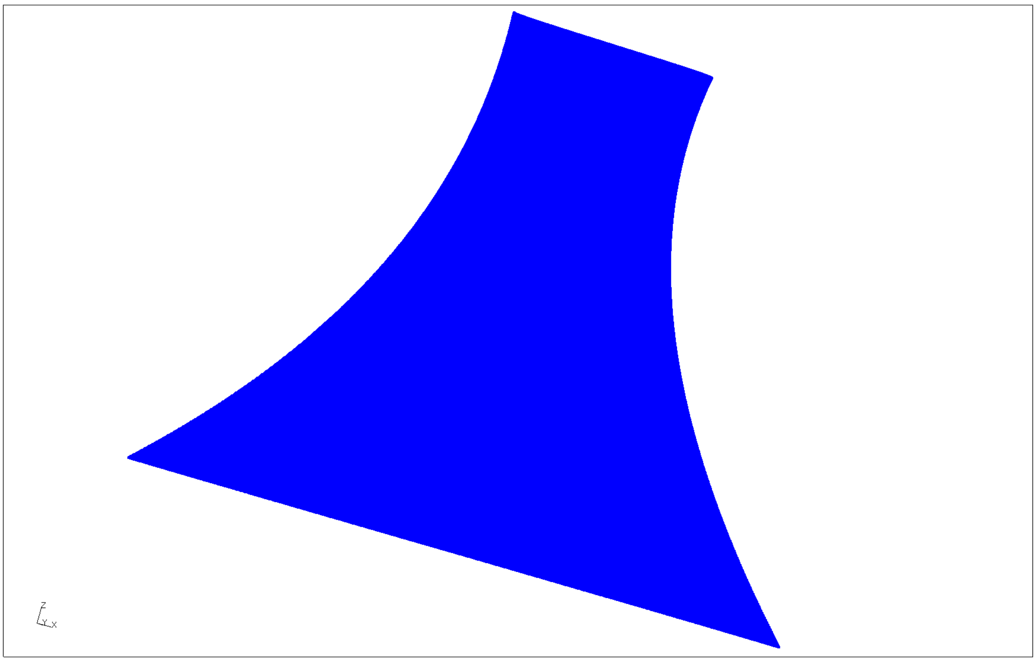

Figure A1.4: Average surface

A2 Main Code: ct3d_validator.f

The main code in fortran (free-form) of CT3D_Validator is shown below:

```
      program ct3d_validator
      implicit none
!
!
      real*8, allocatable  :: xct3d(:), yct3d(:), zct3d(:),&
                              xexp(:), yexp(:), zexp(:),&
                              dist(:), distfront(:), straight_exp(:,:),&
                              areact3d(:),areaexp(:),&
                              cgct3dx(:),cgct3dy(:),cgct3dz(:),&
                              cgexpx(:), cgexpy(:), cgexpz(:),&
                              cgexpx0(:),cgexpy0(:),cgexpz0(:),&
                              xsurf(:),ysurf(:),zsurf(:),&
                              xsurfexp(:),ysurfexp(:),zsurfexp(:)
      real*8               :: d, dave1, dave2, fdummy, tpar,vert(3,3),&
                              check1, check2, check3, straight_ct3d(16),&
                              preal, xp, yp, zp,&
                              devfr,A(10,10),B(10),aa(10),coeff(10),b1,&
                              xlong(800),ylong(800),zlong(800),&
                              zmin,zmax,xmin,xmax,ymin,ymax,&
                              devffn,devffd,devff,devlfn,devlfd,devlf,distfr
      integer, allocatable :: numnct3d(:), numnexp(:), numelct3d(:),&
                              n1ct3d(:), n2ct3d(:), n3ct3d(:), numelexp(:),&
                              n1exp(:), n2exp(:), n3exp(:), nx(:),&
                              ny(:), nz(:), neigh(:), elcract3d(:),&
                              lastfront(:), firstfront(:),&
                              isdist(:),nodeexppos(:),nodect3dpos(:)
      integer              :: i, j, dummy, nnodct3d, nnodexp, nelct3d,&
                              nelexp, kflag, kneigh, t, dt(8),values(8),&
                              nsetct3d, nsets, nelcract3d,&
                              h, hc, pint, pint1, count1, point,&
                              nlastfront,nfirstfront, contisdist, ref,&
                              nmaxnode,nmaxcheck, nind,kneighbefore
      character            :: text*128, textn*128, direction*1,&
                              textlastset*128, cra_plane*2,&
                              parallel*1,dir_par*2,textfistset*128
      logical              :: lnsetct3d, islastfront,isfirstfront, logic
!
!
!
      open(1,file ='CT3D_validator.log',status='replace',action='write')
!
      write(*,*)"##############################################################################&
##########################"
      write(1,*)"##############################################################################&
##########################"
      write(*,*)"#                                     ct3d_validator                   &
                      #"
      write(1,*)"#                                     ct3d_validator                   &
                      #"
      write(*,*)"#                                      Jury Rodella                    &
                      #"
      write(1,*)"#                                      Jury Rodella                    &
```

```
                    #"
      write(*,*)"#                              (12.08.2020)              &
                    #"
      write(1,*)"#                              (12.08.2020)              &
                    #"
      write(*,*)"##########################################################&
####################"
      write(1,*)"##########################################################&
####################"
      write(*,*)"This program calculates the crack propagation deviation&
 between the numerical"
      write(1,*)"This program calculates the crack propagation deviation&
 between the numerical"
      write(*,*)"and experimental results.                                &
                    "
      write(1,*)"and experimental results.                                &
                    "
      write(*,*)"The offset within the propagation directions is measured&
 along a direction   "
      write(1,*)"The offset within the propagation directions is measured&
  along a direction   "
      write(*,*)"choosen by the user (x,y,z or locally normal to CT3D crack&
 surface).          "
      write(1,*)"choosen by the user (x,y,z or locally normal to CT3D crack&
 surface).          "
      write(*,*)"The Factory Roof FR is measured as deviation of the&
 experimental crack from  "
      write(1,*)"The Factory Roof FR is measured as deviation of the&
  experimental crack from  "
      write(*,*)"its average surface (computed by the Least Squares&
 Method).                "
      write(1,*)"its average surface (computed by the Least Squares&
  Method).                "
      write(*,*)"__________________________________________________________&
____________________"
      write(1,*)"__________________________________________________________&
____________________"
      write(*,*)"input files (an input folder is required):              &
                    "
      write(1,*)"input files (an input folder is required):              &
                    "
      write(*,*)"-ct3d.inc: Cracktracer3d crack geometry in abq format.   &
                    "
      write(1,*)"-ct3d.inc: Cracktracer3d crack geometry in abq format.   &
                    "
      write(*,*)"-exp.cra: measured experimental crack geometry in abq&
 format.                "
      write(1,*)"-exp.cra: measured experimental crack geometry in abq&
 format.                "
      write(*,*)"__________________________________________________________&
____________________"
      write(1,*)"__________________________________________________________&
____________________"
      write(*,*)"                                                         &
                    "
      write(1,*)"                                                         &
                    "
```

```
!
      call date_and_time(values=dt)
      write(*,'(i4, 5(a, i2.2))') dt(1), '/', dt(2), '/', dt(3), ' ', &
      dt(5), ':', dt(6), ':', dt(7)
      write(1,'(i4, 5(a, i2.2))') dt(1), '/', dt(2), '/', dt(3), ' ', &
      dt(5), ':', dt(6), ':', dt(7)
!
      write(*,'(A)')"Choose a direction to measure the deviation&
       ('x','y','z'or'n'):"
      write(1,'(A)')"Choose a direction to measure the deviation&
        ('x','y','z'or'n'):"
      read (*,*)direction
      write(1,'(A)')direction
!
      if ((direction.ne.'x').and.(direction.ne.'y')&
      .and.(direction.ne.'z').and.(direction.ne.'n'))then
         write(*,*)"Error typing the direction"
         write(1,*)"Error typing the direction"
      stop
      endif
      write(*,'(A)')"Is the initial crack parallel to the plane xy,yz or&
       zx? ('y' or 'n'):"
      write(1,'(A)')"Is the initial crack parallel to the plane xy,yz or&
       zx? ('y' or 'n'):"
      read (*,*)parallel
      write(1,'(A)')parallel
!
      if ((parallel.ne.'y').and.(parallel.ne.'n'))then
         write(*,*)"Error, type 'y' or 'n'"
         write(1,*)"Error, type 'y' or 'n'"
         stop
      elseif (parallel.eq.'y')then
         write(*,'(A)')"Choose the parallel plane: ('xy','yz' or 'zx'):"
         write(1,'(A)')"Choose the parallel plane: ('xy','yz' or 'zx'):"
         read (*,*)dir_par
         write(1,'(A)')dir_par
!
         if ((dir_par.ne.'xy').and.(dir_par.ne.'yz')&
         .and.(dir_par.ne.'zx'))then
            write(*,*)"Error typing the plane"
            write(1,*)"Error typing the plane"
            stop
         endif
!
      elseif (parallel.eq.'n')then
         write(*,'(A)')"Choose the perpendicular plane to the initial&
       crack ('xy','yz','zx'):"
         write(1,'(A)')"Choose the perpendicular plane to the initial&
          crack ('xy','yz','zx'):"
         read (*,*)cra_plane
         write(1,'(A)')cra_plane
!
         if ((cra_plane.ne.'xy').and.(cra_plane.ne.'yz')&
         .and.(cra_plane.ne.'zx'))then
            write(*,*)'Error typing the plane'
            write(1,*)'Error typing the plane'
            stop
```

```
          endif
        endif
!
        write(*,*)"                                                    &
                                    "
        write(1,*)"                                                    &
                                    "
        write(*,*)"Reading input files:"
        write(1,*)"Reading input files:"
!
!
!_____________________________________________________________________&
_________________
!       read experimental crack input | exp.cra
        nnodexp=0
        nelexp=0
        nmaxnode=0
!
        open(2,file ='input/exp.cra',status='old',action='read')
        do
          read(2,'(a)',end=1)text
          if(text(1:5).eq.'*NODE')then
            do
              read(2,'(a)',end=1)text
              read(text,*,err=2)nmaxcheck,fdummy,fdummy,fdummy
              nnodexp=nnodexp+1
              if (nmaxcheck.ge.nmaxnode)then
                nmaxnode=nmaxcheck
              endif
            enddo
          endif
 2        continue
          if(text(1:8).eq.'*ELEMENT')then
            do
              read(2,'(a)',end=1)text
              read(text,*,err=3)dummy,dummy,dummy,dummy
              nelexp=nelexp+1
            enddo
          endif
 3        continue
        enddo
 1      continue
!
        write(*,'(a50,i15)')"Total number of nodes in the measured crack:&
                ",nnodexp
        write(1,'(a50,i15)')"Total number of nodes in the measured crack:&
                        ",nnodexp
        write(*,'(a50,i15)')"Total number of elements in the measured crack:&
            ",nelexp
        write(1,'(a50,i15)')"Total number of elements in the measured crack:&
            ",nelexp
        allocate(numelexp(nelexp),n1exp(nelexp),n2exp(nelexp),n3exp(nelexp))
        allocate(numnexp(nnodexp),xexp(nnodexp),yexp(nnodexp),zexp(nnodexp))
        allocate(nx(nelexp),ny(nelexp),nz(nelexp))
        allocate(straight_exp(nelexp,16))
        allocate(nodeexppos(nmaxnode))
        allocate(cgexpx(nelexp),cgexpy(nelexp),cgexpz(nelexp),areaexp(nelexp))
```

```fortran
      allocate(cgexpx0(nelexp),cgexpy0(nelexp),cgexpz0(nelexp))
      preal=nelexp/10
      pint1=int(preal)-1
!
      rewind(2)
      do
        read(2,'(a)',end=5)text
        if(text(1:5).eq.'*NODE')then
          do i=1,nnodexp
            read(2,'(a)',end=5)text
            read(text,*,err=6)numnexp(i),xexp(i),yexp(i),zexp(i)
            nodeexppos(numnexp(i))=i
          enddo
        endif
        if(text(1:8).eq.'*ELEMENT')then
          do i=1,nelexp
            read(2,'(a)',end=5)text
            read(text,*,err=7)numelexp(i),n1exp(i),n2exp(i),n3exp(i)
          enddo
        endif
 6      continue
 7      continue
      enddo
 5    continue
      close(2)
!
!
!______________________________________________________________________________&
__________________
!     read Cracktracer3d crack input | ct3d.cra
      nnodct3d=0
      nelct3d=0
      lnsetct3d=.FALSE.
      nsets=0
      nelcract3d=0
      nmaxnode=0
      open(3,file ='input/ct3d.inc',status='old',action='read')
      do
        read(3,'(a)',end=9)text
        if(text(1:5).eq.'*NODE')then
          do
            read(3,'(a)',end=9)text
            read(text,*,err=10)nmaxcheck,fdummy,fdummy,fdummy
            nnodct3d=nnodct3d+1
            if (nmaxcheck.ge.nmaxnode)then
              nmaxnode=nmaxcheck
            endif
          enddo
        endif
 10     continue
        if(text(1:8).eq.'*ELEMENT')then
          do
            read(3,'(a)',end=9)text
            read(text,*,err=11)dummy,dummy,dummy,dummy
            nelct3d=nelct3d+1
          enddo
        endif
```

```
 11         continue
 12         continue
            if(text(1:6).eq.'*ELSET')then
               nsets=nsets+1
               textlastset=text
               if (nsets.eq.2)then
                  textfistset=text
               endif
               if (nsets.gt.1)then
                 do
                    read(3,'(a)',end=9)text
                    read(text,*,err=12)dummy
                    nelcract3d=nelcract3d+1
                    lnsetct3d=.TRUE.
                 enddo
               endif
            endif
         enddo
 9       continue
!
         write(*,'(a50,i15)')"Total number of nodes in Cracktracer3d crack:&
              ",nnodct3d
         write(1,'(a50,i15)')"Total number of nodes in Cracktracer3d crack:&
              ",nnodct3d
         write(*,'(a50,i15)')"Total number of elements in Cracktracer3d crack:&
           ",nelct3d
         write(1,'(a50,i15)')"Total number of elements in Cracktracer3d crack:&
           ",nelct3d
         allocate(numnct3d(nnodct3d),xct3d(nnodct3d),yct3d(nnodct3d),&
         zct3d(nnodct3d))
         allocate(numelct3d(nelct3d),n1ct3d(nelct3d),n2ct3d(nelct3d),&
         n3ct3d(nelct3d))
!
         if(lnsetct3d)then
            allocate(elcract3d(nelcract3d))
            allocate(xsurf(nelcract3d),ysurf(nelcract3d),zsurf(nelcract3d))
            allocate(xsurfexp(nelcract3d),ysurfexp(nelcract3d),&
            zsurfexp(nelcract3d))
            allocate(isdist(nelcract3d))
            allocate(cgct3dx(nelcract3d),cgct3dy(nelcract3d),&
            cgct3dz(nelcract3d),areact3d(nelcract3d))
            allocate(dist(nelcract3d))
            allocate(nodect3dpos(nmaxnode))
            preal=nelcract3d/10
            pint=int(preal)-1
            write(*,'(a50,i15)')"Number of elements in Cracktracer3d crack:&
                    ",nelcract3d
            write(1,'(a50,i15)')"Number of elements in Cracktracer3d crack:&
                    ",nelcract3d
         else
            write (*,*)"NSETs in ct3d.cra not found"
            write (1,*)"NSETs in ct3d.cra not found"
         endif
         rewind(3)
         j=0
         nsets=0
         do
```

```
          read(3,'(a)',end=13)text
          if(text(1:5).eq.'*NODE')then
             do i=1,nnodct3d
                read(3,'(a)',end=13)text
                read(text,*,err=14)numnct3d(i),xct3d(i),yct3d(i),zct3d(i)
                nodect3dpos(numnct3d(i))=i
             enddo
          endif
          if(text(1:8).eq.'*ELEMENT')then
             do i=1,nelct3d
                read(3,'(a)',end=13)text
                read(text,*,err=15)numelct3d(i),n1ct3d(i),n2ct3d(i),n3ct3d(i)
             enddo
          endif
 16       continue
          if(text(1:6).eq.'*ELSET')then
             nsets=nsets+1
             if (nsets.gt.1)then
                do
                   read(3,'(a)',end=13)text
                   read(text,*,err=16)dummy
                   j=j+1
                   elcract3d(j)=dummy
                enddo
             endif
          endif
 14       continue
 15       continue
       enddo
 13    continue
!      read and store first/last front information
       rewind(3)
       nlastfront=0
       nfirstfront=0
       do
          read(3,'(a)',end=17)text
 18       continue
          if(text.eq.textfistset)then
             do
                read(3,'(a)',end=17)text
                read(text,*,err=18)dummy
                nfirstfront=nfirstfront+1
             enddo
          endif
          if(text.eq.textlastset)then
             do
                read(3,'(a)',end=17)text
                read(text,*,err=18)dummy
                nlastfront=nlastfront+1
             enddo
          endif
       enddo
 17    continue
       if(lnsetct3d)then
          allocate(lastfront(nlastfront))
          allocate(firstfront(nfirstfront))
       endif
```

```
      rewind(3)
      i=0
      j=0
      do
         read(3,'(a)',end=20)text
 19      continue
         if(text.eq.textfistset)then
            do
               read(3,'(a)',end=20)text
               read(text,*,err=19)dummy
               i=i+1
               firstfront(i)=dummy
            enddo
         endif
         if(text.eq.textlastset)then
            do
               read(3,'(a)',end=20)text
               read(text,*,err=19)dummy
               j=j+1
               lastfront(j)=dummy
            enddo
         endif
      enddo
 20   continue
      close(3)
!
!
!________________________________________________________________________________&
______________________
!     measurement of crack deviation
      if (lnsetct3d)then
         h=pint1
         hc=0
         do i=1,nelexp
            nx(i)=i
            ny(i)=i
            nz(i)=i
         enddo
         write(*,*)"                                                                   &
                                  "
         write(1,*)"                                                                   &
                                  "
         write(*,*)"Sort and allocation of experimental elements &
         (progress):                    "
         write(*,*)"______________________                                             &
                                  "
         write(*,'(A)',advance='no')"|"
!     cg and area of measured crack elements
         do i=1,nelexp
            if (h.eq.i)then
!     progress bar generation
               hc=hc+1
               h=h+pint1
               if (hc.ne.10)then
                  write(*,'(a)',advance='no')"##"
               endif
               if (hc.eq.10)then
```

```
                write (*,'(a)')"##|100%"
                call sleep(1)
             endif
          endif
          numelexp(i)=i
          nind=nodeexppos(n1exp(i))
          vert(1,1)=xexp(nind)
          vert(2,1)=yexp(nind)
          vert(3,1)=zexp(nind)
          nind=nodeexppos(n2exp(i))
          vert(1,2)=xexp(nind)
          vert(2,2)=yexp(nind)
          vert(3,2)=zexp(nind)
          nind=nodeexppos(n3exp(i))
          vert(1,3)=xexp(nind)
          vert(2,3)=yexp(nind)
          vert(3,3)=zexp(nind)
          call tricgarea(vert,cgexpx(i),cgexpy(i),cgexpz(i),areaexp(i))
          call straighteq3d(vert,straight_exp(i,:),'n')
       enddo
       kflag=2
       cgexpx0=cgexpx
       cgexpy0=cgexpy
       cgexpz0=cgexpz
       call dsort(cgexpx,nx,nelexp,kflag)
       call dsort(cgexpy,ny,nelexp,kflag)
       call dsort(cgexpz,nz,nelexp,kflag)
!
       open(40,file ='lines.fdb',status='replace',action='write')
       open(50,file ='prisms.fdb',status='replace',action='write')
       open(60,file ='flfront.fdb',status='replace',action='write')
       write(*,*)"                                                       &
                         "
       write(*,*)"Measurement (progress):                                &
                         "
       write(*,*)"____________________                                   &
                         "
       write(*,'(A)',advance='no')"|"
       kneigh=3430
       h=pint
       hc=0
       count1=0
       dave1=0.d0
       dave2=0.d0
       devffn=0.d0
       devffd=0.d0
       devlfn=0.d0
       devlfd=0.d0
       contisdist=0
       allocate(neigh(kneigh))
       do i=1,nelcract3d
!      progress bar generation
          if (h.eq.i)then
             hc=hc+1
             h=h+pint
             if (hc.ne.10)then
                write(*,'(a)',advance='no')"##"
```

```fortran
          endif
          if (hc.eq.10) then
             write (*,'(a)')"##|100%"
             call sleep(1)
          endif
       endif
       nind=nodect3dpos(n1ct3d(elcract3d(i)))
       vert(1,1)=xct3d(nind)
       vert(2,1)=yct3d(nind)
       vert(3,1)=zct3d(nind)
       nind=nodect3dpos(n2ct3d(elcract3d(i)))
       vert(1,2)=xct3d(nind)
       vert(2,2)=yct3d(nind)
       vert(3,2)=zct3d(nind)
       nind=nodect3dpos(n3ct3d(elcract3d(i)))
       vert(1,3)=xct3d(nind)
       vert(2,3)=yct3d(nind)
       vert(3,3)=zct3d(nind)
       islastfront=.FALSE.
       isfirstfront=.FALSE.
       do j=1,nlastfront
          if (elcract3d(i).eq.lastfront(j)) then
             islastfront=.TRUE.
          endif
       enddo
       do j=1,nfirstfront
          if (elcract3d(i).eq.firstfront(j)) then
             isfirstfront=.TRUE.
          endif
       enddo
       call tricgarea(vert,cgct3dx(i),cgct3dy(i),cgct3dz(i),&
       areact3d(i))
       call straighteq3d(vert,straight_ct3d,direction)
!
       kneigh=10
       logic=.FALSE.
       kneighbefore=0
       do while((.not.(logic)).and.(kneigh.lt.3430))
          kneigh=kneigh*7
!
          call near3d(cgexpx0,cgexpy0,cgexpz0,cgexpx,cgexpy,&
          cgexpz,nx,ny,nz,cgct3dx(i),cgct3dy(i),cgct3dz(i),&
          nelexp,neigh,kneigh)
!
          logic=.FALSE.
          t=kneighbefore
          tpar=0.d0
          kneighbefore=kneigh
          do while ((t.lt.kneigh).and.(.not.(logic)))
             t=t+1
             point=neigh(t)
             tpar=-(straight_exp(point,13)&
             *cgct3dx(i)+straight_exp(point,14)*cgct3dy(i)&
             +straight_exp(point,15)*cgct3dz(i)&
             +straight_exp(point,16))/&
             (straight_exp(point,13)*straight_ct3d(13)&
             +straight_exp(point,14)*&
```

```
                          straight_ct3d(14)+straight_exp(point,&
                          15)*straight_ct3d(15))
!
                          xp=cgct3dx(i)+straight_ct3d(13)*tpar
                          yp=cgct3dy(i)+straight_ct3d(14)*tpar
                          zp=cgct3dz(i)+straight_ct3d(15)*tpar
!
                          xsurf(i)=xp
                          ysurf(i)=yp
                          zsurf(i)=zp
                          check1=xp*straight_exp(point,1)+yp*straight_exp(point,2)&
                          +zp*straight_exp(point,3)+straight_exp(point,4)
                          check2=xp*straight_exp(point,5)+yp*straight_exp(point,6)&
                          +zp*straight_exp(point,7)+straight_exp(point,8)
                          check3=xp*straight_exp(point,9)+yp*straight_exp(point,10)&
                          +zp*straight_exp(point,11)+straight_exp(point,12)
!
                          if ((check1.le.0).and.(check2.le.0).and.(check3.le.0)&
                          .and.(.not.(logic)))then
                              point=neigh(t)
                              ref=i
                              logic=.TRUE.
                              count1=count1+1
                          endif
                      enddo
!
                      if (logic)then
                          dist(i) =((cgct3dx(i)-xp)**2+(cgct3dy(i)-yp)**2&
                          +(cgct3dz(i)-zp)**2)**(0.5)
!
                          check3=xp*straight_ct3d(13)+yp*straight_ct3d(14)&
                          +zp*straight_ct3d(15)+straight_ct3d(16)
!
                          if (check3.le.0)then
                              dist(i)=-dist(i)
                          endif
                          isdist(i)=1
                          contisdist=contisdist+1
                          write(40,'("PNT p",I0,"  ",F20.8 ,"  ",F20.8 ,"  ",F20.8 )')&
                          i,cgct3dx(i),cgct3dy(i),cgct3dz(i)
                          write(40,'("PNT p",I0,"  ",F20.8 ,"  ",F20.8 ,"  ",F20.8 )')&
                          i+nelct3d,xp,yp,zp
                          write(40,'("LINE L",I0,"  p",I0,"  p",I0,"  2")')&
                          i,i,i+nelct3d
!
                          write(50,'("PNT p",I0,"  ",F20.8 ,"  ",F20.8 ,"  ",F20.8 )')&
                          i, vert(1,1),vert(2,1),vert(3,1)
                          write(50,'("PNT p",I0,"  ",F20.8 ,"  ",F20.8 ,"  ",F20.8 )')&
                          i+nelct3d, vert(1,2),vert(2,2),vert(3,2)
                          write(50,'("PNT p",I0,"  ",F20.8 ,"  ",F20.8 ,"  ",F20.8 )')&
                          i+2*nelct3d,vert(1,3),vert(2,3),vert(3,3)
                          write(50,'("PNT p",I0,"  ",F20.8 ,"  ",F20.8 ,"  ",F20.8 )')&
                          i+3*nelct3d,vert(1,1)+dist(i)*straight_ct3d(13),&
                          vert(2,1)+dist(i)*straight_ct3d(14),&
                          vert(3,1)+dist(i)*straight_ct3d(15)
                          write(50,'("PNT p",I0,"  ",F20.8 ,"  ",F20.8 ,"  ",F20.8 )')&
                          i+4*nelct3d,vert(1,2)+dist(i)*straight_ct3d(13),&
```

```
                      vert(2,2)+dist(i)*straight_ct3d(14),&
                      vert(3,2)+dist(i)*straight_ct3d(15)
                      write(50,'("PNT p",I0,"  ",F20.8  ,"  ",F20.8  ,"  ",F20.8  )')&
                      i+5*nelct3d,vert(1,3)+dist(i)*straight_ct3d(13),&
                      vert(2,3)+dist(i)*straight_ct3d(14),&
                      vert(3,3)+dist(i)*straight_ct3d(15)
                      write(50,'("LINE L",I0," p",I0," p",I0," 2")')&
                      i,i,i+nelct3d
                      write(50,'("LINE L",I0," p",I0," p",I0," 2")')&
                      i+nelct3d,i+nelct3d,i+2*nelct3d
                      write(50,'("LINE L",I0," p",I0," p",I0," 2")')&
                      i+2*nelct3d,i+2*nelct3d,i
                      write(50,'("SURF S",I0," L",I0," L",I0," L",I0)')&
                      i,i,i+nelct3d, i+2*nelct3d
                      write(50,'("LINE L",I0," p",I0," p",I0," 2")')&
                      i+3*nelct3d,i+3*nelct3d,i+4*nelct3d
                      write(50,'("LINE L",I0," p",I0," p",I0," 2")')&
                      i+4*nelct3d,i+4*nelct3d,i+5*nelct3d
                      write(50,'("LINE L",I0," p",I0," p",I0," 2")')&
                      i+5*nelct3d,i+5*nelct3d,i+3*nelct3d
                      write(50,'("SURF S",I0," L",I0," L",I0," L",I0)')&
                      i+nelct3d,i+3*nelct3d,i+4*nelct3d, i+5*nelct3d
                      write(50,'("LINE L",I0," p",I0," p",I0," 2")')&
                      i+6*nelct3d,i,i+3*nelct3d
                      write(50,'("LINE L",I0," p",I0," p",I0," 2")')&
                      i+7*nelct3d,i+nelct3d,i+4*nelct3d
                      write(50,'("LINE L",I0," p",I0," p",I0," 2")')&
                      i+8*nelct3d,i+2*nelct3d,i+5*nelct3d
                      write(50,'("SURF S",I0," L",I0," L",I0," L",I0," L",I0)')&
                      i+2*nelct3d,i,i+6*nelct3d, i+3*nelct3d,i+7*nelct3d
                      write(50,'("SURF S",I0," L",I0," L",I0," L",I0," L",I0)')&
                      i+3*nelct3d,i+nelct3d,i+7*nelct3d,i+4*nelct3d, i+8*nelct3d
                      write(50,'("SURF S",I0," L",I0," L",I0," L",I0," L",I0)')&
                      i+4*nelct3d,i+2*nelct3d,i+8*nelct3d,&
                      i+5*nelct3d,i+6*nelct3d
                      write(50,'("gbod B",I0," NORM + S",I0," + S",I0," +&
                       S",I0," + S",I0," + S",I0)')i,i,i+2*nelct3d,i+3*nelct3d,&
                      i+4*nelct3d,i+1*nelct3d
!
                      if(islastfront.or.isfirstfront)then
                         write(60,'("PNT p",I0,"  ",F20.8  ,"  ",F20.8  ,"  ",&
                      F20.8  )') i, vert(1,1),vert(2,1),vert(3,1)
                         write(60,'("PNT p",I0,"  ",F20.8  ,"  ",F20.8  ,"  ",&
                         F20.8  )')  i+nelct3d, vert(1,2),vert(2,2),vert(3,2)
                         write(60,'("PNT p",I0,"  ",F20.8  ,"  ",F20.8  ,"  ",&
                         F20.8  )')   i+2*nelct3d,vert(1,3),vert(2,3),vert(3,3)
                         write(60,'("PNT p",I0,"  ",F20.8  ,"  ",F20.8  ,"  ",&
                         F20.8  )')i+3*nelct3d,vert(1,1)+dist(i)&
                         *straight_ct3d(13), vert(2,1)+dist(i)*&
                         straight_ct3d(14),vert(3,1)+dist(i)*straight_ct3d(15)
                         write(60,'("PNT p",I0,"  ",F20.8  ,"  ",F20.8  ,"  ",&
                         F20.8  )')&
                         i+4*nelct3d,vert(1,2)+dist(i)*straight_ct3d(13),&
                         vert(2,2)+dist(i)*straight_ct3d(14),&
                         vert(3,2)+dist(i)*straight_ct3d(15)
                         write(60,'("PNT p",I0,"  ",F20.8  ,"  ",F20.8  ,"  ",&
                         F20.8  )')&
```

```
                              i+5*nelct3d,vert(1,3)+dist(i)*straight_ct3d(13),&
                              vert(2,3)+dist(i)*straight_ct3d(14),&
                              vert(3,3)+dist(i)*straight_ct3d(15)
                              write(60,'("LINE L",I0," p",I0," p",I0," 2")')&
                              i,i,i+nelct3d
                              write(60,'("LINE L",I0," p",I0," p",I0," 2")')&
                              i+nelct3d,i+nelct3d,i+2*nelct3d
                              write(60,'("LINE L",I0," p",I0," p",I0," 2")')&
                              i+2*nelct3d,i+2*nelct3d,i
                              write(60,'("SURF S",I0," L",I0," L",I0," L",I0)')&
                              i,i,i+nelct3d, i+2*nelct3d
                              write(60,'("LINE L",I0," p",I0," p",I0," 2")')&
                              i+3*nelct3d,i+3*nelct3d,i+4*nelct3d
                              write(60,'("LINE L",I0," p",I0," p",I0," 2")')&
                              i+4*nelct3d,i+4*nelct3d,i+5*nelct3d
                              write(60,'("LINE L",I0," p",I0," p",I0," 2")')&
                              i+5*nelct3d,i+5*nelct3d,i+3*nelct3d
                              write(60,'("SURF S",I0," L",I0," L",I0," L",I0)')&
                              i+nelct3d,i+3*nelct3d,i+4*nelct3d, i+5*nelct3d
                              write(60,'("LINE L",I0," p",I0," p",I0," 2")')&
                              i+6*nelct3d,i,i+3*nelct3d
                              write(60,'("LINE L",I0," p",I0," p",I0," 2")')&
                              i+7*nelct3d,i+nelct3d,i+4*nelct3d
                              write(60,'("LINE L",I0," p",I0," p",I0," 2")')&
                              i+8*nelct3d,i+2*nelct3d,i+5*nelct3d
                              write(60,'("SURF S",I0," L",I0," L",I0," L",I0&
                              ," L",I0)')&
                              i+2*nelct3d,i,i+6*nelct3d, i+3*nelct3d,&
                              i+7*nelct3d
                              write(60,'("SURF S",I0," L",I0," L",I0," L",I0&
                              ," L",I0)')&
                              i+3*nelct3d,i+nelct3d,i+7*nelct3d,&
                              i+4*nelct3d, i+8*nelct3d
                              write(60,'("SURF S",I0," L",I0," L",I0," L",I0&
                              ," L",I0)')&
                              i+4*nelct3d,i+2*nelct3d,i+8*nelct3d,&
                              i+5*nelct3d,i+6*nelct3d
                              write(60,'("gbod B",I0," NORM + S",I0," + S",I0&
                              ," + S",I0," + S",I0,"&
                               + S",I0)')i,i,i+2*nelct3d,i+3*nelct3d,i+4*nelct3d&
                               ,i+1*nelct3d
                          endif
                          if(isfirstfront)then
                              devffn=devffn+dabs(dist(i))*areact3d(i)
                              devffd=devffd+areact3d(i)
                          endif
                          if(islastfront)then
                              devlfn=devlfn+dabs(dist(i))*areact3d(i)
                              devlfd=devlfd+areact3d(i)
                          endif
                      else
                       !  areact3d(i)=0
                      endif
                  enddo
              enddo
              write(40,'("plus l all")')
              write(50,'("elty all he8")')
```

```
         write(50,'("mesh all")')
         write(60,'("elty all he8")')
         write(60,'("mesh all")')
         close(40)
         close(50)
         close(60)
      endif
      devff=devffn/devffd
      devlf=devlfn/devlfd
      write(*,*)"                                                                        &
                          "
      write(1,*)"                                                                        &
                          "
      write(*,'(a50,i15)')"Number of elements used for the measurement:   &
         ",count1
      write(1,'(a50,i15)')"Number of elements used for the measurement:   &
            ",count1
      write(*,*)"                                                                        &
                          "
      write(1,*)"                                                                        &
                          "
      dave1=0.d0
      dave2=0.d0
      do i=1,nelcract3d
         if (isdist(i).eq.1)then
            dave1=dave1+dabs(dist(i))*areact3d(i)
            dave2=dave2+areact3d(i)
         endif
      enddo
      d=dave1/dave2
!
!
!_________________________________________________________________________&
____________________
!     measurement of average crack surface
      do i=1,10
         do j=1,10
            A(i,j)=0.d0
         enddo
         B(i)=0.d0
      enddo
      xmin=cgct3dx(ref)
      xmax=cgct3dx(ref)
      ymin=cgct3dy(ref)
      ymax=cgct3dy(ref)
      zmin=cgct3dz(ref)
      zmax=cgct3dz(ref)
      do i=1,nelcract3d
         if (((cra_plane.eq.'xy').or.(dir_par.eq.'yz')).and.(isdist(i)&
         .eq.1))then
            xp=ysurf(i)
            yp=zsurf(i)
            zp=xsurf(i)
         endif
         if (((cra_plane.eq.'yz').or.(dir_par.eq.'zx')).and.(isdist(i)&
         .eq.1))then
            xp=xsurf(i)
```

```
            yp=zsurf(i)
            zp=ysurf(i)
         endif
         if (((cra_plane.eq.'zx').or.(dir_par.eq.'xy')).and.(isdist(i)&
         .eq.1))then
            xp=xsurf(i)
            yp=ysurf(i)
            zp=zsurf(i)
         endif
!      localize crack surface of ct3d
         if (xsurf(i).ge.xmax)then
            xmax=xsurf(i)
         endif
         if (xsurf(i).le.xmin)then
            xmin=xsurf(i)
         endif
         if (ysurf(i).ge.ymax)then
            ymax=ysurf(i)
         endif
         if (ysurf(i).le.ymin)then
            ymin=ysurf(i)
         endif
         if (zsurf(i).ge.zmax)then
            zmax=zsurf(i)
         endif
         if (zsurf(i).le.zmin)then
            zmin=zsurf(i)
         endif
!      surface coeff
         aa(1)=xp**3
         aa(2)=yp**3
         aa(3)=(xp**2)*yp
         aa(4)=(yp**2)*xp
         aa(5)=xp**2
         aa(6)=yp**2
         aa(7)=xp*yp
         aa(8)=xp
         aa(9)=yp
         aa(10)=1
         b1=zp
!      matrix LSM
         do j=1,10
            do t=1,10
               A(j,t)=A(j,t)+aa(t)*aa(j)
            enddo
            B(j)=B(j)+b1*aa(j)
         enddo
      enddo
      call gauss_1(A,B,coeff,10)
      open(7,file ='avesurf.fdb',status='replace',action='write')
      open(8,file ='devfr.fdb',status='replace',action='write')
      t=0
      xlong(1)=xmin
      do i=2,800
         xlong(i)=xlong(i-1)+(xmax-xmin)/800.d0
      enddo
      ylong(1)=ymin
```

```
      do i=2,800
         ylong(i)=ylong(i-1)+(ymax-ymin)/800.d0
      enddo
      zlong(1)=zmin
      do i=2,800
         zlong(i)= zlong(i-1)+(zmax-zmin)/800.d0
      enddo
      dave1=0
      dave2=0
      if ((cra_plane.eq.'xy').or.(dir_par.eq.'yz'))then
         do i=1,nelcract3d
            if(isdist(i).eq.1)then
               xp=ysurf(i)
               yp=zsurf(i)
               zp=coeff(1)*xp**3+coeff(2)*yp**3+coeff(3)*&
               (xp**2)*yp+coeff(4)*(yp**2)*xp+&
               coeff(5)*xp**2+coeff(6)*yp**2+coeff(7)*xp*&
               yp+coeff(8)*xp+coeff(9)*yp+coeff(10)
               distfr=((zp-xsurf(i))**2)**(0.5)
               dave1=dave1+dabs(distfr)*areact3d(i)
               dave2=dave2+areact3d(i)
               write(8,'("PNT p",I0," ",F20.8 ," ",F20.8 ," ",F20.8&
               )')i,zp,xp,yp
               write(8,'("PNT p",I0," ",F20.8 ," ",F20.8 ," ",F20.8&
               )')i+nelct3d,xsurf(i),xp,yp
               write(8,'("LINE L",I0," p",I0," p",I0," 2")')i,i,i+nelct3d
            endif
         enddo
         do i=1,800
            do j=1,800
               t=t+1
               xp=ylong(j)
               yp=zlong(i)
               zp=coeff(1)*xp**3+coeff(2)*yp**3+coeff(3)*&
               (xp**2)*yp+coeff(4)*(yp**2)*xp+&
               coeff(5)*xp**2+coeff(6)*yp**2+coeff(7)*xp*&
               yp+coeff(8)*xp+coeff(9)*yp+coeff(10)
               write(7,'("PNT p",I0," ",F20.8 ," ",F20.8 ," ",F20.8&
               )')t,zp,xp,yp
            enddo
         enddo
      endif
      if ((cra_plane.eq.'yz').or.(dir_par.eq.'zx'))then
         do i=1,nelcract3d
            if(isdist(i).eq.1)then
               xp=xsurf(i)
               yp=zsurf(i)
               zp=coeff(1)*xp**3+coeff(2)*yp**3+coeff(3)*&
               (xp**2)*yp+coeff(4)*(yp**2)*xp+&
               coeff(5)*xp**2+coeff(6)*yp**2+coeff(7)*xp*&
               yp+coeff(8)*xp+coeff(9)*yp+coeff(10)
               distfr=((zp-ysurf(i))**2)**(0.5)
               dave1=dave1+dabs(distfr)*areact3d(i)
               dave2=dave2+areact3d(i)
               write(8,'("PNT p",I0," ",F20.8 ," ",F20.8 ," ",F20.8&
               )')i,xp,zp,yp
               write(8,'("PNT p",I0," ",F20.8 ," ",F20.8 ," ",F20.8&
```

```
                  )')i+nelct3d,xp,ysurf(i),yp
                 write(8,'("LINE L",I0," p",I0," p",I0," 2")')i,i,i+nelct3d
              endif
           enddo
           do i=1,800
              do j=1,800
                 t=t+1
                 xp=xlong(j)
                 yp=zlong(i)
                 zp=coeff(1)*xp**3+coeff(2)*yp**3+coeff(3)*&
                 (xp**2)*yp+coeff(4)*(yp**2)*xp+&
                 coeff(5)*xp**2+coeff(6)*yp**2+coeff(7)*xp*&
                 yp+coeff(8)*xp+coeff(9)*yp+coeff(10)
                 write(7,'("PNT p",I0," ",F20.8 ," ",F20.8 ," ",F20.8&
                  )')t,xp,zp,yp
              enddo
           enddo
        endif
        if ((cra_plane.eq.'zx').or.(dir_par.eq.'xy'))then
           do i=1,nelcract3d
              if(isdist(i).eq.1)then
                 xp=xsurf(i)
                 yp=ysurf(i)
                 zp=coeff(1)*xp**3+coeff(2)*yp**3+coeff(3)*&
                 (xp**2)*yp+coeff(4)*(yp**2)*xp+&
                 coeff(5)*xp**2+coeff(6)*yp**2+coeff(7)*xp*&
                 yp+coeff(8)*xp+coeff(9)*yp+coeff(10)
                 distfr=((zp-zsurf(i))**2)**(0.5)
                 dave1=dave1+dabs(distfr)*areact3d(i)
                 dave2=dave2+areact3d(i)
                 write(8,'("PNT p",I0," ",F20.8 ," ",F20.8 ," ",F20.8&
                  )')i,xp,yp,zp
                 write(8,'("PNT p",I0," ",F20.8 ," ",F20.8 ," ",F20.8&
                  )')i+nelct3d,xp,yp,zsurf(i)
                 write(8,'("LINE L",I0," p",I0," p",I0," 2")')i,i,i+nelct3d
              endif
           enddo
           do i=1,800
              do j=1,800
                 t=t+1
                 xp=xlong(j)
                 yp=ylong(i)
                 zp=coeff(1)*xp**3+coeff(2)*yp**3+coeff(3)*&
                 (xp**2)*yp+coeff(4)*(yp**2)*xp+&
                 coeff(5)*xp**2+coeff(6)*yp**2+coeff(7)*xp*&
                 yp+coeff(8)*xp+coeff(9)*yp+coeff(10)
                 write(7,'("PNT p",I0," ",F20.8 ," ",F20.8 ," ",F20.8&
                  )')t,xp,yp,zp
              enddo
           enddo
        endif
        write(7,'("plus p all")')
        write(8,'("plus l all")')
        close(7)
        close(8)
        devfr=dave1/dave2
!
```

```
      write(*,'("Average deviation on ",A," direction:&
            ",F20.8)')direction,d
      write(1,'("Average deviation on ",A," direction:&
            ",F20.8)')direction,d
      write(*,'("Crack initial front deviation on ",A,"&
       direction:",F20.8)')direction,devff
      write(1,'("Crack initial front deviation on ",A,"&
       direction:",F20.8)')direction,devff
      write(*,'("Crack final front deviation on ",A," direction:&
        ",F20.8)')direction,devlf
      write(1,'("Crack final front deviation on ",A," direction:&
        ",F20.8)')direction,devlf
      write(*,'("Factory Roof:&
                                    ",F20.8)')devfr
      write(1,'("Factory Roof:&
                                    ",F20.8)')devfr
      close(1)
!
!
!
      end program ct3d_validator
```