

POLITECNICO DI TORINO

Corso di Laurea Magistrale
in Ingegneria Biomedica

Tesi di Laurea Magistrale

Design of a portable system for monitoring and analysis of head shocks in
helmeted sport practice



Relatore

Prof. Danilo Demarchi

Candidato

Matteo Di Giacobbe

s257166

Anno Accademico 2019/2020

Abstract

Purpose: A pilot study was performed to design an electronic monitoring system, to be mounted on the helmet of hockey players, capable of detecting the shocks suffered by the brain during the sport activity, and then analyzing the shocks with a finite element model of the head-brain system developed during the project.

Methods and approach: A preliminary bibliographic research was carried out on the clinical point of view of the Traumatic Brain Injuries, on the statistics of head injuries in helmeted sports, on the monitoring devices of this type of injuries existing on the market and on the FEM models of the head-brain system developed over the years. Subsequently, the hardware of the monitoring system was designed, including a microcontroller, an accelerometer, a gyroscope, a micro-SD card on which to write data, and it was programmed in the Arduino IDE. An application for Android smartphones was developed in the MIT AppInventor environment, to report the shock, via Bluetooth Low Energy connection, to the medical staff on the sidelines. A finite element model of the head, composed by the skull, brain, cerebellum, corpus callosum, Cerebrospinal fluid, brainstem, neck and helmet, was designed on Abaqus using geometric simplifications, without a biomedical imaging basis. Multiphysics simulations were carried out on the finite element model of the head, by forcing as input the shock data acquired by the monitoring system, simulating an analysis process post-game. **Findings:** The monitoring system showed competitive performance in terms of measurement range and resolution compared to other devices on the market, and an improvement due to its consideration of the two types of acceleration. Concerning the costs, only the values of the individual components can be assessed, the device can be miniaturized and implanted on a real helmet in the future and the cost of packaging can be estimated. The computational time of the multiphysics simulation does not allow a real-time analysis of shocks. The finite element model, however, has showed good performances in computational terms (even less than 1 hour of calculations), thanks to the geometric simplifications with which it was designed. The clinical consequence of the impact can be assessed at the end of the match by the team's medical-scientific staff by analyzing the internal parameters of the brain, first of all the Intracranial Pressure. The system is presented as a support tool for diagnostics and clinical analysis, not a replacement for them.

Keywords: Head trauma, Head trauma in helmeted sports, Monitoring system, Bluetooth, Finite element analysis, Brain modelling, Finite element head model

Acknowledgements

I express my appreciation to those who have helped me along the way of five years at Politecnico di Torino and in the construction of my thesis.

First of all, I would like to thank my supervisors at HES-SO of Fribourg, Pr. Marco Mazza and Eng. Martin Kuhlmann for offering me the opportunity to work on this stimulating topic. Their excellent guidance and extraordinary confidence were a significant contribution to this project.

I especially thank Martin and all iPrint team of Marly for warmly welcoming me to the research center, even if in a difficult period like that of the Covid-19 pandemic.

I would like to thank my family, my parents Fabrizio and Michelina, my little sister Sabina, Antonietta, Guido, Maria Teresa, Piera and my four grandparents, Italo, Giuliana, Giovanni and Assunta, for their curiosity, enthusiasm and support during my studies.

Finally, I would also like to take the time to thank my childhood friends, my University classmates and my “roommates” of Collegio Einaudi, who represented a second family for me, with whom I grew up as a man and to whom I hope I have left good memories and wise teachings for the younger freshmen.

Summary

Introduction to the reading of the paper	1
Bibliographic research: head trauma	1
Incidence of head trauma in sport	1
Statistics from studies on collegiate athletes of helmeted sports	2
Injury Prediction	3
Bibliographic research: state of the art of monitoring systems.....	4
Devices on the market.....	5
Head Impact Telemetry System (HITs)	6
Shockbox HD.....	9
MC10 - Reebok Checklight.....	11
ShockAccel system.....	14
Description of components	14
SparkFun Pro nRF52840 Mini	14
Accelerometer ADXL372.....	15
6-axis Motion Tracking Bosch BMI088	16
SD card.....	17
Introduction to Bluetooth and Bluetooth Low Energy	17
Connection.....	19
Applications	19
Fundamental components	20
Profiles, Services and Features	21
Services and Features.....	21
Advertising, Connection and exchange of data.....	21
Comparison with other similar technologies	22
Why choose Bluetooth Low Energy.....	23
Hardware	23
Software	25
Arduino IDE.....	25
MIT AppInventor	25
Programming	25
Arduino IDE.....	25
MIT AppInventor	34
Results	37
Consumptions.....	38
ADXL372	38

BMI088	38
Carte SD	39
SparkFun Pro nRF52840 Mini	39
Battery sizing	40
Costs	40
Comparisons	41
Theory of the Finite Element Method	42
Multiphysics Simulation	46
Bibliographic research: Finite Element Models of the Head-Brain system.....	46
EEVCWG17 head form FE model	47
Wayne State University FEHM.....	48
Human body head model (HBM).....	49
Image-based FE Method (Simpleware's model)	51
TNO Head Model	53
THUMS FE Model.....	54
The ULP FEM Human Head.....	55
Politecnico di Torino	57
Harvard Medical School.....	59
Dublin College (UCDBTM).....	61
ShockAccel FEHM	63
Part Module	64
Left_Hemisphere	65
Corpus_Callosum	65
Cerebellum	66
CSF	67
Column.....	68
Neck	68
Skull	69
Padding	70
Helmet	71
Property Module	72
Viscoelastic Behavior: Maxwell Model in Abaqus Prony Series	75
SPH Theory	78
Assembly Module	81
Mesh Module	83
Load Module.....	84

Interaction Module.....	87
Step module	88
Job Module	90
Visualization Module	91
Limits of the model and possible improvements	95
Comparisons	96
Conclusions.....	97
References	98

Introduction to the reading of the paper

This paper describes the final project experience of the student of Politecnico di Torino Matteo Di Giacobbe at the HES-SO in Fribourg, during an Erasmus experience of five months (1st March 2020 – 31st July 2020). The topic proposed by the responsible of the project, Pr. Marco Mazza, was the design of a monitoring system for the shocks suffered by the heads of hockey players and a Finite Element Model of the head-brain system to analyze the same shocks. The paper is therefore composed of two parts: the first, of electronic nature, describes the design and programming of the device capable of detecting shocks, the second, of computational biomechanical nature, describes the model of the human head drawn on Abaqus. Both parts are introduced by a bibliographic research on the devices already on the market, for the first, and on the models already designed by other research centers, for the second.

Bibliographic research: head trauma

Traumatic Brain Injuries (TBI) are the injuries that involves the brain after a mechanical shock (impact, blast loading, sudden acceleration or deceleration); they happen when the brain tissue tolerance level is exceeded [1]. They are a cause of hospitalization, a burden on the healthy system, and, in certain cases, can lead to the death of the patient. After the shock, neurons suffer a metabolic crisis: the mechanical deformation caused a perturbation of ion flow across their cellular membranes, resulting in a decreased adenosine triphosphate (ATP) availability, necessary to restore the biochemical and energetic balance inside the neuron [2].

TBI can be classified as mild to severe. The concussion, or mild TBI (m-TBI), is defined as the neurological disfunction of the brain, resulting of an impact directly or indirectly applied on the brain. This shock results in the sudden linear and/or rotational acceleration of the brain. Symptoms associated with concussion include [3]: nausea, vomiting, nausea, balance problems, dizziness, trouble sleeping, difficulty remembering and concentrating, sensitivity to light or noise and blurred vision. These modifications of neurological functions can be short in time, and spontaneously return to a physiological condition, or permanents.

Another common type of TBI is the Diffuse Axonal Injury (DAI), that is the result of an excessive stretching, beyond its functional limit, of the axon, damaging the communication between neurons [4].

[Incidence of head trauma in sport](#)

During the practice of sport, concussion is an inherent risk, caused by the falling or by the contact with other players or surfaces. A research of the 2012 [5] has estimated that, in USA, among the 23.6 million youth athletes, 19% joined collision sports and 57% participated in contact sports. Moreover, the number of sports-related TBIs in the USA between 1.6 and 3.8 million per year, including cases ignored by the health service. [6]. Deepening the statistics of the traumas, head concussive injuries contribute for 8.9% of all high school and 5.8% of all collegiate sport injuries.

A 2013 epidemiological study of the incidence of sport-related TBI reported that 49.2% of the total injuries were concussions, and the most common mechanism (38.1%) of sport-related TBI was caused by the contact between players [7].

In 2017 in USA, the different types of injuries related to the shocks suffered by the head during youth sports were investigated in more detail, obtaining the pie chart in Figure 1.

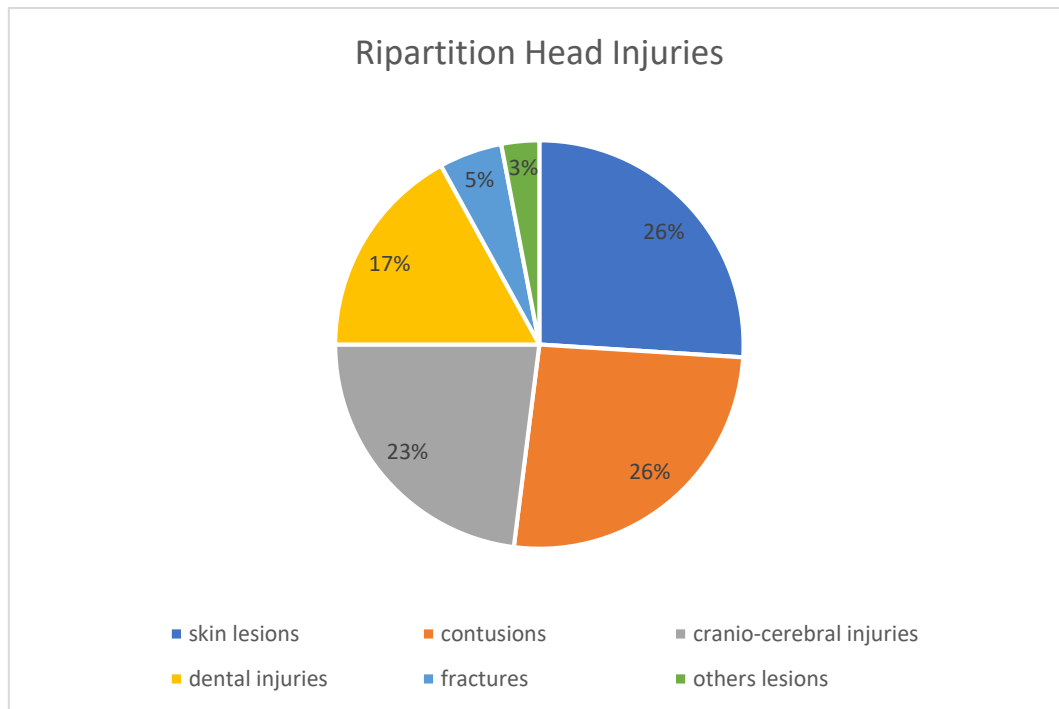


Figure 1. Statistics of Head Injuries during youth sports

Motivated by the sport-related TBI (Traumatic Brain Injury) statistics, the scientific community, in collaboration with sport industries, has shown an interest in reducing the frequency and severity of head impacts experienced by athletes participating in contact sports. The primary protective system used in American football, ice hockey and lacrosse is the helmet, whose evolution over the time has been drastic, concerning the structure, materials and the sensorization.

Sporting helmets are designed to prevent, above all, the skull fracture, a damage more severe than the concussion. Indeed, no helmet can completely eliminate the possibility of a concussion, but these devices are still effective in reducing direct energy transfer inside the impact: the modern idea is that an intermediate layer, called padding, made of a material with a certain behavior, between the outer layer and the skull, can increase the impact duration and, consequently, reduces impact forces. The design of a new model of helmet is not the goal of this project.

Statistics from studies on collegiate athletes of helmeted sports

Between 2007 and 2009, 335 football players were equipped with helmets instrumented by HIT system (a system of monitoring that will be explained in detail in the next chapters) to participate in American collegiate championships [16]. The most significant statistics extracted from this study were:

- total of 300977 head impacts;
- 5.56 concussions per 1000 games played;
- 16.3 impacts per game per player;
- injury incidence rate of 0.341 concussions per 1000 impacts
- head linear accelerations from 24 g to 176 g (for a concussive event)
- head rotational accelerations from 1506 rad/s² to 14431 rad/s² (for a concussive event).

Between 2005 and 2007, the helmets of 12 collegiate hockey players were instrumented by HIT system [60]. The most significant statistics extracted were:

- total of 4,393 head impacts;
- five impacts per player per day;
- The mean of the top 1%, 2%, and 5% of all peak linear accelerations were 104.9 g, 90.0 g, and 62.4 g, respectively;
- 33.2% of all impacts were to the back of the head, 32.6% were to the side, 14.3% were to the forehead, 12.2% were to the facemask, and 7.8% were to the top of the head.

In both studies, impacts exceeding the 10g threshold were considered impacts.

The statistics reported in this chapter are quite scattered, but they help to understand the range of values encountered when dealing with head shock in sports.

Injury Prediction

Studies have shown [8] that linear acceleration is the main actor in the compression of cerebral tissue, whereas rotational acceleration is involved in the shearing of the neurons, suggesting that these two indicators have an equal role in the injury prediction.

The relation between the magnitude of the impact, represented only by linear acceleration, and its duration was firstly studied by the Wayne State University, which developed the Wayne State Tolerance Curve (WSTC), in the first years of 70s [9], that is shown in Figure 2. The idea was to define an injury threshold dependent on the time for which the driver and/or the passenger (WSTC was developed as support of the automotive crash tests conducted by car manufacturers) is subjected to the shock.

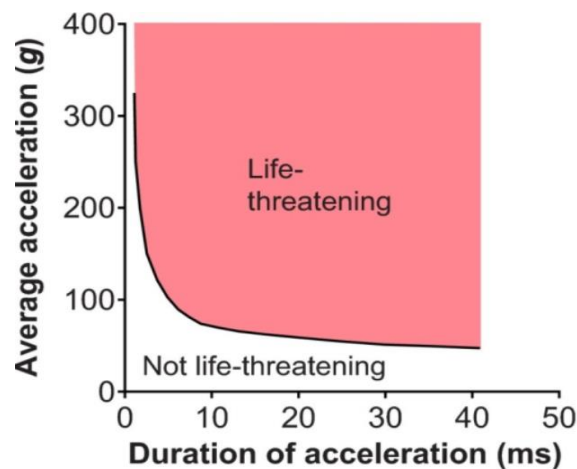


Figure 2. Wayne State Tolerance Curve

The obvious concept behind this curve is that human heads can tolerate strong impacts (larger acceleration magnitudes) for short periods, while for longer periods it can absorb just smaller impacts.

The problem is that WTCS defined an injury threshold based on skull fracture onset, not on concussion, thus a less conservative criterium. Moreover, it didn't contain any rotational information, and it was developed starting from data from the automotive industries of the 1960s, that were less numerous and less accurate than those available today [10].

Nowadays, the parameter most used in order to quantify the severity of a shock on the head because of a collision is the Head Injury Criterion (HIC), evaluated as:

$$HIC = (t_2 - t_1) \left[\frac{1}{t_2 - t_1} \int_{t_1}^{t_2} a(t) dt \right]^{5/2} \quad (1)$$

where t_1 and t_2 represent the initial and final instants (in seconds) of the interval during which $a(t)$, the acceleration measured in g, attains a maximum value. Scientists [11] limited the maximum time duration of HIC, $t_2 - t_1$ to a specific value between 3 and 36 ms, usually 15 ms.

The HIC was proposed by Versace [12] as the improvement of the older value Gadd Severity Index (GSI):

$$GSI = \int_0^T a(t)^{5/2} dt \quad (2)$$

formulated by Gadd in 1961, where T is the duration of the pulse. GSI was less ‘temporally focused’.

The HIC has been strongly criticized and discussed. This parameter is limited by the consideration of only linear accelerations. Moreover, literature suggests the use of HIC only for the cases of impacts against rigid surfaces, which do not cover all the possibilities of impact during sporting activity.

On the contrary, there are several studies [13] conducted on cerebral lesions that showed the influence of other mechanical aspects, as the angular accelerations, the location of the impact on the skull and the conditions of the contact; the latter are related to the stress and pressure distributions inside the cranium.

A new method to analyze and predict different type of head injuries is the design of a finite element human head model, validated by experimental results, in which other parameters can be obtained in a non-invasive way, such as the Intracranial Pressure (ICP), Max Principal Strain (MPS), Brain Von Mises Stress, Global Strain Energy of the Brain/Skull interface, all non-inertial parameters, which would not be measurable otherwise, nor real-time during sporting activity.

Bibliographic research: state of the art of monitoring systems

The head-impact monitoring systems have the aim to collect data real-time, providing a measure of head-impact exposure: they are diagnostic support tools only in terms of subsequent system analysis, they can’t be used to diagnose concussions [14].

One of the goals of the monitoring of shock on the head during sport practice is the possibility to recall an injured athlete from the field to the bench to reduce the risk of secondary injury, when they are still vulnerable to the cumulative effect of TBIs [15]. Moreover, the medical staff can proceed with specific cognitive tests to check the cerebral condition. Another possible application, derived from monitoring systems, can be the analysis of the comparison between the results of the cognitive tests before and after a detected shock.

One of the most discussed topics in the bibliography of monitoring systems is the choice of the variables to be monitored, whether of inertial, dynamic or pressure nature, or electrophysiological, etc... In the HIT system (taken as a reference in this paper, it will be deepened in a next chapter) linear acceleration has directly measured by an array of 6 accelerometers [16], while peak rotational acceleration has been calculated through an algorithm (owned by the company), which estimates the position of the head CG, resulting in error propagation.

A possible improvement could be to use directly a 6-axis Inertial Measurement Unit (IMU): it would measure simultaneously 3 linear accelerations and 3 angular velocities, and then the microcontroller could evaluate,

by time derivation, the 3 angular accelerations. This would result in 9 'impact features' with savings in terms of footprint, consumption, computational cost and accuracy.

The search for a sensor of this kind was a target of this project. Unfortunately, there is no 6-axis IMU with sufficient ranges ($\geq 200g$ and $\geq 2000^\circ/s$), to avoid the risk of saturation after the shock, in the market. Probably, this is due to lack of demand for the sensor industries. These type of IMU sensors with high ranges of measure are demanded by car manufacturers for crash test, experimental field in which there is no need of compactness of the acquisition system: the measures can be spread into varies sensors implanted in many places inside the car or on the dummy. Instead of implanting 1 6-axis IMU, these types of industries can implant many 1-axis IMU without problem of space.

Moreover, other studies showed a big difference between measuring a shock with a sensor installed in the helmet and a sensor implanted in the subject's head [61]. The differences noted concerned the amplitudes of the kinematic variables and the temporal dynamics, such as the time to peak.

There are systems on the market that monitor shocks with devices attached to the athlete's skin, usually under the ear, eliminating the problem of kinematic inconsistency between the helmet and the head: one of these systems will be analyzed in a chapter. The target of this project, however, was the design of a monitoring system to be implanted in the helmet of a hockey player, so this issue was not investigated.

In conclusion, it is important to assess what a monitoring system is not (and that no device on the market is):

- not a tool to diagnose concussions;
- not a tool to prevent concussions;
- not a tool to treat concussions;
- not an unquestionable judge to remove the player from the field.

Devices on the market

In this chapter the various systems in the market will be discussed in detail, their theory of operation, hardware setting, component specifications and accuracy. Moreover, the packaging and costs associated will be analyzed. Figure 3 provides an overview of images of the different devices on the market.

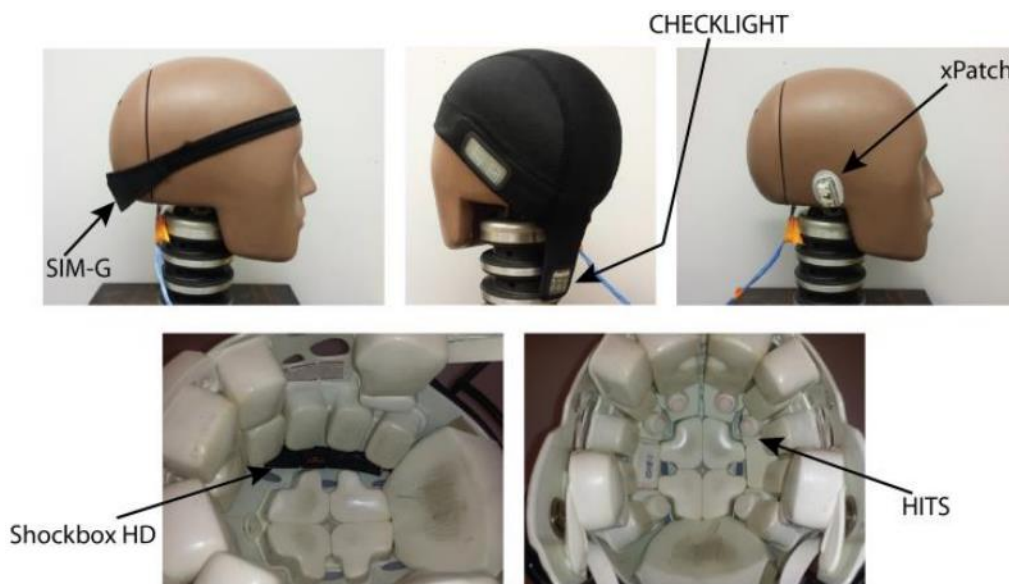


Figure 3. The first three are the devices wearable directly on the head (head-mounted): SIM-G, CHECKLIGHT and xPatch. The remaining two are the implantable systems in the helmet (helmet-mounted): Shockbox HD and HITS.

Head Impact Telemetry System (HITs)

The Head Impact Telemetry (HIT) System was marketed by Simbex as “the first and only commercially available system that can measure head accelerations (impacts) in real-time during [football] games and practices” [17]. In 2003, Riddell [18] implanted the HIT System with the line of production of helmets, calling it Riddell Sideline Response System, focusing high school football teams as target.

The system is composed by 3 major components (Figure 4): (1) individual sensor ‘pods’, fixed to individual players’ helmets inside padding layers, (2) a base station that wirelessly (915MHz ISM radio band) receives information of the detected head impacts from each sensor ‘pod’, displaying relevant information on a laptop screen and (3) a proprietary algorithm, the Head Impact Telemetry Severity Profile (HIT-SP), developed by the producer (Simbex), able to associate the single impact with a diagnosis of concussion, with a certain clinical reliability [19].



Figure 4. Riddell base station unit (left). Riddell helmet fitted with HIT system (right). The red circles mark the pods.

The system can support connection with an entire team on the pitch, the laptop displays a list of connected players, with health parameters associated with each pod. Sensors acquire linear and rotational acceleration, from which the HIT-SP algorithm deems a hit ‘significant’ or not. All significant hits are stored in an exportable database, giving the possibility of a later analysis of the clinical situation of the athlete.

Moreover, there is a processing algorithm responsible of the removal certain detected impacts from the data set. The removal occurs:

- if the magnitude of linear acceleration is less than 10g (threshold method);
- if the profile of the acceleration pulse does not correspond to the characteristics of an impact. The aim is to detect only shock on the head events, and not also the action, for example, of placing the helmet on the bench.

The hardware system includes 6 linear 1-axis accelerometers (ADXL193) [20] with high range ($\pm 250G$) and low power consumption (1.5mA). The data from the sensors are sampled by the TI MSP430F148 [21], an ultra-low power microcontroller, with a sample rate of 400Hz: it is the executor of the calculations necessary to detect the impact.

The microcontroller is equipped with Microchip 24FC512 EEPROM, which contains the HIT-SP.

The embedded system implanted in the helmet and the base station are equipped with a Semtech XE1203F [22] transceiver, that transmits wirelessly the data, that can be reviewed by the athletic staff.

The system includes a 1000mAh Ni-MH battery. The expected and declared duration of the device is 15 hours, which corresponds to the sum of the consumption of the individual components, as indicated in Table 1.

Major component	Current consumption (mA)		Operating Voltage (V)	Power Consumption (mW)
	Typical	Maximum		
Texas Instruments MSP430F148	0.42	0.56	3.30	1.39
Semtech XE1203F (Transmit mode)	62.00	75.00	3.30	204.60
Microchip 24FC512 (Write mode)	5.00	5.00	5.50	27.50
Quantum QT310	0.60	1.50	5.00	3.00
Analog Devices ADXL193 (x6)	1.50	2.00	5.00	45.00
Total Power Consumption:				281.49
Voltage supplied by battery (V):				3.70
Current drawn from battery (mA):				76.08
Battery life expected (hours):				13.14

Table 1. Power consumption of the components inside the HIT System

The Simbex HIT System is only available for high schools, colleges and centers of research, not for individual consumers, with a cost between \$1500 - \$2000 per helmet (it depends on the quantities purchased) [23]. Thus, this is not a device available for all type of high schools, requiring an expense of \$45000 - \$60000, considering a team of 30 players.

Table 2 lists the actual costs of the single components, estimating the real value of the system around 77\$.

Major component	Cost
Texas Instruments MSP430F148	\$4.87
Semtech XE1203F (Transmit mode)	\$4.90
Microchip 24FC512 (Write mode)	\$1.41
Quantum QT310	\$2.82
Analog Devices ADXL193 (x6)	\$48.12
Misc. passive components	\$5.00
PCB fabrication costs (estimated)	\$10.00
Total electronic costs:	\$77.12

Table 2. Costs of the individual components of the HIT System

The packaging is voluminous (Figure 5). Each pod contains an ADXL193 incorporated in a molded foam, fixed to a half ring in the padding layer of the helmet. Some wires, soldered directly to the boards, connect each accelerometer to the main PCB, that contains the microcontroller and the transceiver. The main PCB is located on the left side, in a rectangular foam cut-out, while the battery on the right side.



Figure 5. Packaging of HIT System

Simbex HIT System measures linear accelerations and infers rotational accelerations, through the algorithm HIT-SP: this inferred information is obviously cause of inaccuracies. Accuracy studies have been conducted and reported in literature [24], comparing data from HIT System with that acquired from Hybrid III dummy (Figure 6).



Figure 6. Hybrid III dummy

The Hybrid III is an anthropometric dummy, representing a 50th percentile male. It is the gold standard for the frontal crash tests performed by automotive manufacturers. It was originally developed by General Motors in the 70s. Its main qualities are excellent biofidelity and instrumentation capability.

The comparison of experimental data from HIT and from Hybrid III showed that over 60% of the impacts recorded by the HIT System had an error greater than 15%.

The HIT System has always played a key role in many long-term mTBI investigations for many years, it was the most used monitoring system in the research field, but it always showed limits of hardware and accuracy.

Shockbox HD

Shockbox was marketed by Impakt Protective in 2011. As pronounced by Danny Crossman's (Impakt Protective CEO) at AFCA 2012 (ceremony of College Football All-America), Shockbox "is the world's first small affordable wireless helmet impact sensor". It is a long-range wireless sensor, it sends a Bluetooth signal 350 feet from helmet, taken an impact, to the sideline to an iPhone or Android or a BlackBerry device, where impact information is displayed. The device is implanted along the sagittal plane of a helmet with the aid of Velcro strips or adhesive (Figure 7). The two main markets interested by this device are the youth categories of ice hockey and football.



Figure 7. Shockbox device, alone (left) and mounted on a hockey helmet (right)

One single smartphone, with Shockbox app installed, can support a connection with the entire team. Of each player it is possible to insert the metadata (name, age, team information, impact history recorded). Moreover, the app notified in real time if any hits exceed the user-defined threshold, indicating the player name, the instant, and approximate magnitude and direction of the hit. Once the detection is presented, the user has the possibility to save it, or discard if it was a false positive.

The sensitive part of the device is not based on an inertial sensor, but on the use of 4 binary electromechanical switches [25]. The data of force collected by the switches are processed by a microprocessor, that could be programmed through a micro-USB connector, and later are sent through a Bluetooth module. Impakt Protective has chosen to use the Bluegiga WT11i, a Bluetooth module of Class 1. The PCB layout of Shockbox is showed in Figure 8.

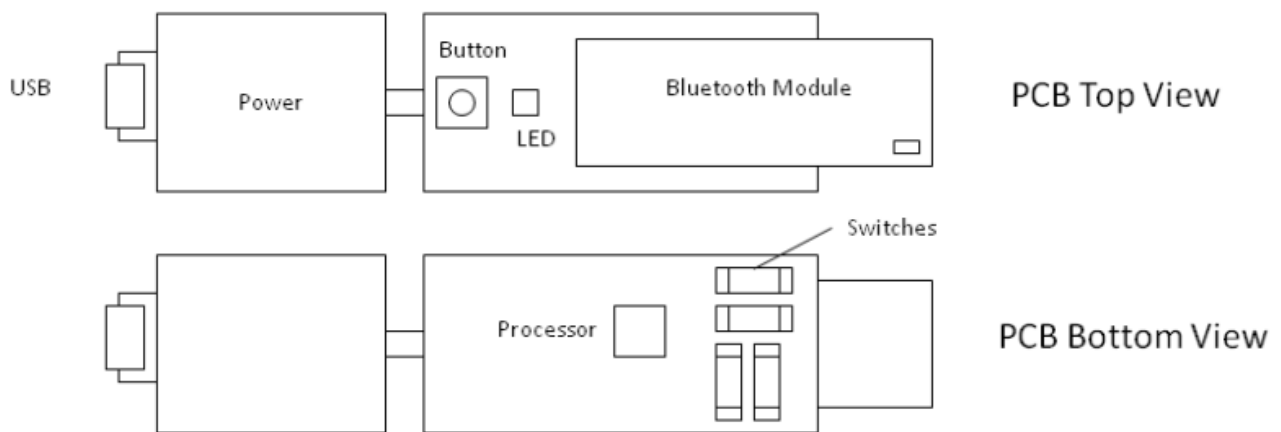


Figure 8. Shockbox PCB layout

The software inside the processor converts data of the impact to a visual color-coded alert (Figure 9), which is performed by the LED: a color indicates a hit belonging to a range of g. The code is elementary and easy to understand: green signal indicates a non-dangerous impact, yellow that the hit was within a certain range (50g – 90g), orange indicates a larger and more dangerous impact (>90g). Thus, the algorithm is simply based on a threshold system.

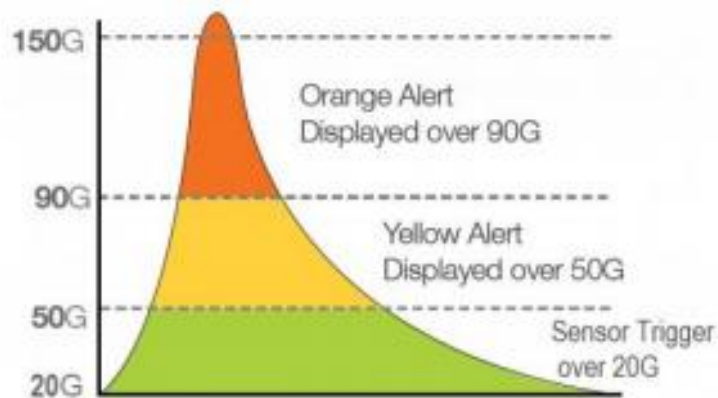


Figure 9. Color code of the impacts in the Shockbox algorithm

The user, informed about the shock and with a quantifiable measure, can start the clinical protocols for the rescue of a concussion, if the signal is yellow or orange.

According to the manufacturer's estimates, the device guarantees 500 hours of battery life between one charge and the next: this is allowed by the low power consumption of the electromechanical force switches, and by low current required by the Bluegiga WT11i (50 μ A) [26] when it is inactivated. When the Bluetooth module transmits data requires a current of 180mA. Obviously, the duration of device battery life is conditioned by the number of hit that the players get.

The Shockbox's price on the market is \$149.99. The most expensive component is the Bluegiga WT11i Bluetooth module, which costs \$21.11. With an estimate of the value of the other components and considering an assembly cost of \$10, the manufacturing cost is presumed to be around \$40. The companion app downloadable for free.

The PCB is made of flexible polyamide (FPCB). Later the FPCB is immersed in a container with melted rubber, thus forming the exterior casing of the Shockbox. The device is not a single block, but is segmented into

various pieces, thus allowing a relative flexion, to better adapt to the curvatures of the different helmet sizes and models.

The accuracy of the device is severely limited by the type of sensor used, the binary electromechanical force switches. These sensors are set to detect if a hit exceeds a fixed acceleration threshold, with a low resolution, allowing the device to classify the shock as 'mild' (green), 'moderate' (yellow) or 'severe' (orange). The microcontroller is not equipped with any onboard memory or EEPROM, not allowing the storage of data if the device is out of the spatial range.

MC10 - Reebok Checklight

In 2013 MC10, in collaboration with Reebok, launched the head impact indicator MC10 – Reebok Checklight. "Keep your Head in the Game with CHECKLIGHT!", this was Reebok's slogan at the product launch. The goal of the featuring between the two companies was to design a Sport Impact Indicator, a breakthrough technology to maximize performance while minimizing risk of injury to athletes of all skill levels. A device with high performance, comfortable and conformal, without distraction from the play, able to give objective measurement of force of impact. It represents an extra set of eyes for coaches, trainers and parents, and put athletes on a pathway to assessment. According to a study by MC10 the first reason of injuries not diagnosed is lack of symptom reporting, and Checklight gives the opportunity for real-time symptom reporting [27].

The system is composed of two parts: a sensor, designed by mc10, and skullcap produced by Reebok (Figure 10). The player needs to wear the breathable fabric skullcap, and later can insert the Checklight into the pocket.



Figure 10. MC10-Reebok Checklight with the breathable fabric skull cap

The PCB of the hardware system is showed in Figure 11. The device uses a system of LEDs (three, red, yellow and green) to indicate the severity and the count of hits. The assessment of the severity (moderate, less severe or more severe) of the shock is based on a proprietary algorithm, where the most significant value is the HIC.

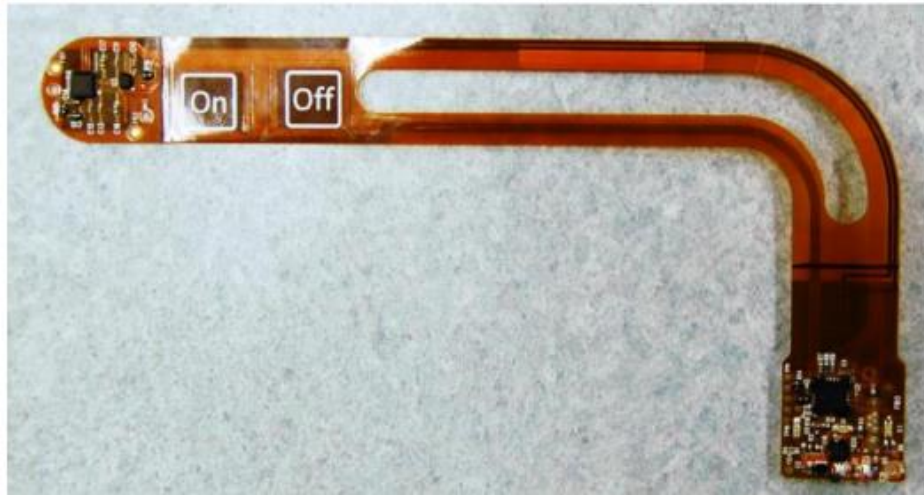


Figure 11. MC10-Reebok Checklight PCB

The alert codes of the LEDs are listed below:

- for every 100 less severe hits, the green LED flashes once;
- the yellow LED starts to flash if the device detects a moderate hit;
- the red LED starts to flash if the device detects a more severe hit;
- when the red LED emits steady light, the device is recommending the staff to assess the condition of the player, to understand if it has occurred a concussion. This notification (the most important) is dismissible only powering the cycle of the device.

Only during the charging operation (charging mode) it is possible to see displayed the total impact counts.

The system is composed by a microcontroller (PIC 24FJ64), a 3axial digital gyroscope (ST Microelectronics L3G4200D MEMS) and a 3 axial digital accelerometer (Bosch BMA250 MEMS). The range of the gyroscope is $\pm 2000^\circ/\text{s}$, the range of the accelerometer is $\pm 16\text{g}$. The data are stored in the flash memory of the PIC 24FJ64. The device doesn't contain an external EEPROM. The system is powered by an 80mAh battery at 3.7V, which guarantees 6 hours of works and requires 4 hours of charging. The currents consumed by each single component of the system are summarized in Table 3.

Component	Current consumption
Microchip PIC24FJ64	25mA
Linear Tech. LTC4080	0,3mA
Bosch BMA250	0,2mA
ST L3G4200D	6,1mA

Table 3. Consumption of the MC10-Reebok Checklight

The packaging of this device must pander to the flexion of the head during physical activity: in this way, a base of polyamide contains a flexible PCB and all the components are soldered into it. To avoid an excessive bending of the PCB, the sections of the microcontroller and of the sensors are stiffened, and epoxy reinforced.

The MC10-Reebok Checklight is sold at price of \$149.99, while the estimated value, considering the price of the individual components, is about \$33.10, as showed in Table 4. The device is open for sale to athletic team staff and to not agonistic purposes.

Major component	Cost
Microchip PIC24FJ64	\$3.89
Bosch BMA250	\$2.78
ST Microelectronics LG34200D	\$3.20
Linear Tech. LTC4080	\$2.24
80mAh Li-Po battery pack	\$5.99
Misc. passive components	\$5.00
PCB fabrication costs (estimated)	\$10.00
Total electronic costs:	\$33.10

Table 4. Component costs of the MC10-Reebok Checklight

The most noticeable limitation is the extremely low range accelerometer used: 16g represents a low level of acceleration during a football and hockey activities, where the impacts exceed regularly 30g, till reaching sometimes 200g.

The proprietary algorithm used for the alert signal is based above all on the value of HIC, that was established not completely indicative for an assessment of the concussion, while the device tries to classify an impact as mild, moderate or severe.

Some lab tests on the device showed how a simple flick with a finger caused it to flash red, while dropping the MC10-Reebok Checklight from varying heights was barely registered.

The advantages of the device, according to the manufacturers, are summarized below:

- Reliable impact data, thanks to the tight coupling to head;
- Easy to use for athlete;
- Does not interfere with game play;
- Multi-sport compatible, also for non-helmeted sports;
- Easy to see and interpret data (LED system);
- Low cost.

ShockAccel system

After having done a bibliographic research on the monitoring devices on the market, and after having analyzed the advantageous and less advantageous characteristics of each, the thesis path involved the design of a new monitoring system based on 3 main elements: the Sparkfun board, the ADXL372 accelerometer and shock signal transmission via Bluetooth Low Energy. The aim of the project was to create a device with those elements that would try to mimic the existing devices in their strengths and to follow clinical developments regarding head injuries.

The final system, called ShockAccel system, consisted of:

- the SparkFun Pro nRF52840 Mini, a development board programmable with Arduino and capable of connecting via Bluetooth with other devices;
- the ADXL372, a 3-axial accelerometer;
- the BMI088, a 6-axial IMU, but only the 3 gyroscopic axes were considered;
- the Breakout Board which supported the micro SD card;
- the Android application with which medical staff can monitor athletes' head shocks in real-time.

Description of components

SparkFun Pro nRF52840 Mini

The SparkFun Pro nRF52840 Mini (Figure 12) is a development board for Nordic Semiconductor's nRF52840. The device incorporates the ARM Cortex-M4 central processing unit (CPU) and a 2.4GHz Bluetooth radio.

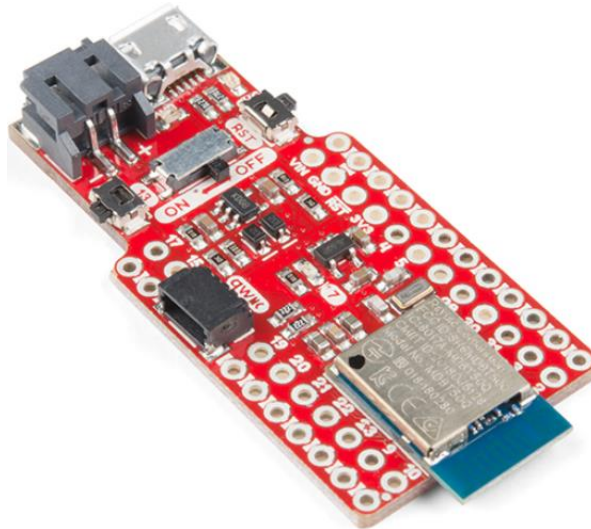


Figure 12. SparkFun Pro nRF52840 Mini

Although the very small package, the most important input/output (I/O) pins are available on the card, without limiting the possibilities of project.

It has a USB interface which can be used to program, power and communicate with the chip. A LiPo battery charger, a qwiic connector, an ON / OFF switch, two buttons and a user LED are also included.

The device is pre-programmed with a USB bootloader. There are several ways to program this type of CPU, using either Arduino, Circuit Python or C using Nordic's nRF5 SDK.

A very practical feature of the Cortex M4 is the possibility of pin multiplexing: almost any pin can support any device. To change the function of the spindle, it is necessary change the definition of the spindle in the header file "sparkfun_nrf52840.h" in the folder of Arduino.

The following is a short list of the technical characteristics of the device:

- ARM Cortex-M4 CPU with floating point unit (FPU)
 - 1MB internal Flash
 - 256kB internal RAM
- Integrated 2.4GHz radio with support for:
 - Bluetooth Low Energy (BLE), with peripheral and/or central BLE device support
 - Bluetooth 5, for the Bluetooth mesh networking
 - Adaptive Network Topology (ANT), with the possibility to turn the device into a heart-rate or exercise monitor
 - Nordic's proprietary RF protocol, to communicate, securely, with other Nordic devices.
- I/O peripheral
 - USB
 - Universal Asynchronous Receiver-Transmitter (UART), to convert bit streams of data from a parallel format to an asynchronous serial format or vice versa
 - I²C
 - SPI
 - Analog-to-digital converters (ADC), there are 8 pins on the nRF52840 Mini Breakout that support analog inputs
 - PWM -- Timer support on any pin means PWM support for driving LEDs or servo motors.
 - Real-time clock (RTC) -- Keep close track of seconds and milliseconds, also supports timed deep-sleep features.
- Peripheral-multiplexing -- (Nearly) any pin can support any of the above features.

Accelerometer ADXL372

The ADXL372 (Figure 13) is 3-axis Micro-Electro-Mechanical Systems (MEMS) accelerometer, with ± 200 g measurement range, produced by Analog Device.

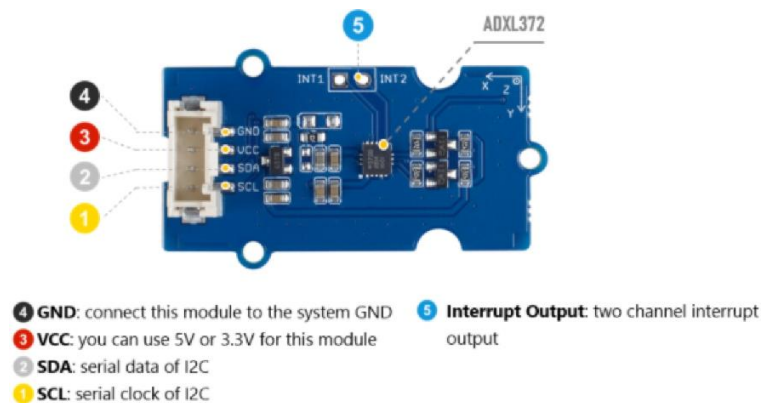


Figure 13. ADXL372

As indicated in its datasheet [62], it has many features to enable shock detection, above all the high range of measure, which allows the device not to saturate following huge impacts.

ADXL372 is a 12-bit device, with a resolution of 100 mg/LSB scale factor.

The user can configure its registers via the serial peripheral interface (SPI) or limited Inter Integrated Circuit (I2C) protocol.

The sampling frequency (ODR) and the bandwidth can be selected, accessing and writing into the registers of the accelerometer. The possible combinations of choice of these two parameters are shown in Table 5.

ODR (Hz)	Bandwidth (Hz)
400	200
800	400
1600	800
3200	1600
6400	3200

Table 5. ODR and Bandwidth settable in ADXL372

The ADXL372 is defined as an ultralow power MEMS, in fact it consumes only 22 μ A in measurement mode, at a 3200 Hz output data rate (ODR).

The connection between the Sparkfun and ADXL372 was managed through the I2C protocol.

6-axis Motion Tracking Bosch BMI088

The BMI088 (Figure 14) is 6-axis IMU, composed by a triaxial 16-bit gyroscope plus a triaxial 16-bit accelerometer, produced by Bosch. The internal gyroscope has a range of measure of ± 2000 $^{\circ}$ /s, while the internal accelerometer has a range of measure of ± 24 g. The main applications of this device are robotics and drone control, especially for its capability to suppress vibrations above few Hz, that can represent an issue in these fields.

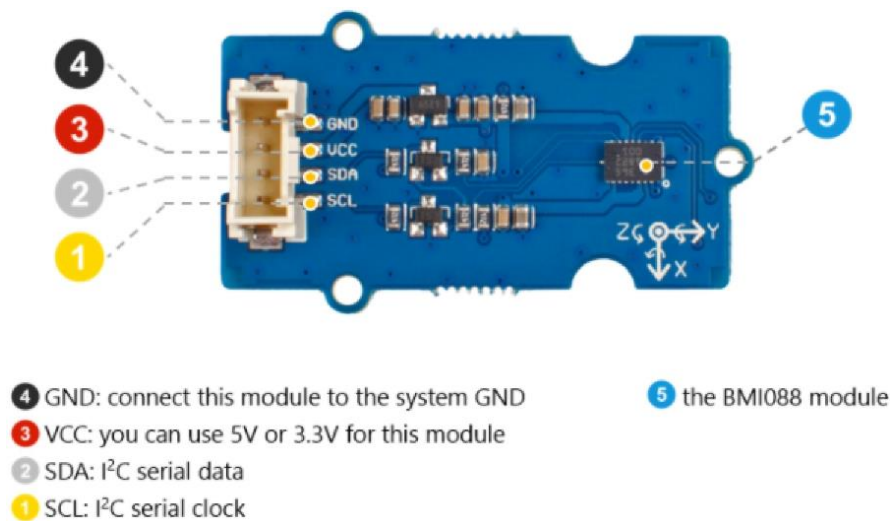


Figure 14. BMI088

The sampling frequency of the gyroscope can be selected, accessing and writing into the registers, between 100Hz / 200Hz / 400Hz / 1000Hz / 2000Hz, and the bandwidth can be selected between 12Hz / 23Hz / 47Hz / 116Hz / 230Hz.

The sampling frequency of the accelerometer can be selected, accessing and writing into the registers, between 12.5Hz / 25Hz / 50Hz / 100Hz / 200Hz / 400Hz / 800Hz / 1600Hz.

About consumptions, the accelerometer requires a total supply current in normal mode of 0.150 mA, while the gyroscope one of 5 mA, for a total of 5.15 mA.

The range of the accelerometer inside BMI088, lower than that of ADXL372 (± 200 g), is not suitable for the detection of huge impacts, such as those that occur on the head during the sport activities considered in this project. For this reason, data acquired from the accelerometer were not considered, keeping only those from ADXL.

Instead the gyroscope measures the rotational speed, from which the microcontroller can calculate the rotational acceleration by time derivation, using as the basis of the incremental ratio the inverse of the sampling rate. Later the microcontroller, at the user's discretion, can perform the conversion from degrees to radians, reference unit of measurement in the scientific literature.

The user can configure its registers via the serial peripheral interface (SPI) or limited Inter Integrated Circuit (I2C) protocol.

The connection between the Sparkfun and BMI088 was managed through the I2C protocol.

SD card

The data was stored on a micro Secure Digital card (micro SD), which has been interfaced to the Sparkfun through a Breakout Board (Figure 15), produced by Adafruit Industries. The micro SD used was produced by SanDisk and had a memory capacity of 16 Gb.

The Breakout Board has a 5V->3V regulator, which provides 150mA for cards that require more energy. Moreover, with 3V level shifting, it is easily usable with 5V or 3V devices.

It uses a proper level shifting chip, without resistors, guaranteeing a faster read and write access.

A yellow LED lights up when the user reads or writes on the SD card.

The connection between the Sparkfun and the Breakout Board, containing the micro SD card, was managed through the SPI protocol.



Figure 15. Breakout Board

Introduction to Bluetooth and Bluetooth Low Energy

In this chapter the Bluetooth protocol and the Bluetooth Low Energy will be described, the latter used for the realization of this project.

Bluetooth (logo in Figure 16) is a wireless networking technology that has been elevated to the industry standard for radio communication by the IEEE. The IEEE standardized the protocol with the IEEE 802.15.1 code.

It represents a cost-effective and with low power consumption standard for the communication between two or more devices placed at short distance. Nowadays many devices contain Bluetooth chips: mobile phones, printers, keyboards, microphones, computers, embedded systems...



Figure 16. Bluetooth Logo

In the telecommunications field, the frequency at which Bluetooth works is 2.4GHz of the ISM band. Precisely, data exchanges take place on frequencies between 2.402 GHz and 2.480 GHz. Various communication channels are established on them: 79 channels with width bandwidth of 1 MHz for the first versions or 40 with bandwidths of 2 MHz for version 4.0. The devices use these channels to send data using a technology called Frequency-Hopping Spread Spectrum. It allows to exchange data at a considerable speed by changing the transmission channel, according to a pseudo-random order shared between transmitter and receiver, with a frequency up to 1600 times per second.

Depending on the class of the device, a different maximum transmission power is allowed which consequently allows a certain range, as shown in Table 6:

Class	Maximum permitted power	Spatial Range
cl.1	100 mW	about 100 m
cl.2	2,5 mW	about 10 m
cl.3	1 mW	about 1 m
cl.4	0.5 mW	about 0.5 m

Table 6. Bluetooth Classes

Due to the low power, it is necessary that the devices that must communicate can see each other: any obstacle that blocks or reflects the radio waves transmitted drastically decreases the transmission range. Power transmission, antenna configuration, battery condition and sensitivity of the receiving device are Other factors affecting the distance at which two or more devices can communicate.

The Bluetooth protocol is based on packet transmission between a master and a slave. A master can be connected with up to 7 slaves, with aim to communicate with them and to create a network called 'piconet'. The period of the clock, shared between all devices, is of 312.5 ms. The transmission works in blocks called slots. A slot consists of 2 clock cycles: in the even slot the master transmits while the slave receives, vice versa in the odd slot the master receives while the slave transmits. Each pack can be 1, 3 or 5 slots long.

Depending on the version used, different transmission rates are achieved. Table 7 shows the data relating to the most common versions.

Bluetooth Version	Theoretical transfer rate
Bluetooth 1.2	0.721 Mbit/s
Bluetooth 2.0 Enhanced Data Rate	2.1 - 3 Mbit/s
Bluetooth 3.0 High Speed	24 Mbit/s
Bluetooth 4.0	24 Mbit/s

Table 7. Bluetooth Versions

The architecture of Bluetooth protocols is organized in stacks. The fundamental and universally supported ones are the Link Management Protocol, the Logical Link Control and Adaptation Protocol and the Service Discovery Protocol. The first is useful to start and monitor the radio link between the slaves; the second

allows to manage slaves working with different protocols; the third supports the master to read and analyze the services offered by the slaves. Moreover, there is another series of protocols that may not be shared by all devices.

Connection

Each Bluetooth device in 'discoverable mode' (i.e. capable of being detected by other devices) transmits, at regular intervals, packets containing:

- Name of the device
- Class of the device
- List of services offered
- Other information (for example brand or other characteristics of the device)

This operation is called advertising.

Other devices can search to detect devices in discoverable mode and obtain information about them by receiving advertising messages. Two different types of connection are defined:

- ACL: Asynchronous Connection-Less, connectionless and asynchronous service. It supports data traffic and provides a best-effort service.
- SCO: Synchronous Connection Oriented, connection-oriented synchronous service. Generally used for the transport of the voice.

Depending on the type of service requested, it may be necessary to use one or the other type of connection. Pairing operation may also be required: this is an operation that increases the security of the connection by allowing connection only to specific devices. In fact, user intervention is required to perform pairing. Once this operation has been performed, a bond is created between the two devices that will allow the connection to be made again without user intervention.

Bluetooth Low Energy (BLE), at first known as Bluetooth Smart (logo in Figure 17), is a technology produced by Bluetooth SIG group specifically for applications that require wireless transmission with even lower energy consumption than classic Bluetooth, but with a higher bitrate.

The project was conceived in 2001 and marketed in 2006 by Nokia under the name of Wibree. In 2007, the brand was then included within the Bluetooth specification. The integration with version 4.0 happened in early 2010. The first devices that implemented this feature came out in 2011. Today Bluetooth Low Energy is based on the Bluetooth 4.1 specification, released in December 2013.



Figure 17 Bluetooth Smart Logo

Applications

Some profiles have been defined, with a series of characteristics that the devices must have in order to be compatible and suitable for a certain application, including Biomedical application, Human interface device, sensors, alert systems and Sport.

Fundamental components

The first fundamental component of BLE is the Generic Access Profile (GAP). It deals with the management of the connection and the advertising phase, making the device visible to the outside world and determining which devices can or cannot interact with others.

The GAP defines one of two different roles that can be covered by the two connected devices:

- Peripheral: the device is visible to other devices (thus it does advertising) and can accept incoming connections (slaves). It is not, however, able to initiate a connection. This role is generally used for sensor devices.
- Central: the device is able to search for visible devices and initiate the connection (master). However, it cannot accept incoming connections. This role is generally played by smartphones, tablets or other devices that connect to sensors.

Two devices that play the same role are unable to connect to each other. Moreover, a Peripheral device can only connect to one Central device by one. However, it is possible to create a broadcast topology by sending the data directly in the advertising packet, visible to all other devices.

Figure 18 and Figure 19 show examples of the two possible topologies.

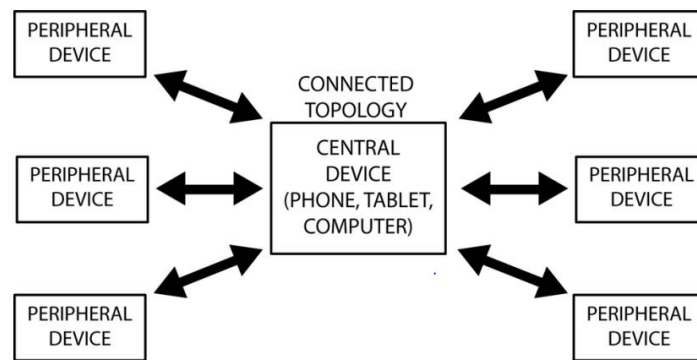


Figure 18. Connected topology: a Central device is connected and exchanges data with multiple Peripheral devices.

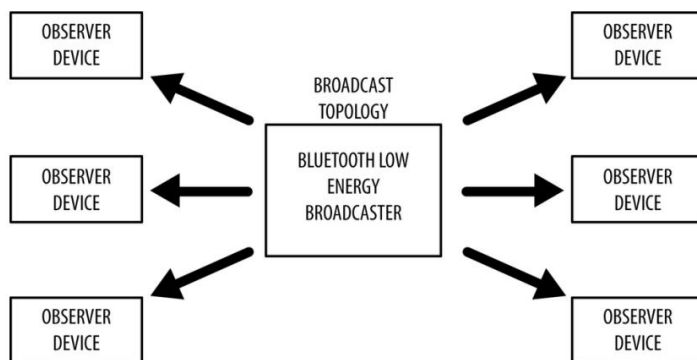


Figure 19. Broadcast topology: a Peripheral device sends data to multiple Central devices simultaneously.

All services using BLE are based on the Generic Attribute Profile (GATT). It is a software interface that defines how devices can send and receive data, describing the concepts of Service and Feature. The GATT exploits the Attribute Protocol (ATT), which is used to contain the data of the Services and characteristics that GATT makes available externally.

The data related to Services and Features are stored in a specific lookup-table using a 16 bytes long identifier called UUID. The first 4 bytes are chosen by the programmer while the others are established by the device itself. Since in BLE it is important to limit the amount of transmitted data as much as possible, the SIG has established a base UUID consisting of the first 12 bytes of the complete UUID. In this way it is not necessary to transmit the entire UUID every time, but it is enough to communicate only the last 4 bytes.

Profiles, Services and Features

The exchange of data between BLE devices is based on high-level objects which are called Profiles, Services and Features, as shown in Figure 20. Each object can be accessed only in the modality provided by the designer (read only, write only, both or none) and, if necessary, only after obtaining an authorization.

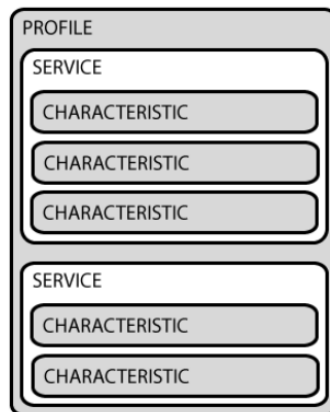


Figure 20. Interface diagram made available by GATT

The Profiles are Services collected and defined by the Bluetooth Special Interest Group (for example the Emergency Profile, the Continuous Glucose Monitoring Profile, the Heart rate Profile, the Blood Pressure Profile) or by the device designer. They combine Services that make available various types of information.

Services and Features

The Services are identified by a UUID, they work as containers for several characteristics.

The Features are the lowest level objects. Each of them encapsulates a pointer to the data (single data of varying length, such as integers or decimals, or an array of values). They are identified by a UUID.

In addition to the read and / or write permissions, there are other properties stored in the descriptor of the characteristic that allow, for example, to define whether the value can be broadcast (i.e. inserted within the advertising packets) or to automatically notify the connected device the availability of new data.

Advertising, Connection and exchange of data

In this phase the BLE device with the Peripheral role sends, at regular intervals, an advertise packet. They are between 6 and 37 bytes in size and contain information about the device that sent them. These packets are sent on 3 of the 40 channels dedicated to the function: 37, 38, 39, highlighted in orange in Figure 21.

If another device is interested to connect with the device that is doing advertising, it can make a scan request to ask additional data.

The advertising interval is set by the programmer and can vary between a few tens of ms and a few seconds. A pseudo-random interval between 0 and 10 ms is also added to avoid the conflicts between advertisements of different slaves.

It is important to carefully select the length of the interval to reasonably balance energy consumption and the speed of being recognized by any other device.

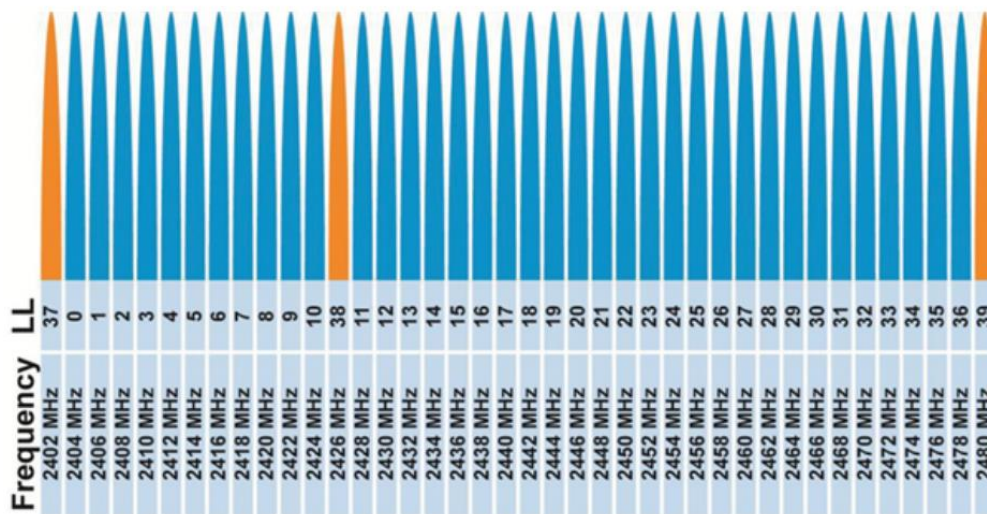


Figure 21. Diagram of the channels used by the BLE. The channels dedicated to advertising are highlighted in orange, while the others are used for data transmission.

Once the Central device has requested additional information from the Peripheral device, it can establish a connection. At this point, some parameters must be established: the connection interval and the Slave Latency.

The devices keep the connection by periodically exchanging data even if there is nothing to transmit: this operation is called Connection event.

The period between one exchange and another is dictated by the connection interval. It can vary between 7.5 ms and 4 s and is chosen independently.

To further save energy, the Peripheral device, if it has no information to exchange, may decide not to respond to the connection event. The number of connection events that the Peripheral device can ignore without dropping the connection is defined by the parameter called Slave Latency.

In this way the devices will communicate more frequently when it is necessary to send data, while more rarely when it is not necessary.

Comparison with other similar technologies

Bluetooth Low Energy is not the only technology that allows you to create Wireless Personal Area Networks. Among them, the most famous are WiFi Direct, NFC and ZigBee.

The first, WiFi Direct is a technology that allows to connect multiple devices to each other without the need for an access point and with a transmission speed comparable to that of classic WiFi. Only one of the devices on the network must be compatible with this technology while the others can be normal WiFi devices.

The main differences with Bluetooth Low Energy are shown in Table 8:

Characteristic	WiFi Direct	BLE
Transmission speed	250 Mbit/s	25 Mbit/s
Transmission distance	About 200 m	About 100 m

Table 8. Comparison between WiFi and BLE

Moreover, it has been estimated that the BLE allows to consume about 3% of the energy consumed by WiFi Direct to perform the same operations.

The second, Near Field Communication, or NFC, is a technology that allows to exchange data at close range. Furthermore, it allows to power the slave device directly via radio waves.

The main defects of this technology are the very limited transmission range (maximum 10 cm) and the absence of an authentication system. Some of the advantages are, however, the very low energy consumption and the ease of use (it is not necessary to enter a PIN or do other procedures to initialize the connection).

The third, ZigBee is a communication standard developed by the ZigBee Alliance which involves the use of low power and low consumption digital antennas. The standard is designed for use with embedded systems that require a low transfer rate and low power consumption. Unlike BLE, it allows to create multiple networks (mesh networks).

However, it is much less widespread, in fact there is no mobile device (smartphone or tablet) that implements it.

Why choose Bluetooth Low Energy

For an application like the one that has been created in this report, one of the fundamental characteristics is the low energy consumption. The sensor must, in fact, be powered by battery and a limited energy consumption allows to increase its duration. However, it is necessary to have a transmission range greater than that allowed by NFC.

The developed interface must also be universal, it must allow the use of different types of sensors, and it must be compatible with the largest possible number of devices.

It is therefore evident that the best technology among those just described is the Bluetooth Low Energy.

Lastly, SparkFun Pro nRF52840 Mini, the core component of the system, is a Bluetooth Development Board, it supports only the Bluetooth technology.

Hardware

The ShockAccel system hardware is shown in Figure 22. The Sparkfun Pro nRF52840 Mini board is the core of the system: the two sensors and the Breakout Board, which houses the micro SD, are connected to it. As described in the previous chapters of the individual components, the connection between Sparkfun and ADXL372 and that between Sparkfun and BMI088 were managed according to the I2C protocol, through the creation of a new node, welding the connections to a plate (red circle in Figure 23). The connection between Sparkfun and SD Breakout Board was managed according to the SPI protocol. Furthermore, the Sparkfun is powered by a LiPo battery of 130 mAh: battery sizing will be explained in a later chapter.

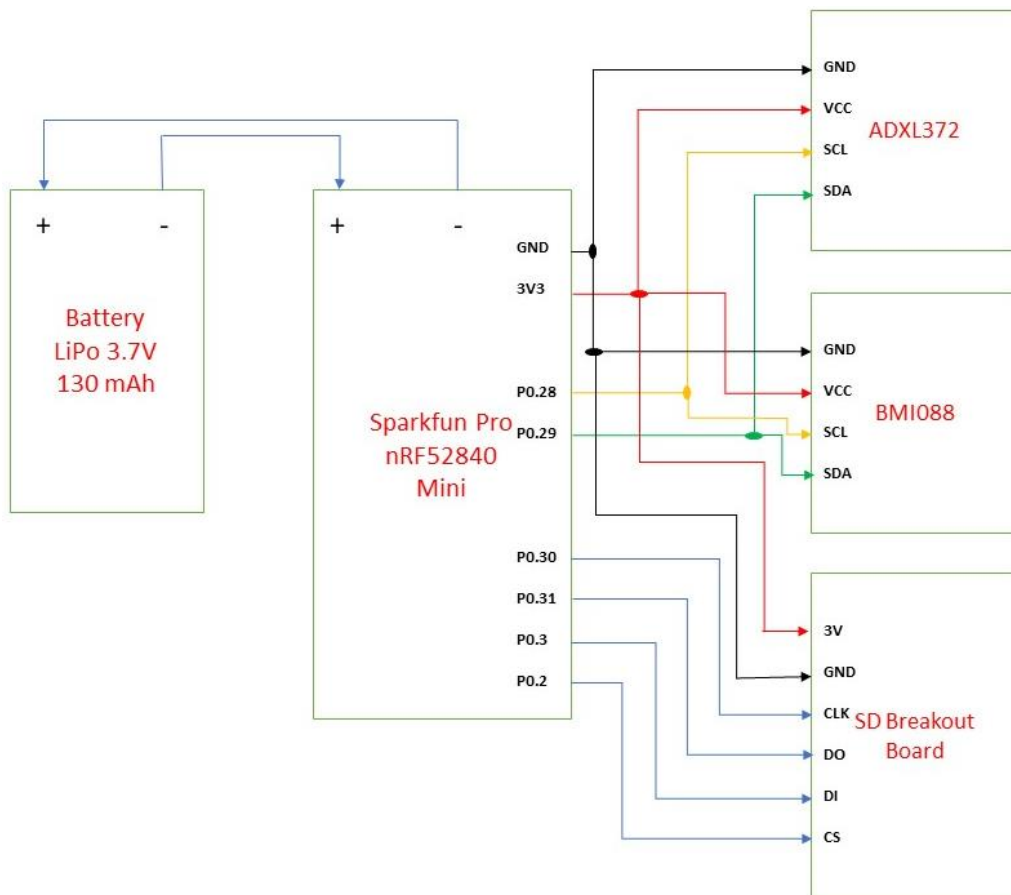


Figure 22. Connection diagram hardware of the ShockAccel system

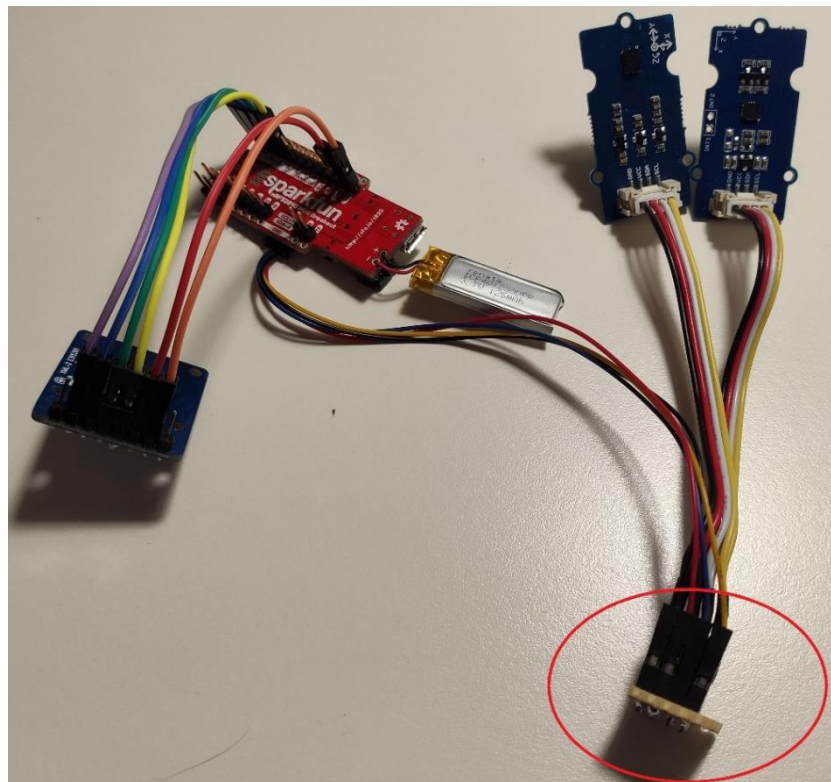


Figure 23. Picture of the hardware configuration

Software

After the description of the devices used in the system and the structure of the hardware, the programming phase of the project starts. The Sparkfun, and the connected sensors, was programmed inside the integrated development environment of Arduino (Arduino IDE).

The web application IDE MIT App Inventor was used for the development of the Android application that has allowed the communication via BLE: the microcontroller, once detected a shock, sent an alarm signal to the clinician's device on the pitchside.

Arduino IDE

C or C++ type are the two programming languages that can be used inside the Arduino IDE, where technical details, such as header files, are largely hidden from users. The existence of extensive libraries and examples simplifies programming.

The programming of the Sparkfun Pro nRF52840 Mini in the Arduino IDE needs a preparatory phase: after installing the latest version of the platform, it is necessary to install the Arduino kernel and add the SparkFun board. Next, the board definitions need to be downloaded, added to the Arduino kernel, and some definition changes need to be made. The details of these procedures are explained more precisely in a tutorial made by Sparkfun, free available online.

MIT AppInventor

MIT AppInventor is an integrated development environment originally developed by the American company Google Inc. to program applications for Android under the Massachusetts Institute of Technology (MIT) license. The graphical interface used allows the users to create an application for a smartphone with the Android system, using a graphical programming language.

No special software is needed to create applications with this development environment. With a single login, all created projects are stored online and can be edited on the main page. Moreover, many extensions can be purchased or downloaded for free. For the application created in this project, the BluetoothLE extension was used, which is available at the link in reference [28].

Programming

Arduino IDE

The program (sketch) consists of 3 main parts or phases:

- introduction phase, where the programmer includes the libraries and defines the global variables and the function prototypes;
- setup phase, where there are instructions executed only at the beginning of the program;
- loop phase, the main part of the program, repeated continuously until the power supply is disconnected or pressed the reset button.

At the end of the loop phase, there are all the functions called up in the setup and loop phases.


```

ShockAccel_final_project
// Board: SparkFun Pro nRF52840 Mini

// Include Libraries
#include <Wire.h>
#include <math.h>
#include <SPI.h>
#include <SD.h>
#include <bluefruit.h>

#include "BMI088.h"

// Defines
#define LED_OFF LOW
#define LED_ON HIGH
#define pin_CS 2 //pin 2 is the CS of the Sparkfun

File file;

```

The libraries included in the sketch are used to:

- manage the connection protocols between the microcontroller and the sensors (Wire.h);
- manage the connection protocols between the microcontroller and the Breakout Board for the SD card (SPI.h);
- have predefined instructions for the operations of reading/writing on SD card (SD.h);
- manage the mathematics aspects (math.h);
- configure the BLE (Bluefruit.h);
- have predefined instructions for the programming of BMI088 (BMI088.h).

```

// Registers for ADXL372
#define ADXL372_ADDRESS (0x53)           // Default Address of the ADXL372
#define ADXL372_REGISTER_XMSB (0x08)     // Address of the XDATA_H Register

```

The address of the accelerometer (ADXL372) is 0x53, while 0x08 is the address of the register where the MSB byte of the X axis value is stored.

As can be seen in Figure 24, the LSB byte of the axis is always on the next address.

0x08	XDATA_H	[7:0]
0x09	XDATA_L	[7:0]
0x0A	YDATA_H	[7:0]
0x0B	YDATA_L	[7:0]
0x0C	ZDATA_H	[7:0]
0x0D	ZDATA_L	[7:0]

Figure 24. Register addresses for X, Y and Z axes


```

// Global variables
const int LED_PIN = 7; // Pin number of the blue LED

uint16_t MAX_G_VALUE = 10; // Threshold of the G value
uint16_t *ptrMaxGVal = &MAX_G_VALUE;
uint16_t MAX_RADS2_VALUE = 2000; // Threshold of the RADS2 value
uint16_t *ptrMaxGVal = &MAX_RADS2_VALUE;

char BLE_VALUE = 0; // Value of the received bluetooth data
char *ptrBleVal = &BLE_VALUE;

bool lin_THRESHOLD = false; // Activating/deactivating the linear threshold
bool *ptrLinThreshold = &lin_THRESHOLD;

bool ang_THRESHOLD = false; // Activating/deactivating the angular threshold
bool *ptrAngThreshold = &ang_THRESHOLD;

```

Then, there is the phase of defining the global variables. MAX_G_VALUE is used to adjust the level of the linear acceleration threshold. MAX_RADS2_VALUE is used to adjust the level of the rotational acceleration threshold. BLE_VALUE is used to store the character sent via Bluetooth from the smartphone. lin_THRESHOLD indicates whether the linear acceleration threshold is activated or not. ang_THRESHOLD indicates whether the rotational acceleration threshold is activated or not. Defined pointers are used to modify the values of variables.

```

uint16_t accelerometer_data[3]; // Storage for axis values
int16_t accelerometer_data_formatted[3];
float accelerometer_data_formatted_float[3];

float wx = 0, wy = 0, wz = 0;
float alpha_x = 0, alpha_y = 0, alpha_z = 0;
int fc_gyr = 400; // 400Hz, important data for passing from w to alpha

BLEUart bleuart; // Uart over ble

```

Other variables have been initialized in which to save the sensor data. Moreover, the mathematical transition between angular velocity and angular acceleration is introduced: the derived operation is approximated to a simple incremental ratio, given by the difference of two consecutive angular velocity values divided by the time step (or multiplied by the sampling frequency).

```

// Function prototypes
void i2c_write(int, byte, byte); // Writes data to the slave's buffer
uint8_t i2c_read(int, byte, int, byte*); // Reads data from the sensor's input register
void init_adxl372(); // Initialization of ADXL372 registers
void init_bmi088(); // Initialization of BMI088 registers
void read_adxl372(); // Read the X-, Y- and Z-axis acceleration data and formats the data
void read_bmi088(); // Read the X-, Y- and Z-axis rotational acceleration data and formats the data
void logData(float, float, float, float, float, float, float, float); // Continuously writes the readed axis values on SD and serial monitor
void check_incoming_BLEdata(char*, bool*); // Check the received bluetooth data
void set_new_Gvalue(uint16_t, uint16_t*); // Sets the new G value threshold
void check_for_linear_threshold_exceeding_value(); // Check for linear threshold exceeding value and sends via bluetooth
void check_for_angular_threshold_exceeding_value(); // Check for angular threshold exceeding value and sends via bluetooth

```

The previous functions have been created to realize the functionality of the system: some have been inserted in the setup and others in the loop.

In the Setup phase the elements connected to the Sparkfun are initialized and set.

```

void setup(void) {

    //-----BLE-----

    // Initialize hardware:
    Serial.begin(9600); // Serial is the USB serial port
    pinMode(LED_PIN, OUTPUT);
    digitalWrite(LED_PIN, LED_OFF); // Turn on-board blue LED off
}

```

First, the LED has been configured, which should signal the Advertising.

```
// Initialize Bluetooth:
Bluefruit.begin();
Bluefruit.setTxPower(8); // Set max power. Accepted values are:
                        // -40, -30, -20, -16, -12, -8, -4, 0, 4, 8
Bluefruit.setName("nRF52_1"); // Set device name, number
bleuart.begin();

// Start advertising device and bleuart services
Bluefruit.Advertising.addFlags(BLE_GAP_ADV_FLAGS_LE_ONLY_GENERAL_DISC_MODE); // LE General Discoverable Mode.
Bluefruit.Advertising.addTxPower(); // Set Tx Power
Bluefruit.Advertising.addService(bleuart); // Add bleuart service
Bluefruit.ScanResponse.addName(); // Add name of device

Bluefruit.Advertising.restartOnDisconnect(true); // Restart if disconnected
Bluefruit.Advertising.setInterval(32, 244); // Set advertising interval (in unit of 0.625ms), first n sec. fast, then slow
Bluefruit.Advertising.setFastTimeout(30); // number of seconds in fast mode
Bluefruit.Advertising.start(0); // Start Advertising
```

Then the Bluetooth is initialized by setting the transmit power, assigning the name of device, adding the service and selecting the Advertising interval.

The Advertising has been set so that if there is a losing of the connection to a device, it will revert to Advertising mode. The device is first in Quick Advertising mode, which means it has an Advertising interval of $32 \times 0.625\text{ms} = 20\text{ms}$. After 30 seconds it then switches to Slow Advertising mode, with an interval of $244 \times 0.625\text{ms} = 152.5\text{ms}$.

```
//-----SD Card-----

Serial.begin(115200);
while ( !Serial ) delay(10); // for nrf52840 with native usb

if (!SD.begin(pin_CS)) {

    delay(6000);
    Serial.println("Error");
    return;
}

delay(6000);
Serial.println("SD");

if (SD.exists("example.txt")) {
    SD.remove("example.txt");
}
```

For SD card initialization, CS pin of the Sparkfun has been selected and tested, if the card can be initialized. If successful, the old text file will be deleted.

```
//-----ACCEL-----

Wire.begin(); // Initiate the Wire library
for(int i=0; i<3; ++i) { // Reset the accelerometer data
    accelerometer_data[i] = 0;
    accelerometer_data_formatted[i] = 0;
    accelerometer_data_formatted_float[i] = 0;
}

init_adxl372(); // Initialization of ADXL372 registers

init_bmi088(); // Initialization of BMI088 registers

delay(1000);
}
```

To save and format the acceleration values, 3 tables have been created. They are reset during initialization. The initialization function of the sensor is then called, which makes it possible to define certain register values. An existing library was used for the initialization of the BMI088.

Thus, the setup phase ends, which the microcontroller performs only once.

In the loop function, various functions are called continuously.

```
void loop(void) {

  //-----BLE-----

  check_incoming_BLEdata(ptrBleVal, ptrThreshold); // Check the received bluetooth data

  if ((lin_THRESHOLD == true) && (BLE_VALUE == 'a')){
    check_for_linear_threshold_exceeding_value();
  }

  if ((ang_THRESHOLD == true) && (BLE_VALUE == 'a')){
    check_for_angular_threshold_exceeding_value();
  }
}
```

First, the function is called to check whether Bluetooth data has been received and if so, it is being processed. The decision of saving or stopping to save the data on SD card is controlled by Bluetooth. The threshold setting and the deleting the old text file work in the same way. Therefore, depending on the Number code received via Bluetooth, different function will be performed.

```
//-----ACCEL-----

read_adxl372();

delay(10);

read_bmi088();

logData(accelerometer_data_formatted_float[0],accelerometer_data_formatted_float[1],accelerometer_data_formatted_float[2],alpha_x,alpha_y,alpha_z);

delay(25);//40Hz 'virtual' sampling frequency. This is fc used in the script Matlab
}
```

The values of the axes of the sensor (firstly ADXL372 and later BMI088) are then read and can be output to the serial monitor of Arduino if necessary. The delay between the ADXL372 and BMI088 reading operation allows to respect the I2C protocol. The logData function was created to write the 6 formatted values (3 linear accelerations and 3 angular accelerations) on the SD card.

Thus, the loop phase ends, which the microcontroller performs continuously.

```
// Initialization of accelerometer registers
void init_adxl372() {

  i2c_write(ADXL372_ADDRESS, 0x3D, 0x00); // Set output data rate at 400 Hz
  i2c_write(ADXL372_ADDRESS, 0x3E, 0x08); // 200 Hz Bandwidth & Low noise operation
  i2c_write(ADXL372_ADDRESS, 0x3F, 0x03); // Full bandwidth measurement mode
}
```

During initialization, TIMING register 0x3D is set at 0x00: an ODR of 400 Hz is selected, the acceleration data is written to the output registers every 2.5 ms. Low noise operation is selected in MEASURE register 0x3E. This increases consumption slightly but reduces the noise level to 1/3 normal noise level. The selected bandwidth for the output signal is set to 200 Hz. The bandwidth of the output signal is thus limited by a 4-pole low-pass filter. This ensures that the Nyquist criterion is not violated with an ODR of 400 Hz. Then the full bandwidth measurement mode is selected in the POWER_CTL (power control) register 0x3F.

```

void init_bmi088() {

    Serial.begin(115200);

    while (1) {
        if (bmi088.isConnected()) {
            bmi088.initialize();
            Serial.println("BMI088 is connected");
            break;
        } else {
            Serial.println("BMI088 is not connected");
        }

        delay(2000);
    }

    bmi088.setGyroScaleRange(RANGE_2000);
    bmi088.setGyroOutputDataRate(ODR_400_BW_47); //ODR of the giroscope set at 400Hz(as ADXL372)
}

```

A library found on the net on the GitHub portal was used for the initialization of the BMI088, with functions already created ad hoc. The correct hardware connection has been checked with an If-else structure, the measurement range of the angular speeds has been set at 2000 ° / s (the maximum) and the ODR at 400Hz (like the ADXL372).

```

// Writes data to the slave's buffer
void i2c_write(int address, byte reg, byte data) {
    Wire.beginTransaction(address);           // Connect to device
    Wire.write(reg);                         // Send output register address
    Wire.write(data);                       // Send data
    Wire.endTransmission();                 // Disconnect from device
}

```

The i2c_write() function is used to write data to the sensor registers: in this program it is used only in the ADXL372 case.

```

// Reads data from the sensor's input register
uint8_t i2c_read(int address, byte reg, int count, byte* data) {
    Wire.beginTransaction(address);           // Connect to device
    Wire.write(reg);                         // Send input register address
    Wire.endTransmission(false);             // Keep the connection active
    Wire.requestFrom(address, count);         // Request data from slave,
                                              // count stands for number of bytes to request

    for (uint8_t i=0; i<count; i++) {
        while(!Wire.available()){            // Slave may send less than requested
        }
        data[i] = Wire.read();                // Read a byte transmitted from slave
    }
}

```

With the i2c_read() function, the registers of the ADXL372 can be read. After establishing the connection with device, multiple bytes can be read. This function is used to read the acceleration values of the axes as shown in Figure 24.

```

// Read the X-,Y- and Z-axis acceleration data and formats the data
void read_adxl372() {

    float acc[10][3]; // Creation Matrix: 10 rows (10 values to average), 3 columns (x,y,z)

    for (int jj=0;jj<10;++jj) { // Fill each rows with the 3 values

        byte bytes[6] = {}; // Buffering of the axis values
        i2c_read(ADXL372_ADDRESS, ADXL372_REGISTER_XMSB, 6, bytes); // Read 6 bytes from the ADXL372

        for (int i=0;i<3;++i) { // Unpack data and formatting
            accelerometer_data[i] = ((bytes[2*i] <<8) | (bytes[2*i + 1])); // 16 bit 2-Complement
            accelerometer_data[i] = accelerometer_data[i] >> 4 ; // 12 bit 2-Complement
            if (accelerometer_data[i] & 0x8000) { // if negativ number, fill with 1 (bit[12-15])
                accelerometer_data[i] |= 0xF000;
            }
            else { // if positiv number, fill with 0 (bit[12-15])
                accelerometer_data[i] |= 0x0000;
            }
            accelerometer_data_formatted[i] = (int16_t)accelerometer_data[i];
            accelerometer_data_formatted_float[i] = (float)(accelerometer_data_formatted[i])/10;// 100mg/LSB scale factor

            acc[jj][i]=accelerometer_data_formatted_float[i]; // Fill the matrix acc
        }
    }

    float somma[3]= {0,0,0};
    for (int jj=0;jj<10;++jj) {
        for (int i=0;i<3;++i) {
            somma[i]=somma[i] + acc[jj][i];
        }
    }

    for (int i=0;i<3;++i) {
        accelerometer_data_formatted[i]=somma[i]/10; // Averaging
    }
}

```

The read_adxl372() function is used to read the 6 bytes or respectively the 6 registers of the sensor. Then the displayed values are formatted such that acceleration values are available in size "g" and have a resolution of 0.1g.

The formatting operation combines the values of the two registers containing the MSB and LSB of the acceleration value of the axis. Values in registers are stored in two's complement. XDATA_H contains all 8 bits most significant (MSB), and XDATA_L contains the 4 least significant bits (LSB) of the 12-bit value (Figure 25). So the two register values are combined, shifted to the right by 4 positions and according to whether it is a positive or negative number, the bits 12-15 are filled with 1 or 0. Then a Cast operation is necessary to have an integer value and finally again to get to the float type. With the division by 10, the scale factor of 100mg/LSB is obtained.

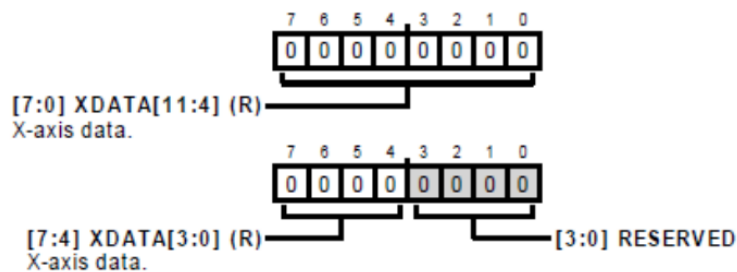


Figure 25. XDATA_H et XDATA_L

Moreover, an averaging operation, through a matrix system, was performed between 10 consecutive values to stabilize the measurement.

```

void read_bmi088() {

    float acc_ang[10][3]; // Creation Matrix: 10 rows (10 values to average), 3 columns (x,y,z)

    for (int jj=0;jj<10;++jj) { // Fill each rows with the 3 values

        float wx_0 = wx; //save the previous value, to calculate the incremental ratio
        float wy_0 = wy;
        float wz_0 = wz;

        bmi088.getGyroscope(&wx, &wy, &wz);

        alpha_x=(wx-wx_0)*fc_gyr; //rotational acceleration in °/s^2
        alpha_y=(wy-wy_0)*fc_gyr;
        alpha_z=(wz-wz_0)*fc_gyr;

        alpha_x = alpha_x * 3.14 / 180; //rotational acceleration in rad/s^2
        alpha_y = alpha_y * 3.14 / 180;
        alpha_z = alpha_z * 3.14 / 180;

        acc_ang[jj][0] = alpha_x; // Fill the matrix acc_ang
        acc_ang[jj][1] = alpha_y;
        acc_ang[jj][2] = alpha_z;
    }

    float somma_ang[3]= {0,0,0};

    for (int jj=0;jj<10;++jj) {
        for (int i=0;i<3;++i) {
            somma_ang[i]=somma_ang[i] + acc_ang[jj][i];
        }
    }

    alpha_x=somma_ang[0]/10; // Averaging
    alpha_y=somma_ang[1]/10;
    alpha_z=somma_ang[2]/10;
}

```

The read_BMI088() function is used to read the 3 values of angular speed captured by the gyroscope inside the BMI088. In this case the reading procedure is lighter than the one performed for the ADXL372, thanks to the already working bmi088.getGyroscope instruction found in the library. The linear acceleration values measured by BMI088 were ignored, because the values of ADXL372 had already been read: also in this case there was an ad hoc instruction, bmi088.getAcceleoremeter.

Subsequently the derivative operation, the conversion from $\frac{^{\circ}}{s^2}$ to $\frac{rad}{s^2}$ and the averaging (the same method as for linear accelerations) were performed.

```

void logData(float lin_x, float lin_y, float lin_z, float ang_x, float ang_y, float ang_z) {

    String msg =
        String(lin_x)+"\t"+
        String(lin_y)+"\t"+
        String(lin_z)+"\t"+
        String(ang_x)+"\t"+
        String(ang_y)+"\t"+
        String(ang_z);

    Serial.println(msg);

    file = SD.open("example.txt", FILE_WRITE);

    if(file){
        file.println(msg);
        file.close();
    }
}

```

The logData function is used to write the message on the SD card: the 6 values in line are spaced by a tab, each line is transcribed every 25ms, as the delay at the end of the loop indicates. Moreover, the values are printed continuously also on the serial monitor.

```
//Check the received bluetooth data
void check_incoming_BLEdata(char *ptrBleVal, bool *ptrThreshold){
    char c;
    if (bleuart.available()) {          // If data has come in via BLE

        c = bleuart.read();             // store recived character sent over BLE

        if (c == 200){
            *ptrBleVal = 200;           // Start saving on SD card
        }
        else if(c == 201){
            *ptrBleVal = 201;           // Stop saving on SD card
        }
        else if(c == 202){
            *ptrBleVal = 202;           // Remove old file to create new one
        }
        else if(c == 203){
            *ptrThreshold = true;        // Activate Threshold
        }
        else if(c == 204){
            *ptrThreshold = false;       // Desactivate Threshold
        }
        else{
            set_new_Gvalue(c,ptrMaxGVal); // Set the new G value Threshold
        }
        else{
            set_new_rads2value(c,ptrMaxrads2Val); // Set the new rad/s^2 value Threshold
        }
    }
}
```

If there is data received via Bluetooth, the value is analyzed. The data are either characters, or integer unsigned values. Numbers are used to control various features. The integer value represents the new threshold value that the user wants to set.

```
// Sets the new G value threshold
void set_new_Gvalue(uint16_t c, uint16_t *ptrMaxGVal){
    *ptrMaxGVal = c;
}

// Sets the new rad/s^2 value threshold
void set_new_rads2_value(uint16_t c2, uint16_t *ptrMaxrads2Val){
    *ptrMaxrads2Val = c2;
}
```

The set_new_Gvalue() and set_new_rads2value() functions update the old threshold values with the new values by pointers.

```
// Check for linear threshold exceeding value and sends via bluetooth
void check_for_linear_threshold_exceeding_value(){

    if(abs(accelerometer_data_formatted_float[0]) > MAX_G_VALUE){          // Test if one of the axes value is bigger than defined MAX_G_VALUE
        bleuart.write(1);                                                  // Send device number
        bleuart.write(abs(accelerometer_data_formatted_float[0]));         // Send value via bluetooth
    }
    if(abs(accelerometer_data_formatted_float[1]) > MAX_G_VALUE){
        bleuart.write(1);
        bleuart.write(abs(accelerometer_data_formatted_float[1]));
    }
    if(abs(accelerometer_data_formatted_float[2]) > MAX_G_VALUE){
        bleuart.write(1);
        bleuart.write(abs(accelerometer_data_formatted_float[2]));
    }
}
```

```

// Check for angular threshold exceeding value and sends via bluetooth
void check_for_angular_threshold_exceeding_value() {

    if(abs(alpha_x) > MAX_RADS2_VALUE) {                // Test if one of the axes value is bigger than defined MAX_RADS2_VALUE
        bleuart.write(1);                               // Send device number
        bleuart.write(abs(alpha_x));                     // Send value via bluetooth
    }
    if(abs(alpha_y) > MAX_RADS2_VALUE) {
        bleuart.write(1);
        bleuart.write(abs(alpha_y));
    }
    if(abs(alpha_z) > MAX_RADS2_VALUE) {
        bleuart.write(1);
        bleuart.write(abs(alpha_z));
    }
}

```

Finally, it is checked whether one of 6 axes value exceeds the threshold value. If so, the value is sent to the smartphone via Bluetooth.

MIT AppInventor

Regarding the smartphone application, the MIT AppInventor development environment was used. Certain elements, such as configuration of the Bluetooth connection, could be used for the application. The application interface is visible in the Figure 26. It shows some buttons, labels, two text fields and a list.

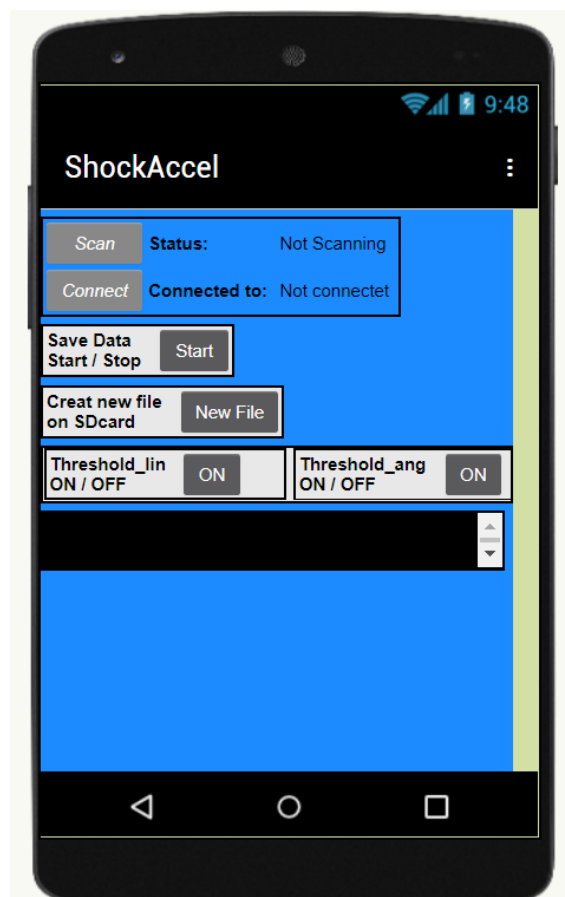


Figure 26. Interface of ShockAccel application

Pushing the "Scan" button (Figure 27), the process of scan of available devices with a BLE connection can be started.

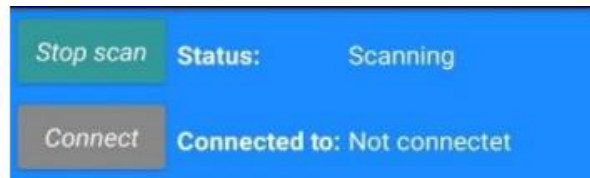


Figure 27. Scanning operation

A list opens showing all devices available to connect (Figure 28).



Figure 28. List of the available devices

After having selected the device and pressed the "Connect" button, the device is connected (Figure 29). The "Disconnect" button then allows the user to disconnect the device if necessary.

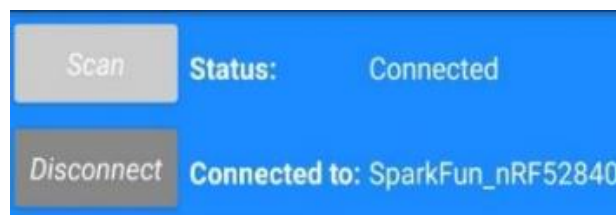


Figure 29. Connection with the SparkFun achieved

If the connection to the SparkFun card is established, the "Save Data Start / Stop" switch allows to start or stop the data storage on SD card. The "New File" button allows to delete the existing text file and create a new one.

By pressing the switch "Threshold_lin ON/OFF", or "Threshold_ang ON/OFF", a text field and another button are enabled. In the text field, a desired threshold value can be now entered: pressing the button "Set max. G value" (in the second case, the new button is "Set max. Rad value"), the threshold value is updated. If a value acceleration exceeds the threshold value, it is sent to the application and displayed on the "Threshold exceeded value" label (Figure 30). In addition, an alarm tone, different for the linear and the angular cases, sounds and the mobile phone vibrates.

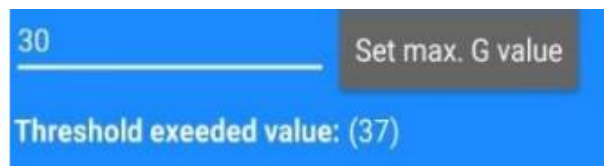


Figure 30. Threshold exceeded video signal

Below, the block diagram of the program in the MIT AppInventor environment is showed.

```

initialize global Gvalue to 0
initialize global Radvalue to 0
initialize global Service_UUID to "6e400001-b5a3-f393-e0a9-e50e24dcca9e"
initialize global Characteristic_UUID2 to "6e400002-b5a3-f393-e0a9-e50e24dcca9e"
initialize global Characteristic_UUID3 to "6e400003-b5a3-f393-e0a9-e50e24dcca9e"

```

```

initialize global Service_UUID2 to "6e400004-b5a3-f393-e0a9-e50e24dcca9e"
initialize global Characteristic_UUID4 to "6e400005-b5a3-f393-e0a9-e50e24dcca9e"
initialize global Characteristic_UUID5 to "6e400006-b5a3-f393-e0a9-e50e24dcca9e"

```

```

when ScanButton -> .Click
do
  if ScanButton -> .Text == "Scan"
  then
    call BluetoothLE1 -> .StartScanning
    set ScanButton -> .Text to "Stop scan"
    set ScanButton -> .BackgroundColor to #000000
    set BLEdeviceList -> .Visible to true
    set LabelStatus -> .Text to "Scanning"
  else
    call StopScan
  end if
end do

to StopScan
do
  set ScanButton -> .Text to "Scan"
  set LabelStatus -> .Text to "Stopped Scanning"
  set ScanButton -> .BackgroundColor to #000000
  call BluetoothLE1 -> .StopScanning
end do

when BluetoothLE1 -> .DeviceFound
do
  set BLEdeviceList -> .ElementsFromString to BluetoothLE1 -> .DeviceList
end do

```

```

when ON_OFF_save -> .Click
do
  if BluetoothLE1 -> .IsDeviceConnected
  then
    if ON_OFF_save -> .Text == "Start"
    then
      call BluetoothLE1 -> .WriteBytes
      serviceUuid get global Service_UUID
      characteristicUuid get global Characteristic_UUID2
      signed false
      values "200"
      set ON_OFF_save -> .Text to "Stop"
    else
      call BluetoothLE1 -> .WriteBytes
      serviceUuid get global Service_UUID
      characteristicUuid get global Characteristic_UUID2
      signed false
      values "201"
      set ON_OFF_save -> .Text to "Start"
    end if
  end if
end do

```

```

when ConnectButton -> .Click
do
  if ConnectButton -> .Text == "Connect"
  then
    call StopScan
    call BluetoothLE1 -> .Connect
    index BLEdeviceList -> .SelectionIndex
    set LabelStatus -> .Text to "Connecting"
  else
    call BluetoothLE1 -> .UnregisterForValues
    serviceUuid get global Service_UUID
    characteristicUuid get global Characteristic_UUID3
    call BluetoothLE1 -> .Disconnect
    set Text_ConnectedTo -> .Text to "Not connected"
    set ConnectButton -> .Text to "Connect"
  end if
end do

when BluetoothLE1 -> .Connected
do
  set LabelStatus -> .Text to "Connected"
  set LabelConnect -> .Text to BluetoothLE1 -> .ConnectedDeviceName
  set BLEdeviceList -> .Visible to false
  set ConnectButton -> .Text to "Disconnect"
  call BluetoothLE1 -> .RegisterForBytes
  serviceUuid get global Service_UUID
  characteristicUuid get global Characteristic_UUID3
  signed false
end do

when BluetoothLE1 -> .Disconnected
do
  set LabelStatus -> .Text to "Disconnected"
end do

```

```

when NewFile_Button -> .Click
do
  if BluetoothLE1 -> .IsDeviceConnected
  then
    call BluetoothLE1 -> .WriteBytes
    serviceUuid get global Service_UUID
    characteristicUuid get global Characteristic_UUID2
    signed false
    values "202"
  end if
end do

when ON_OFF_LIN_Th -> .Click
do
  if BluetoothLE1 -> .IsDeviceConnected
  then
    if ON_OFF_LIN_Th -> .Text == "ON"
    then
      call BluetoothLE1 -> .WriteBytes
      serviceUuid get global Service_UUID
      characteristicUuid get global Characteristic_UUID2
      signed false
      values "203"
      set ON_OFF_LIN_Th -> .Text to "OFF"
      set HorizontalArrangement9 -> .Visible to true
      set HorizontalArrangement7 -> .Visible to true
      set ButtonSetGValue -> .BackgroundColor to #000000
    else
      call BluetoothLE1 -> .WriteBytes
      serviceUuid get global Service_UUID
      characteristicUuid get global Characteristic_UUID2
      signed false
      values "204"
      set ON_OFF_LIN_Th -> .Text to "ON"
      set ButtonSetGValue -> .BackgroundColor to #000000
      set HorizontalArrangement9 -> .Visible to false
      set HorizontalArrangement7 -> .Visible to false
    end if
  end if
end do

```

```

when ON_OFF_ANG_Th -> .Click
do
  if BluetoothLE1 -> .IsDeviceConnected
  then
    if ON_OFF_ANG_Th -> .Text == "ON"
    then
      call BluetoothLE1 -> .WriteBytes
      serviceUuid get global Service_UUID2
      characteristicUuid get global Characteristic_UUID4
      signed false
      values "203"
      set ON_OFF_ANG_Th -> .Text to "OFF"
      set HorizontalArrangement9 -> .Visible to true
      set HorizontalArrangement12 -> .Visible to true
      set ButtonSetRadvalue -> .BackgroundColor to #000000
    else
      call BluetoothLE1 -> .WriteBytes
      serviceUuid get global Service_UUID2
      characteristicUuid get global Characteristic_UUID4
      signed false
      values "204"
      set ON_OFF_ANG_Th -> .Text to "ON"
      set ButtonSetRadvalue -> .BackgroundColor to #000000
      set HorizontalArrangement9 -> .Visible to false
      set HorizontalArrangement12 -> .Visible to false
    end if
  end if
end do

```

```

when ButtonSetRadvalue -> .Click
do
  if BluetoothLE1 -> .IsDeviceConnected
  then
    if ON_OFF_ANG_Th -> .Text == "OFF"
    then
      set global Radvalue to format as decimal number TextRadvalue -> .Text
      places 1
      call BluetoothLE1 -> .WriteBytes
      serviceUuid get global Service_UUID2
      characteristicUuid get global Characteristic_UUID2
      signed false
      values get global Radvalue
    end if
  end if
end do

```

```

when ButtonSetGValue -> .Click
do
  if BluetoothLE1 -> .IsDeviceConnected
  then
    if ON_OFF_LIN_Th -> .Text == "OFF"
    then
      set global Gvalue to format as decimal number TextGvalue -> .Text
      places 1
      call BluetoothLE1 -> .WriteBytes
      serviceUuid get global Service_UUID
      characteristicUuid get global Characteristic_UUID2
      signed false
      values get global Gvalue
    end if
  end if
end do

```

```

when BluetoothLE1 -> .BytesReceived
do
  serviceUuid characteristicUuid byteValues
  if BluetoothLE1 -> .IsDeviceConnected
  then
    set ValueThreshold -> .Text to get byteValues
    call Sound1 -> .Play
    call Sound1 -> .Vibrate
    millisecs 200
  end if
end do

```

Results

After the saving data, a txt file is created in the micro SD card (Figure 31). 3 values for X, Y and Z linear acceleration are followed by 3 values of rotational acceleration (around X, Y and Z axes). With a small modification of the sketch, similar graphics can also be obtained on the Arduino IDE serial monitor. The resolution of 0.1g of the linear acceleration is visible: this resolution is sufficient for the purpose of the project, which is to monitor head shocks during sports activities that can even exceed 100g.

The time column has not been printed on the screen, the sampling is in any case uniform and is at a frequency of 40Hz, as the loop is updated every 25ms, as indicated with the delay function.

The example measurement shown in the Figure was carried out keeping the sensors on a flat surface and facing upwards (it is possible to see the 1g measured on the z axis), making some few small movements to test the reactions of the sensors.

*ShockAccel.txt - Blocco note di Windows

File	Modifica	Formato	Visualizza	?		
-0.20	0.30	1.00	0.00	0.85	0.43	
-0.10	-0.10	1.10	-1.28	0.00	0.00	
0.20	0.10	1.00	0.85	-2.98	-0.43	
0.00	-0.20	1.10	-0.43	0.43	-0.85	
0.20	0.00	1.20	1.28	2.13	0.43	
0.10	-0.40	1.10	-0.85	-1.70	0.43	
0.10	0.10	1.20	-0.43	1.70	0.00	
0.20	-0.10	1.30	0.43	-2.13	0.43	
0.40	0.40	1.20	-0.43	0.00	-1.28	
0.20	-0.10	1.10	0.43	1.28	0.00	
-0.10	0.00	1.10	0.43	-0.43	0.00	
0.00	-0.10	1.00	-2.13	0.85	1.28	
0.00	-0.10	1.30	2.56	0.43	-0.43	
0.00	-0.30	1.10	-1.28	-0.43	0.00	
-0.30	-0.20	1.10	-0.85	0.00	-0.43	
0.10	0.00	1.10	1.28	0.43	0.43	
0.20	-0.30	1.00	0.00	-1.28	0.00	
0.20	0.00	1.40	-0.43	0.43	-0.43	
0.10	0.30	1.10	1.28	0.00	0.00	
0.10	0.10	1.10	0.00	-0.43	0.85	
-0.10	-0.10	1.10	-0.85	0.43	-0.85	
0.10	0.30	1.20	-1.28	2.13	1.28	
0.30	0.50	1.10	1.70	-2.98	-1.28	
-0.30	0.10	1.30	0.43	0.00	-0.85	
1.50	0.00	1.10	-0.43	-0.43	1.28	
0.20	0.10	1.50	-0.85	0.43	-0.85	
0.10	-0.20	1.00	1.70	-0.43	0.85	
0.20	-0.20	1.10	0.85	-0.43	-1.28	
0.20	0.00	1.40	-1.28	0.00	0.85	

Figure 31. File ShockAccel.txt

The txt file was used as input for a Matlab script capable of reading the file and of plotting the 3 linear accelerations and the 3 rotational accelerations in two different subplots (Figure 32).

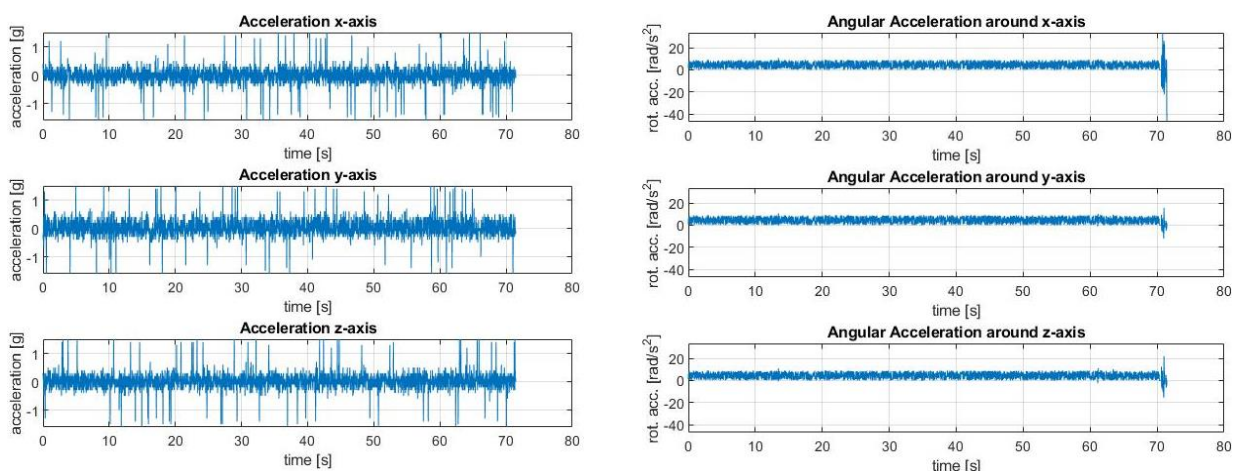


Figure 32. Two subplots from the Matlab script representing 3 linear accelerations and 3 rotational accelerations

Furthermore, the GSI and HIC calculations have been implemented in the script if the shock exceeds the threshold criteria (10 g on one of the 3 linear axes or $2000 \frac{rad}{s^2}$ on one of the 3 angular axes).

The test was carried out by making small movements with the sensors, as can be seen from the abrupt and small variations of the angular accelerations, while in the last seconds of the test even larger movements were tried. The test lasted about 70s, but it could continue measuring for another time.

Consumptions

The sizing of the battery is an important step inside the design chain. A theoretical estimate of consumption was made on the basis of the datasheets.

ADXL372

The consumption of the acceleration depends on the selected ODR, as shown in the Figure 33. The sampling frequency of the project is set at 400Hz. In addition, consumption increases if the ADXL372 works in “Low Noise” mode: Table 9 shows that consumption increases by a third at ODR=3200Hz.

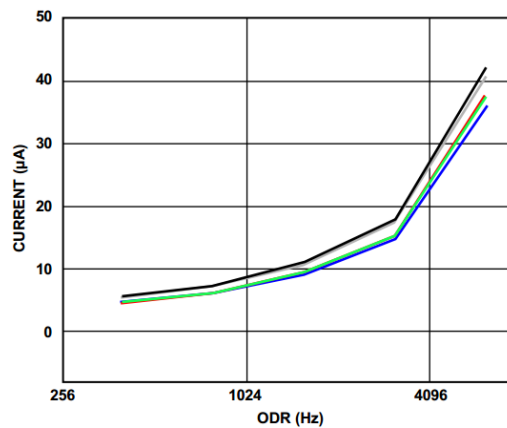


Figure 33. Trend of the Measurement Mode Current versus the variation of ODR

Mode	Typical RMS Noise (LSB)	Typical Current Consumption (µA)
Normal Operation ¹	3.5	22
Low Noise ¹	3	33

¹ $V_s = 2.5$ V, ODR = 3200 Hz, and bandwidth = 1600 Hz.

Table 9. Consumption of ADXL372 in different modes

The current for a 400 Hz ODR is approximately 6 µA. In “Low Noise” mode, consumption is therefore around 10 µA.

BMI088

In the datasheet of BMI088 the consumptions are indicated for a specific ODR or for a specific range of measurement; it is presumed that there is a variation in consumption based on the variation of the ODR, (reasonably, as the sampling frequency increases, the supply current should increase), but for the calculations for the sizing of the battery the data in Table 10 and Table 11 were considered: 150µA as Supply Current for the accelerometer (with a range of measurement of 24g) and 5mA as Supply Current for the gyroscope (with a ODR of 1kHz).

Parameter	Symbol	Condition	Min	Typ	Max	Units
Total Supply Current in Normal mode	I_{DD}	VDD = VDDIO = 3.0V, 25°C, g_{FS4g}		150		μA
Total Supply Current in Suspend Mode	I_{DDsum}	VDD = VDDIO = 3.0V, 25°C		3		μA
Power-up time	t_{s_up}	Time to first valid sample from suspend mode			1	ms

Table 10. Electrical parameter specification accelerometer inside BMI088

Parameter	Symbol	Condition	Min	Typ	Max	Unit
Supply Current in Normal Mode	I_{DD}	VDD = VDDIO = 3.0V, 25°C, ODR = 1kHz		5		mA
Supply Current in Suspend Mode	I_{DDsum}	VDD = VDDIO = 3.0V, 25°C		25		μA
Supply Current in Deep Suspend Mode	I_{DDdsum}	VDD = VDDIO = 3.0V, 25°C		<5		μA
Start-up time	t_{su}	to $\pm 1\%$ of final value; from power-off		30		ms
Wake-up time	t_{wusm}	From suspend- and deep suspend-modes		30		ms
Wake-up time	t_{wufpm}	From fast power-up mode		10		ms

Table 11. Electrical parameter specification gyroscope inside BMI088

Carte SD

For the SD card, in the datasheet is only indicated that in standard mode a maximum of 100 mA is required for the operations of reading and writing data, as shown in Table 12.

Mode	Maximum Value
Standard Mode (25 MHz)	
Sleep	350 μA
Read	100 mA
Write	100 mA

Table 12. Current consumption in SanDisk microSD

SparkFun Pro nRF52840 Mini

The component that consumes the most on the SparkFun board is the Bluetooth transceiver. For an output power of +4 dBm, the estimated only run current is 21.4 mA (Table 13).

Symbol	Description	Min.	Typ.	Max.	Units
$I_{TX,PLUS8dBm,DCDC}$	TX only run current (DC/DC, 3 V) $P_{RF} = +8$ dBm	..	14.8	..	mA
$I_{TX,PLUS8dBm}$	TX only run current $P_{RF} = +8$ dBm	..	32.7	..	mA
$I_{TX,PLUS4dBm,DCDC}$	TX only run current (DC/DC, 3 V) $P_{RF} = +4$ dBm	..	9.6	..	mA
$I_{TX,PLUS4dBm}$	TX only run current $P_{RF} = +4$ dBm	..	21.4	..	mA

Table 13. Radio Current Consumption of the Sparkfun

Table 14 summarizes the maximum consumption estimated by each component after analyzing the respective datasheets.

Components	Estimated consumptions
ADXL372	10 μ A
BMI088	(0.15+5) mA
SD CARD	Maximum 100 mA
SPARKFUN	21.4 mA

Table 14. Estimated consumption of the ShockAccel system components

The measurements carried out showed an absorption current of the entire system (Sparkfun equipped with two sensors, connected via Bluetooth with the smartphone and during the data writing phase on the SD card) of 15mA.

Battery sizing

An ice hockey match normally lasts 60 minutes (three halves - also called "thirds" - of 20 minutes). Since each stoppage of play involves stopping the clock, a match actually lasts about twice the actual time, so 2 hours. The battery must therefore have a minimum nominal capacity of:

$$15 \text{ mA} \cdot 2h = 30 \text{ mAh}$$

In a conservative view of the project, a 125mAh LiPo battery of 3.7V was used (Figure 34).

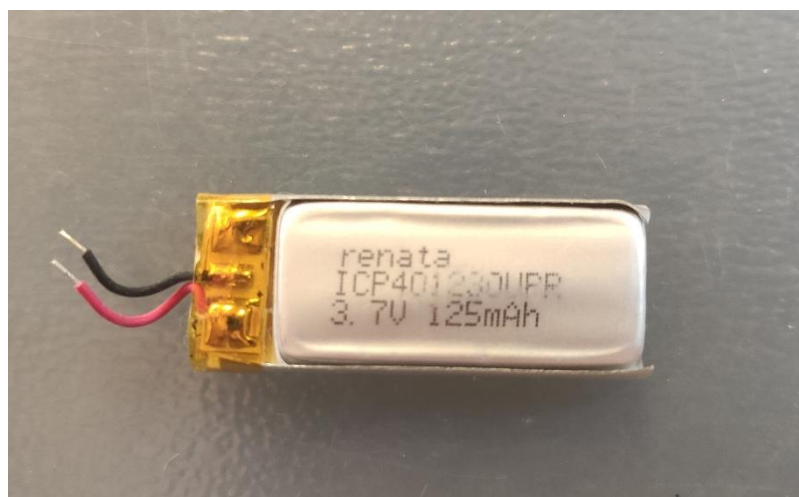


Figure 34. 125mAh LiPo battery

Costs

SparkFun Pro nRF52840 Mini can be purchased on the Sparkfun website for \$ 29.95. The ADXL372 is sold, equipped with the Seeed Groove Studio packaging, at \$ 24.50. The BMI088 is sold, equipped with the Seeed Groove Studio packaging, at € 15.02. The SD Breakout Board can be purchased on the Digi-Key Electronics website for € 6.28. The 16 gb micro SD can be purchased in an electronics store for € 15.94. The 125mAh LiPo

battery of 3.7V can be purchased on the Amazon website for \$ 9.72. The Table summarizes the costs in Euro of the individual components, with the relative exchange rates. The costs of PCB (Printed circuit board) miniaturization and the subsequent housing in the helmet have not been assessed.

The prices, summarized in Table 15, were assessed only through commercial advertisements: costs would drastically decrease considering the commercial conventions between research centers and electronic production companies.

COMPONENTS	COST €
SparkFun Pro nRF52840 Mini	25,74
ADXL372	21,05
BMI088	15,02
SD Breakout Board	6,28
Micro SD 16gb	15,94
LiPo battery 125mAh	8,35
TOTAL (single components)	92,38

Table 15. Commercial price of single components of ShockAccel system

Comparisons

After carefully analyzing 3 devices on the market, and after describing the product designed, it was possible to compose a summary Table 16 of the main features.

	HIT System	Shockbox	Checklight	ShockAccel
Year	2003	2011	2013	2020
On helmet	YES	YES	NO	YES
Microcontroller	TI MSP430F148	Not declared	PIC24FJ64GA004	Sparkfun nRF52840 mini
Sensor	6 x 1axis accelerometers (ADXL193)	Binary electromechanical force switches	1 x 3axis accelerometer (Bosch BMA250) + 1 x 3axis gyroscope (ST L3G4200D)	1 x 3axis accelerometer (ADXL372) + 1 x 3axis gyroscope (BMI088)
Range	±250g	10g – 250g	• ±16g • ±2000°/s	• ±200g • ±2000°/s • ±28000 $\frac{rad}{s^2}$
Resolution	Not declared	Not declared	• 31,25 mg/LSB • 16 °/s/LSB	• 100 mg/LSB • 16 °/s/LSB
ODR	1000 Hz	10 kHz	• 2 kHz (max of acc.) • 800 Hz (max of gyr.)	• 400 Hz for ADXL372 • 400 Hz for BMI088
Consumption	• Battery: 1000mAh • 84mA (Transmitted mode)	180mA (Bluetooth ON)	• Battery: 80mAh • 31,6mA	• Battery: 125mAh • 15mA
Cost	\$160.00	\$149.00	\$149.99	\$108.15

Table 16. Comparison among the head-shock monitoring systems investigated in this study and ShockAccel system

Theory of the Finite Element Method

The Finite Element Method (FEM) is a numerical modeling technique suitable for solving complex and large engineering problems, described by differential equations, in a simplified and approximate way: the use of discretization operations allows to transform the aforementioned differential equations into algebraic equations, easier to solve and accessible to automatic calculation.

As known, the origin of the FEM is the matrix calculation of structures. At the end of the 60s the new powerful calculation machines started to allow the use of discretization methods to solve structural analysis. Since then, the method has evolved and extended to the analysis of issues of various physical nature, thus becoming fundamental for the world of industry and research.

To study a phenomenon, in a given spatial domain, the variables of temperature, displacement or any other physical measurement are functions of each point of that domain, thus presenting an infinite number of unknowns to be evaluated. By discretizing everything into a finite number of subdomains, called finite elements, which are easy to solve, the original problem then becomes approximated and described by a finite number of unknowns.

The discretization, used in complex calculations, is not easy to learn and requires specific knowledge of the subject studied, the mechanical behavior of the structure and numerical analysis techniques. The use of calculation and pre-processing programs makes it possible to simplify the complexity of the computation, through the simplified generation of discretization, with the addition of a series of parameters useful for the various choices of the user, in order to solve the problem in the most suitable way possible.

The initial phase consists in the discretization of the spatial domain in which we want to study the phenomenon in a number of finite elements, of various shapes and sizes, represented by a series of base points, called Nodes, in which the responses of the system are calculated. The information necessary for the description of each element represents the Mesh of the finite elements, the grid, and within them the field of unknowns is approximated, by interpolating the physical values that each unknown assumes in the aforementioned Nodes, placed on the outline of each element.

The FEM method, through the discretization, is able to significantly reduce the complexity of the structural problem of the element under consideration, creating a system of algebraic equations that can be expressed in the matrix form:

$$[K] \cdot \{V\} = \{F\} \quad (3)$$

in which the arrays $\{V\}$ and $\{F\}$ represent the displacements e and the applied loads, referred to specific nodes, while $[K]$ is the stiffness matrix of the elements present in the system.

The stiffness matrix of each single element must be obtained, first, in the local system of the single element and then in the global system of the entire structure, in this way they can then be aggregated in such a way as to create a global stiffness matrix for the entire structure. Later, known the nodal loads and the system of applied forces, the nodal displacements can be determined.

The definition of the matrix system of equations (3) is the main step to apply the finite element method. In fact, it is based on the hypothesis that in the smallest elements, created during the discretization, the trend of the variables is quite regular and can be found relatively simply, with variation laws that can be exponentials or polynomials of suitable degree, so that they can be described with a limited approximation.

In most cases, to simplify and speed up the calculations, the average 2D surface is extracted from the global element in 3D, as shown in Figure 35, while the thickness is added as an associated property on the 2D element. The use of this surface allows to significantly reduce the mesh creation times and the analysis times.

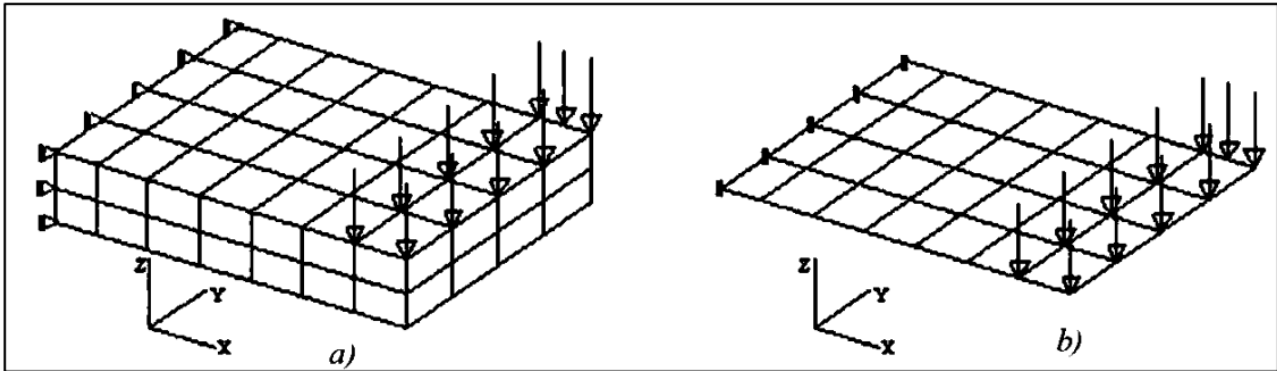


Figure 35. Shelf discretized with 3D (a) and 2D elements (b).

The realization of the various conditions to describe equilibrium and congruence happens only if the model is well delineated, in its geometric characteristics, in a complete and precise way, so that the discretization creates a set of elements and nodes, uniquely identifying in the system of reference adopted both locally and globally. Therefore, a numbering (Figure 36) is adopted for both entities in such a way as to associate the local coordinates and connectivity to the reference system under consideration.

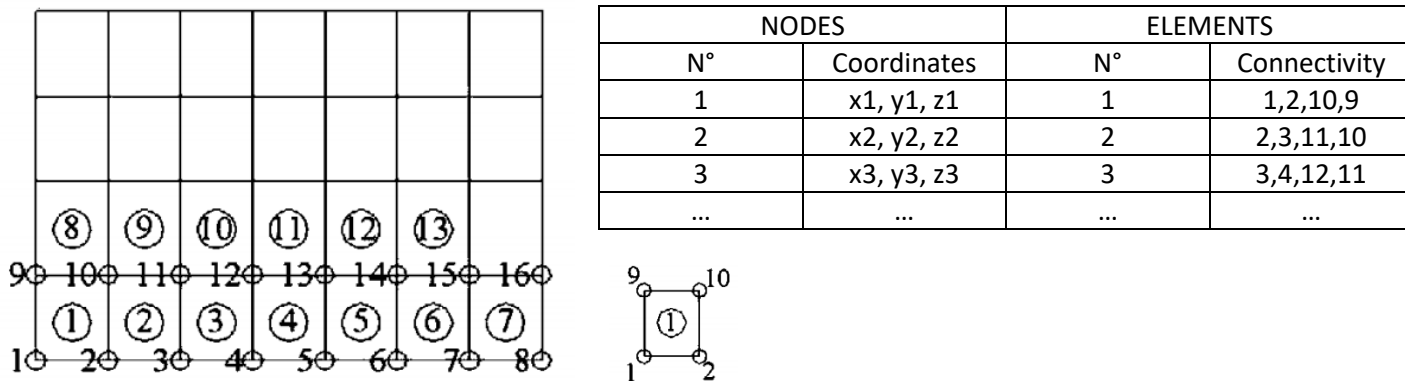


Figure 36. Numbering and identification Nodes and elements

Implicit and Explicit Method

In FEM modeling it's possible to choose two different integration procedures to solve the system of equations that has been acquired through the discretization of the model geometry:

- Calculation method Implicit;
- Calculation method Explicit

Both methods are defined 'direct' because before numerical integration there are no transformations of coordinates and equations.

$$[M]\{\ddot{S}\} + [C]\{\dot{S}\} + [K]\{S\} = \{F(t)\} \quad (4)$$

To solve the system (4) it is necessary to calculate the values \ddot{S} , \dot{S} , S from the initial time $t = 0$ to time $t = T$ known the values \ddot{S}_0 , \dot{S}_0 , S_0 . To do this, the time interval U is divided into n time intervals $\Delta t = \frac{T}{n}$ and the

solution is calculated using discrete values of t , interspersed of Δt , assuming a law of variation of displacements, velocities and accelerations inside the interval Δt . It is an approximate method, because the error depends on the time step chosen.

In the "Implicit" method, there is a "step by step" calculation system in which the accuracy of the results of each single step allows the continuation of the analysis based on certain convergence criteria. The result will be independent of time and this leads it to be preferred in models where the touch or impact between the surfaces occurs at a reduced speed. Furthermore, it is a method that is generally unconditionally stable and without critical time step.

In the implicit method the displacement vector is a function of the previous displacements, already known, while accelerations and velocities remain unknown and are actuals. The equilibrium equation is written at time $t + \Delta t$ (5):

$$[M] \{\ddot{S}\}_{t+\Delta t} + [C] \{\dot{S}\}_{t+\Delta t} + [K] \{S\}_{t+\Delta t} = \{F(t)\}_{t+\Delta t} \quad (5)$$

The Houbolt method, whose graph is shown in Figure 37, is considered an implicit method.

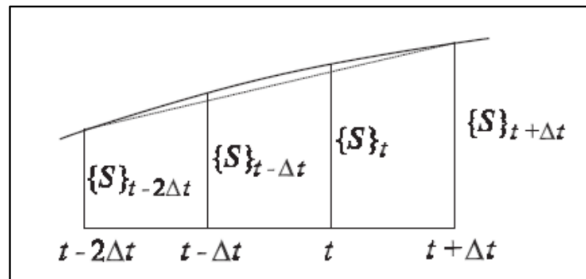


Figure 37. Calculation scheme, Houbolt method

Acceleration and velocity are assumed at time $t + \Delta t$ as [2.4] and [2.5]:

The unknown displacements $\{S\}_{t+\Delta t}$ are obtained from the equilibrium equation written at time $t + \Delta t$ in (5), to then get:

$$\begin{aligned} \left(\frac{2}{\Delta t^2} [M] + \frac{11}{6\Delta t} [C] + [K] \right) \{S\}_{t+\Delta t} \\ = \{F(t)\}_{t+\Delta t} + \left(\frac{5}{\Delta t^2} [M] + \frac{3}{\Delta t} [C] \right) \{S\}_t - \left(\frac{4}{\Delta t^2} [M] + \frac{3}{2\Delta t} [C] \right) \{S\}_{t-\Delta t} \\ + \left(\frac{1}{\Delta t^2} [M] + \frac{1}{3\Delta t} [C] \right) \{S\}_{t-2\Delta t} \end{aligned}$$

Set $[\bar{K}]$ and $\{\bar{F}\}_{t+\Delta t}$ as in (6):

$$[\bar{K}] = \frac{2}{\Delta t^2} [M] + \frac{11}{6\Delta t} [C] + [K] \quad (6)$$

$$\begin{aligned} \{\bar{F}(t)\}_{t+\Delta t} = \{F(t)\}_{t+\Delta t} + \left(\frac{5}{\Delta t^2} [M] + \frac{3}{\Delta t} [C] \right) \{S\}_t - \left(\frac{4}{\Delta t^2} [M] + \frac{3}{2\Delta t} [C] \right) \{S\}_{t-\Delta t} \\ + \left(\frac{1}{\Delta t^2} [M] + \frac{1}{3\Delta t} [C] \right) \{S\}_{t-2\Delta t} \end{aligned}$$

$\{S\}_{t+\Delta t}$ is obtainable from (7) as:

$$[\bar{K}] \{S\}_{t+\Delta t} = \{\bar{F}\}_{t+\Delta t} \quad (7)$$

Houbolt's method requires the factorization of $[\bar{K}]$.

In the “Explicit” method the displacement vector is a function of the solutions calculated at the previous instant, with speeds and accelerations already known because they refer to a previous instant. The method is conditionally stable.

The equilibrium equation is written at the instant of time t (8).

$$[M] \{\ddot{S}\}_t + [C] \{\dot{S}\}_t + [K] \{S\}_t = \{F(t)\}_t \quad (8)$$

The Central Differences method (Figure 38) is the most used as example among the explicit methods.

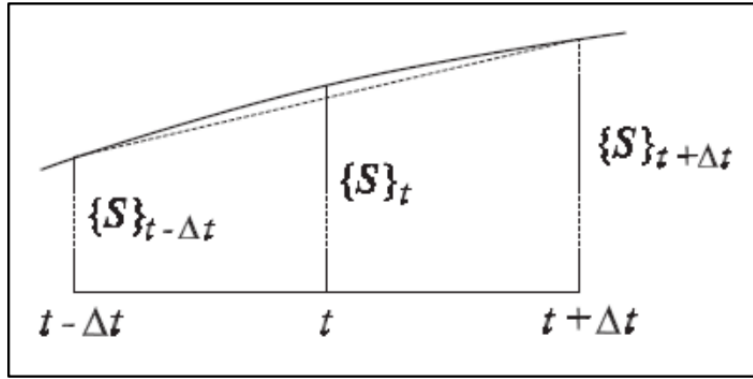


Figure 38. Calculation scheme, Central Differences method

Acceleration (9) and velocity (10) are assumed:

$$\{\ddot{S}\}_t = \frac{1}{\Delta t^2} (\{S\}_{t-\Delta t} - 2\{S\}_t + \{S\}_{t+\Delta t}) \quad (9)$$

$$\{\dot{S}\}_t = \frac{1}{2\Delta t} (-\{S\}_{t-\Delta t} + \{S\}_{t+\Delta t}) \quad (10)$$

From the equilibrium equation (8), the unknown displacements $\{S\}_{t+\Delta t}$ are obtained by substituting:

$$\left(\frac{1}{\Delta t^2} [M] + \frac{1}{2\Delta t} [C] \right) \{S\}_{t+\Delta t} = \{F\}_t - \left([K] - \frac{2}{\Delta t^2} [M] \right) \{S\}_t - \left(\frac{1}{\Delta t^2} [M] - \frac{1}{2\Delta t} [C] \right) \{S\}_{t-\Delta t}$$

set $[\bar{M}]$ and $\{\bar{F}\}_t$ in (11):

$$\begin{aligned} [\bar{M}] &= \frac{1}{\Delta t^2} [M] + \frac{1}{2\Delta t} [C] \\ \{\bar{F}\}_t &= \{F\}_t - \left([K] - \frac{2}{\Delta t^2} [M] \right) \{S\}_t - \left(\frac{1}{\Delta t^2} [M] - \frac{1}{2\Delta t} [C] \right) \{S\}_{t-\Delta t} \end{aligned} \quad (11)$$

Obtaining therefore $\{S\}_{t+\Delta t}$ from (12):

$$[\bar{M}] \{S\}_{t+\Delta t} = \{\bar{F}\}_t \quad (12)$$

The method requires the factorization of the matrix $[\bar{M}]$ and, moreover, since the previous value is required, it is necessary to set up a starting procedure when $t = 0$. It could be (13):

$$\{S\}_{-\Delta t} = \{S\}_0 - \Delta t \{\dot{S}\}_0 + \frac{\Delta t^2}{2} \{\ddot{S}\}_0 \quad (13)$$

To choose the integration step Δt the procedure is as in (14):

$$\Delta t \leq \Delta t_{crit} = \frac{T_n}{\pi} \quad (14)$$

Where T_n is the smallest period of the system.

Multiphysics Simulation

The suggested way to program simulations that lead to correct results is to design models that can represent an alternative to the actual experimental phase in the laboratory. The model, which must be validated with a standard, always represented by an experimental phase, will allow to obtain a deeper knowledge of the process under analysis, as well as a higher level of accuracy and a reduced timing.

A multiphysics simulation (or rather, a multiphysics simulation software) allows to assemble any number of physical phenomena (mechanical, thermodynamical, electronical etc.) and lets the user free to characterize, through equations and expressions, the physical behavior of the components of the model. For this type of customization, the software uses a graphical user interface (GUI).

Moreover, usually, the effects that occur in one physical domain also affect behavior in the other physical domains. It is therefore clear that understanding multiphysics behavior is one of the most important challenges to overcome in order to accurately predict product performance or analyze a physical event.

Bibliographic research: Finite Element Models of the Head-Brain system

With the evolution of the technology and the improving of the computational power of the computer, since the late 70s several research groups have developed Finite Element Models of the Head-Brain system (FEHMs) [29].

Through this type of simulation is possible to compute variables and experimental data that would otherwise be inaccessible, because of both legal and ethical reasons, such as stress and strain of the brain tissue and the Intracranial Pressure (ICP), that would be better injury indicators than linear or angular acceleration, measured by inertial sensors implanted externally on the head or on the helmet [30]. Thus, the idea was to study head trauma also from a biomechanical point of view, through the design of mathematical models.

The data extracted from these models have been used to better study the mechanism and dynamics of TBIs, to test new predictors comparing the traditional ones, and as a reference to validate subsequent FEM models.

The evolution of FEHMs over the years has consisted in an improvement of geometrical details and anatomical features, caused by the progression of the medical imaging, a better understanding, and therefore better characterization of the materials, of the biological tissues inside the head, a greater computational capacity of modern computers and a boom of births of new commercial FEM softwares (Abaqus, Ansys, LS-Dyna, Comsol...).

The majority of the FEHM described in the literature have been developed in response to the request of the car manufacturers (to study the head impacts in car accidents) and of the neurosurgical teams, for the training of preoperative phase. For these purposes, the model must be founded on a solid anatomical base to obtain results that are as closer as possible to the reality. The starting points are a huge collection of images of the head-brain system (Computer Tomography, CT, and Magnetic Resonance Imaging, MRI) and a database of material properties, in terms of elasticity, viscosity, viscoelasticity, plasticity... not only of the tissues, but also of the objects that will interact with them, as the helmet, the ground, the bumper of the car (in case of pedestrian incident), the surgical instruments....

In the last fifty years the consequences of head trauma have been studied also from a biomechanical point of view using mathematical models. Currently, the parameter used in order to quantify the severity of a head damage because of a collision is the HIC. The limits of this parameter were already discussed in the previous chapters.

FE Head models can be used to study the possibility of an injury due to an external load, as indicated in the flow chart in Figure 39, that is the type of injury involved in sport activity.

The human anatomy is the base for the develop of the models to reach the maximum level of precision in head-brain representation and to obtain realistic results: for this purpose the characterization of the materials of the internal organs of the skull (meninges, hemispheres, CSF, cerebellum, basal ganglia etc.) and the interactions between them becomes crucial.

Most of the FE models of the skull and brain described in the literature have been developed in response to the request of the car manufacturers (to study the head impacts in car accidents) and of the neurosurgical teams, for the training of preoperative phase.

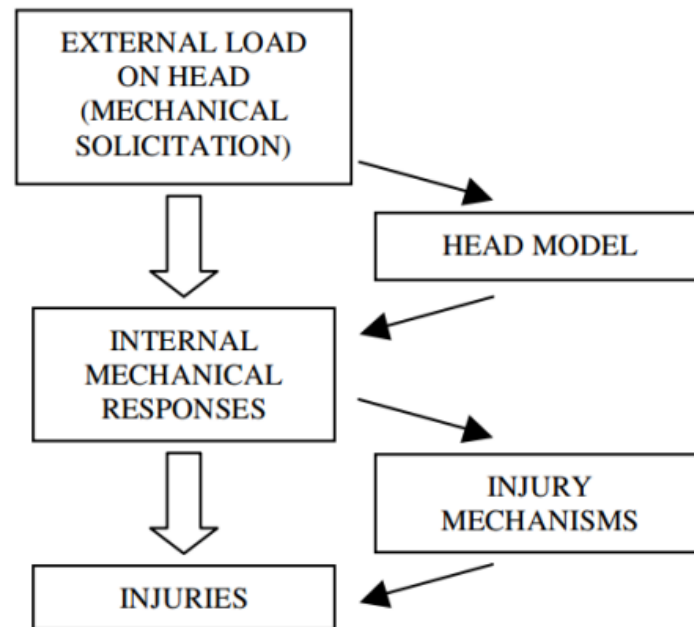


Figure 39. Block diagram representing injury sequence model

In the next chapters the various FEHMs found in the literature will be discussed in detail, explaining the characteristics, the software used, the geometry, the materials used, the number of nodes and components, the boundary condition applied on them, the field of application, the obtainable parameters and the methods of validation.

EEVCWG17 head form FE model

This model was born by the collaboration of the European Enhanced Vehicle-Safety Committee Working Group 17 (EEVC WG17) and the commercial company ARUP. The model consists in a head-form impactor, with 3 different sizes: child (2.5kg and 130mm of diameter), adult (4.8kg and 160mm of diameter) and small adult (3.5kg and 160mm of diameter), to cover as much of the population as possible (Figure 40). All of 3 sizes show four parts: the back plate, sphere, skin, and null shell. The basic idea of the geometry is very simple: the head-form is composed by aluminum spheres covered by thick vinyl skins.

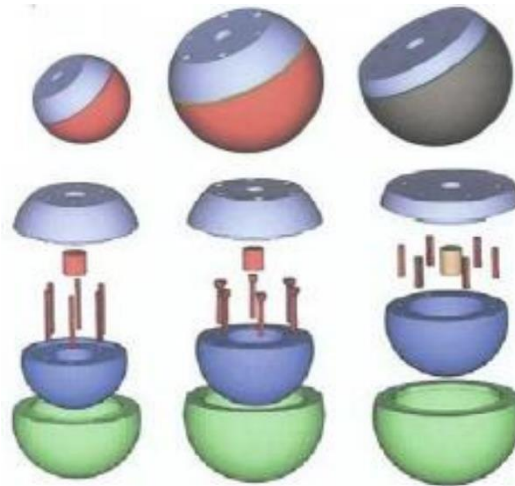


Figure 40. EEVC WG17 head-form models of heads of three categories of pedestrians / persons involved in the accident: on the left Child-size, on the center Adult-size, on the right Small Adult-size

The model is composed by about 14000 nodes and more than 12000 shells [31].

During the simulation, initial linear and rotational conditions have been imposed on the head-form: it was lifted up till 107mm, and its initial angular speed was set to 4rad/s. The aim of the project was to reproduce a car accident, in which the pedestrian hits his head against the hood of the car.

The head-form contains an “accelerometer” (point where the software measures the acceleration), placed in the hypothetical center of gravity of the model, from which the software is able to calculate the HIC. The parts of the model and the internal accelerometer are visible in Figure 41.

The model was verified and validated through a comparison of the output data (HIC values) with practical tests [32].

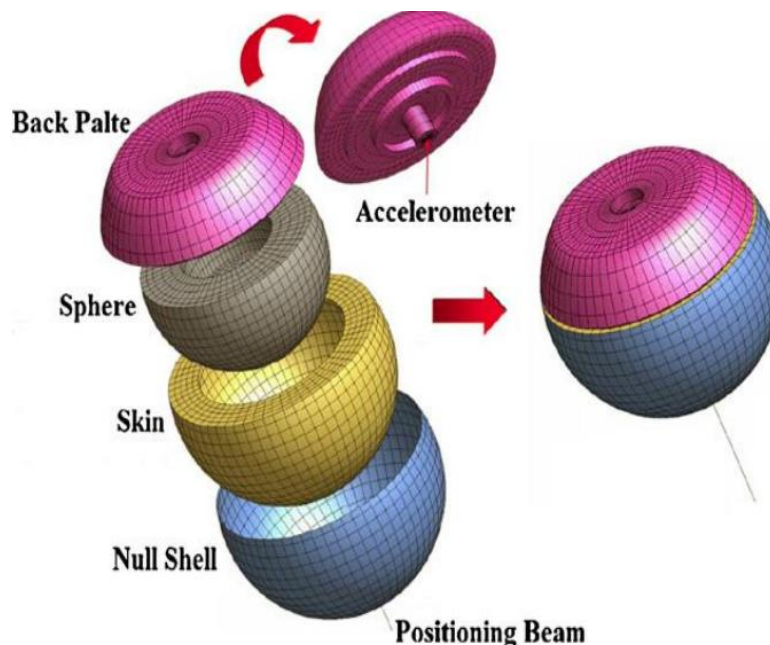


Figure 41. Exploded view of the EEVCWG17 head form

Wayne State University FEHM

Wayne State University developed a FEHM called Head Injury Model (WSUHIM), with the aim to study the human head and brain injury. The model included all the essential anatomical features, such as the scalp,

skull, diploë, dura, falx cerebri, tentorium, arachnoid pia, sagittal sinus, transverse sinus, cerebral spinal fluid (CSF), hemispheres of the cerebrum, cerebellum, brainstem, lateral ventricles, third ventricles, and bridging veins (Figure 42) [33]. The facial model is composed by zygomatic bones, maxillae, the hard palate, the lacrimal bones, the nasal bones, then turbinate bones, the vomer, and the mandible. In total, the elements are over 330000 elements.

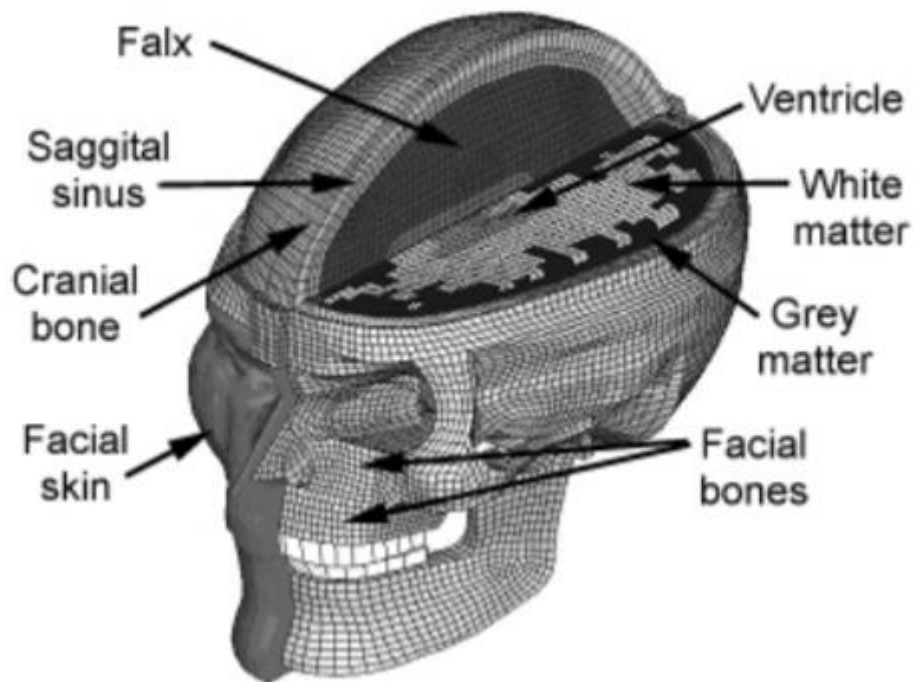


Figure 42. Section view of the WSUHIM

The characterization and specification of tissues with different materials, materials such as the gray and white matter, allowed to increase the accuracy in predicting the DAI, based on maximum principal strain, and in determining the pressure inside the cranium: in this view, the model was used in the medical-surgical field.

Moreover, this model can calculate brain Von Mises stress, maximum principal strain, linear acceleration, MPS contour, DAI map, and HIC.

The origin of the geometry of the model are X-ray images, then processed by the software PAM-CRASH [34]. The author of the model has chosen to associate a linear viscoelastic behavior to all parts, defining separately the gray and white matter.

The validation phase has been passed through the comparison with the results (intracranial pressure) obtained by the work of Nahum in 1977 [35].

In conclusion, the WSUHIM didn't contain a metrics for the risk of injury. The detection was based on a threshold (set by the user) system on the local stress/strain, to determine if there were any focal injuries.

Human body head model (HBM)

Yang et al. [36], in 2008, designed a FEHM (Figure 43), called Human body head model (HBM), using the software LS-DYNA. It was composed by 32 parts, 66624 nodes, 11514 shell elements and 49607 brick elements, with an effective mass of 4.4 kg. The model, entirely bilateral symmetric, represented an adult head-form with all essential anatomical features belonged to a 50th percentile man.

The anatomical parts are scalp, skull with outer table, diploe, and inner table, dura, falx cerebri, tentorium, falx cerebelli, pia, cerebral spinal fluid (CSF), cerebrum, cerebellum, brain stem and ventricles.

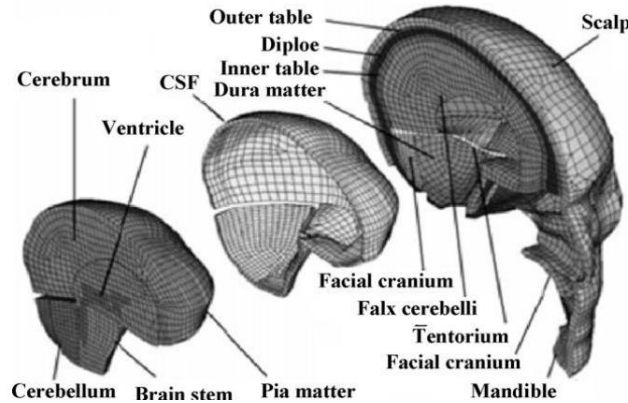


Figure 43. Exploded view of the HBM

The connections of the meninges are designed in this way: dura adhered to the internal surface of skull, pia is attached to the outer brain surface, while there is a tied contact definition between pia and dura, miming the behavior of arachnoid, not present in this model. Moreover, a sliding contact is defined between the CSF and the dura mater. In conclusion, the neck is isolated from the head.

The material properties are based on the study of Willinger, et al. [37], the parameters are presented in Table 17.

Tissue	Material type	Bulk modulus K/GPa	Poisson ratio μ	G_0 /kPa	G_∞ /kPa	β/s^{-1}
Cortical bone	Elastic	15.000	0.21	—	—	—
Trabecular bone	Elastic	4.600	0.05	—	—	—
Falx tentorium	Elastic	0.032	0.45	—	—	—
Scalp	Elastic	0.017	0.42	—	—	—
Cerebellum	Viscoelstic	2.190	—	10	2	80
Cerebrum	Viscoelstic	2.190	—	12.5	2.5	80
CSF	Viscoelstic	1.050	—	1	0.9	80
Brain stem	Viscoelstic	2.190	—	22.5	4.5	80

Table 17. Material properties of HBM

The authors have developed an algorithm, based on the specific intracranial pressure measured by the model, to identify the severity and mechanism of the injury and its location on the head.

The validation phase was conducted through a comparison with the results of Nahum [38], in particular, his test n°37, where the impactor, with a mass of 5.59 kg. had an impact speed of 9.94 m/s, and the head model is inclined of 45°.

HBM was also validated, against the results obtained by EEVC WG17, for the injury of the head of a pedestrian versus the hood of a car (Figure 44), showing excellent performances. His great bio-fidelity, being very close to a real human head, has allowed to calculate HIC with high efficiency.

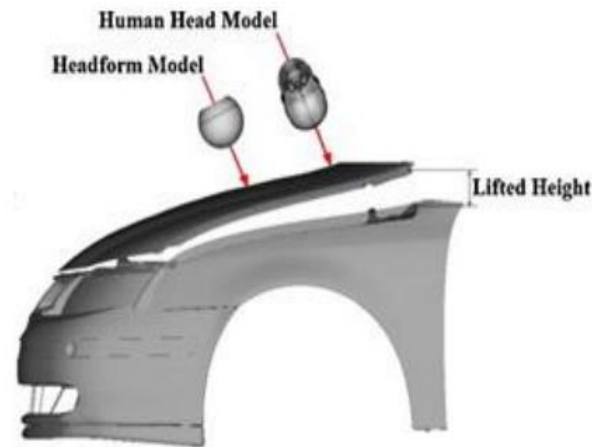


Figure 44. Simulation of an accident between the pedestrian head model and the hood of a car

Image-based FE Method (Simpleware's model)

Synopsys Simpleware, an American electronic design automation company, and the U.S. Naval Research Laboratory (NRL) developed a FEHM, starting from high-resolution scan data, to simulate head impacts. The phase of image processing was conducted through the software ScanIP, while the multiphysics simulation was conducted through the software Abaqus [39].

The phase of image processing consisted in 4 steps:

1. Data Acquisition, from an MRI, T1-weighted scan in coronal plane (Figure 45), with a resolution of 1.03516 mm x 1.03516 mm. The characteristics of the volunteer correspond to the American 50th percentile male.
2. Image segmentation, through the software ScanIP, when regions of interest (ROI's) were identified and labelled within the greyscale data. In this way the anatomical features inside the system head-brain were reconstructed to generate volume meshes. 33 anatomical structures were identified in total.
3. Image enrichment and modification, CAD models of the anatomical structures were incorporated to the segmented images. This technique is useful also to incorporate new structure external to the image, such as the helmet or a specific pathology.
4. Image-based mesh generation, the segmented images were converted into meshes, through the approach multi-part Extended Volumetric Marching Cubes (EVOMAC). All elements of the model (3.72 million) are tetrahedral.



Figure 45. MRI image data

The authors did not provide the scientific literature with details of the materials used within the model to characterize the anatomical features. The contact definitions between two different layers were very similar to those explained in the previous models.

For running the simulations, the authors have chosen the software Abaqus/Explicit, for a total time of simulation of 24 hours, due to an implementation of the technique of mass scaling which has set a time increment of 50 ns.

As the previous models, the validation phase was conducted through the comparison with the results of the Nahum's 37th experimental test in 1977 (Figure 46), showing a correlation of the intracranial pressure of parietal and frontal regions over 95%.

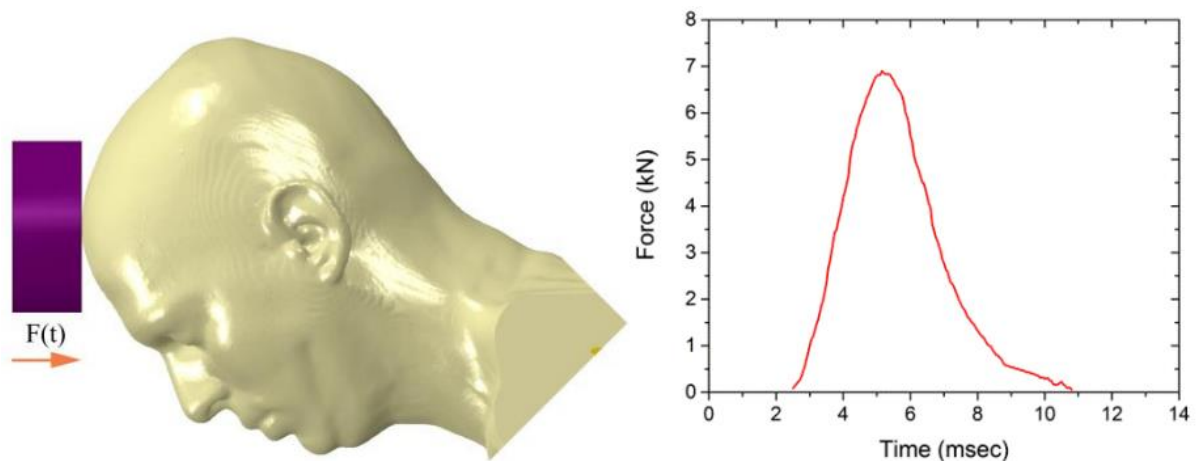


Figure 46. Set up and results of the validation phase of Simpleware's model

Figure 47 shows the trend of the pressure wave, which goes on, during the loading, from the frontal to the occipital region of the brain. This temporal-spatial representation can be used as tool to predict injury regions or injury patterns, basing the detection on a threshold system on the effective stress or on the maximum principal strain.

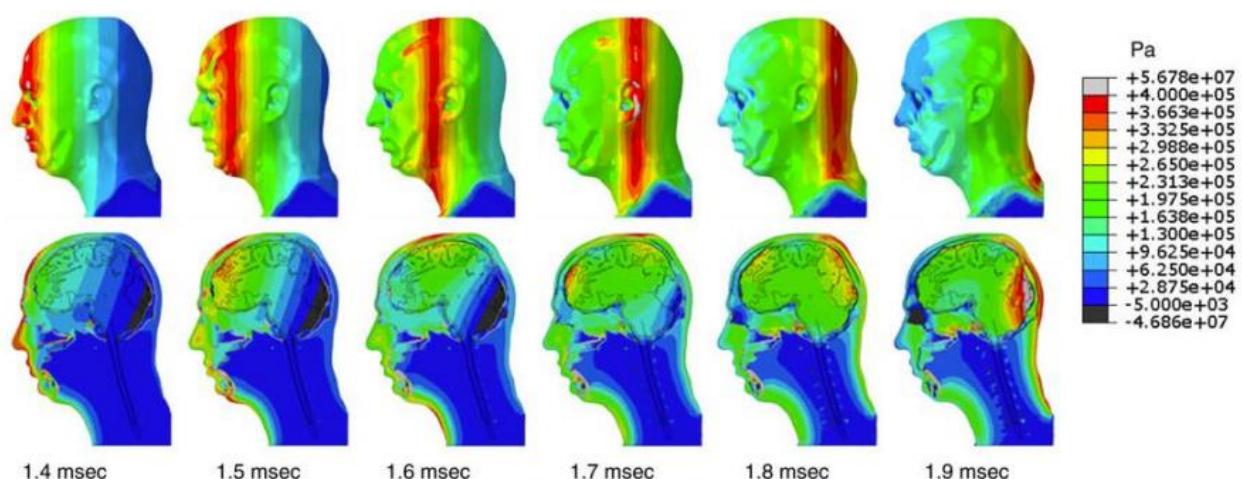


Figure 47. Pressure wave calculated during the simulation of the Simpleware's model

In conclusion, Simpleware's approach was the most capable to generate detailed structures from the brain geometry and head profile, thanks to the high fidelity allowed by MRI images. The idea of adapting meshes to the segmented image is also innovative and represents a new challenge. On the other hand, the model is

obviously limited by the number of elements, which, being very computationally heavy, can be a problem for certain applications.

TNO Head Model

The Nederlandse Organisatie voor Toegepast Natuurwetenschappelijk Onderzoek (in English Netherlands Organization for Applied Scientific Research, or TNO) created a FEHM in 2008, showing it during then International Workshop on Modelling [40]. The model (Figure 48) was composed of 22 parts, 13868 elements, all hexagonal, and 15637 nodes.

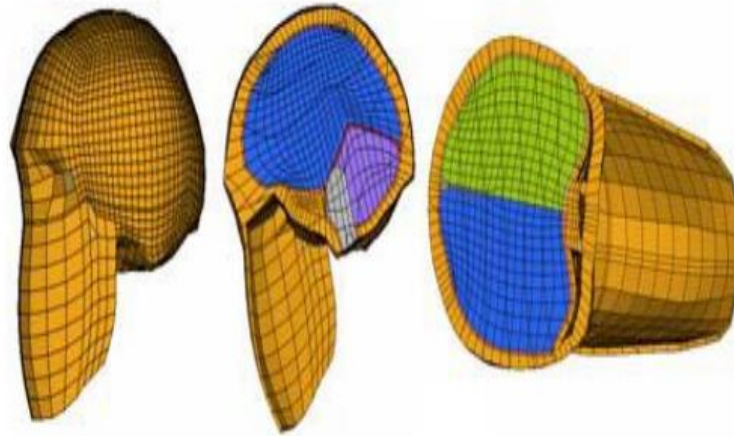


Figure 48. Three different views of TNO Head model

The material properties of some parts of the model (cranium visceral, dura mater, falx cerebri, and tentorium cerebellar) are shown in Table 18, and were represented by homogeneous isotropic linear elastic material, while others (brain, cerebellum and brainstem) were characterized by a viscoelastic behavior.

	Name	Material properties			
		Material model	ρ , kg/m ³	E, MPa	γ
1	cranium visceral	linear elastic	1 800	6 500	0.21
2	dura mater	linear elastic	1 133	31.5	0.45
3	falx cerebri	linear elastic	1 133	31.5	0.45
4	tentorium cerebelli	linear elastic	1 133	31.5	0.45
5	road	linear elastic	2 500	40 000	0.26

Table 18. Linear elastic properties of the parts of TNO

The phases of modeling and assembly of all the components was carried out on program Hypermesh [41], while the software LS-DYNA was adopted for the simulations. In total, the model was composed by 22 parts, 15637 nodes and 13868 elements.

3 different types of simulations were carried out, 3 collisions that can occur during road accidents between a pedestrian and a vehicle: the collision of the frontal, temporal and occipital region of the pedestrian's head with the ground, varying the impact speed in three steps, 10 km/h, 15 km/h and 20 km/h. In all 3 cases the simulations showed an increase in the stress wave at the impact location, accompanied by peaks or valleys in the intracranial pressure trend. Moreover, a linear relationship was noted between the speed of impact and the amplitude of the mechanical response only in cases of collisions in the occipital and temporal regions, linear relationship not so clear after a frontal impact [40].

As mentioned before, the trend of intracranial pressure characterized by the presence of simultaneous peaks and valleys, in opposite points of the brain, is the manifestation of the coup-contrecoup injury phenomenon, which will be explained later in the description of the Harvard University model. The coup-contrecoup injury is composed of two different types of stresses, one of extension and one of compression: in all 3 simulation cases the results of this model showed a greater amplitude in the compression phase.

The scientific literature did not provide information regarding the modalities with which the validation phase of the model was conducted.

THUMS FE Model

In 2002 the professor Iwamoto and other researchers of Toyota Labs developed a finite element model of human body called THUMS and launched it at 7th International LS-DYNA Users' Conference [42], being LS-DYNA the software used in the project. The aim of the project, commissioned by an automotive manufacturer such as Toyota, was to predict multiple skeletal injuries that a whole body could suffer during car crashes. The human model was designed to represent the American male 50 percentile) occupant, including bones, ligaments, and some soft tissues introduced for absorbing impact energy. The total body model can be divided in 2 'sub-models': the head/brain and the thoracic/abdominal. In this chapter only the first will be described in detail (Figure 49).

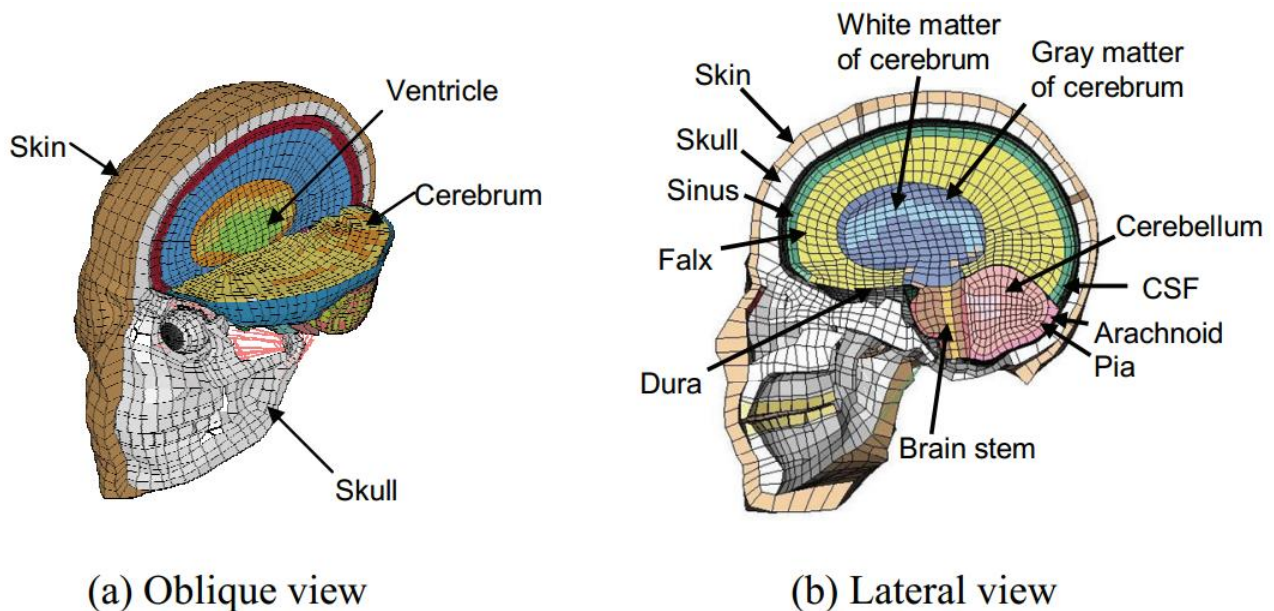


Figure 49. Oblique view (a) and lateral view (b) of the THUMS

The FEHM is composed by three main parts: the skull, the brain, and the skin. The skull contained cortical bone (solid elements) and spongy bone (shell elements). The brain was composed by hexagonal solid elements and shell elements: cerebrum, with distinct white and gray matter, cerebellum, brainstem and

cerebral spinal fluid (CSF) were represented by hexagonal solid elements, while dura mater, arachnoid, membrane meningitis, sickle brain and tentorium by shell elements.

The total weight of the model was 4.39 kg, with 49700 elements, 24100 solids and 25200 shells [43].

About materials used, the authors have chosen an elastic-plastic material for the skull and a non-compressible viscoelastic material for the brain [44].

The validation phase was conducted through a comparison with the test data reported by Trosseille et al. [45]. The experimental test consisted in a cylindrical impactor of 23.4 kg that hit the facial region with 7 m/s: the hit occurred in the anterior-posterior direction. The results of the impact were 1000 m/s^2 of linear acceleration plus 7600 rad/s^2 of rotational acceleration applied to the center of gravity of a head of a passenger sitting in the seat of a car (Figure 50).

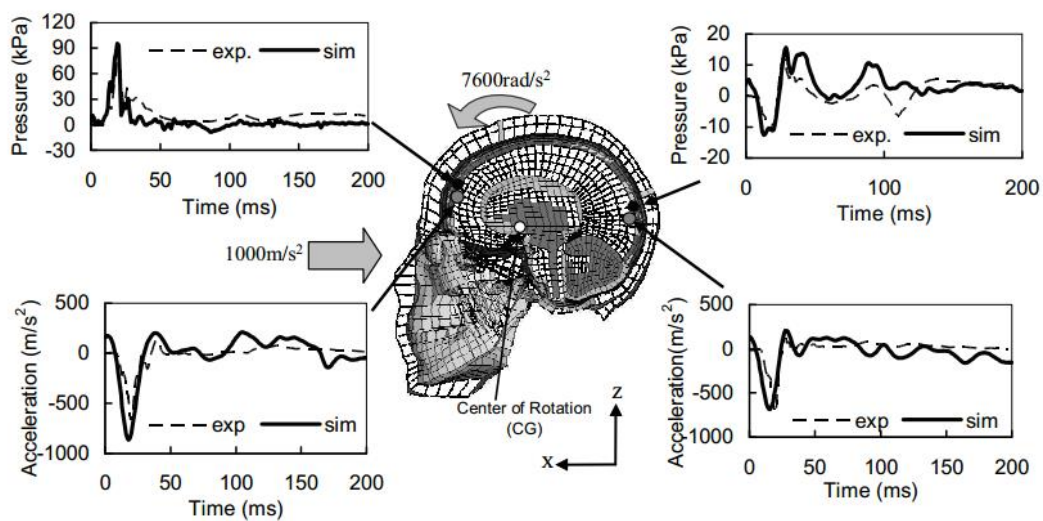


Figure 50. Set up and results of the validation phase of THUMS model

The comparisons for the validation were conducted in two regions of the brain, frontal and occipital, and with two parameters, linear acceleration and internal pressure. Figure shows the time history curves of these parameters, where solid lines represent predicted results, while dashed lines represent experimental data. The comparisons showed a reasonable agreement, with a high percentage of correlation.

The ULP FEM Human Head

The University Louis Pasteur of Strasbourg (ULP) developed a FEHM based on anatomical designs of the human head. The model was composed by 13208 elements and 2813 shell elements, for a total weight of 4.8kg, designed by the Hypermesh software [46].

The ULP FEM of the human head was a complex model that contained details of the whole brain, cerebellum, CSF, and layers surrounding each of these parts; it was used to calculate the intracranial pressure of the brain, brain Von Mises stress, global strain, energy of the brain/skull interface, and HIC.

The different parts and the characteristics of the materials are summarized in Table 19.

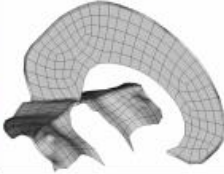
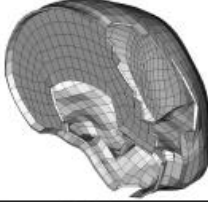
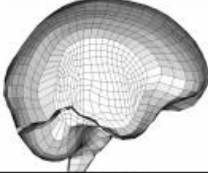
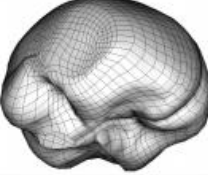
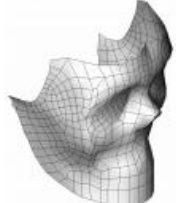
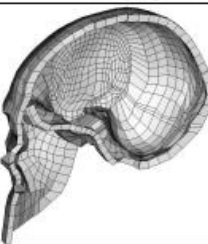
Anatomical Segment	Illustration	Mesh	Mechanical Behaviour	Mechanical Characteristics	Mechanical Characteristics
Falx of the Brain and Tentorium of the Cerebellum		471 shell elements	Linear Elastic	$e = 1 \text{ mm}$ $\rho = 1140 \text{ kg/m}^3$ $E = 31.5 \text{ MPa}$ $\nu = 0.45$	/
Brain/skull Interface		2591 brick elements	Linear Elastic	$\rho = 1040 \text{ kg/m}^3$ $E = 0.012 \text{ MPa}$ $\nu = 0.49$	/
Brain and Cerebellum		5508 brick elements	Elastic Plastic	$\rho = 1040 \text{ kg/m}^3$ $K = 1125 \text{ MPa}$ $G_0 = 0.049 \text{ MPa}$ $G_{inf} = 0.0167$ $\beta = 145 \text{ s}^{-1}$	/
Skull		1813 three layered composite shell elements	Elastic Plastic Brittle	Cortical $e = 2 \text{ mm}$ $\rho = 1900 \text{ kg/m}^3$ $E = 15000 \text{ MPa}$ $\nu = 0.21$ $K = 6200 \text{ MPa}$ $UTS = 90 \text{ MPa}$ $UTC = 145 \text{ MPa}$	Trabecular $e = 3 \text{ mm}$ $\rho = 1500 \text{ kg/m}^3$ $E = 4600 \text{ MPa}$ $\nu = 0.05$ $K = 2300 \text{ MPa}$ $UTS = 35 \text{ MPa}$ $UTC = 28 \text{ MPa}$
Face		529 shell elements	Linear Elastic	$e = 10 \text{ mm}$ $\rho = 2500 \text{ kg/m}^3$ $E = 5000 \text{ MPa}$ $\nu = 0.23$	/
Skin		2296 brick elements	Linear Elastic	$\rho = 1000 \text{ kg/m}^3$ $E = 16.7 \text{ MPa}$ $\nu = 0.42$	/

Table 19. Material properties and mesh characteristics of the parts of ULP model

The advantage of the model was a particularly efficient meshing, which allowed to perform simulations of up to 150 ms [47] (defined as 'long term' in this field). The defect was that for an impulse of this duration the effect of the damping of the impact of the neck became relevant, but the neck was not present in the model.

The validation phase was conducted through a comparison with the results reported by Turquier et al. [48]. The experiments were performed with an impactor of 23.4 kg, with an initial speed of 7 m/s, the same set-up of the experiments conducted by Nahum in 1977. The accelerations in the frontal and in the occipital regions were compared, as well as the internal pressures in frontal, occipital and in lateral regions, showing a good agreement (Figure 51).

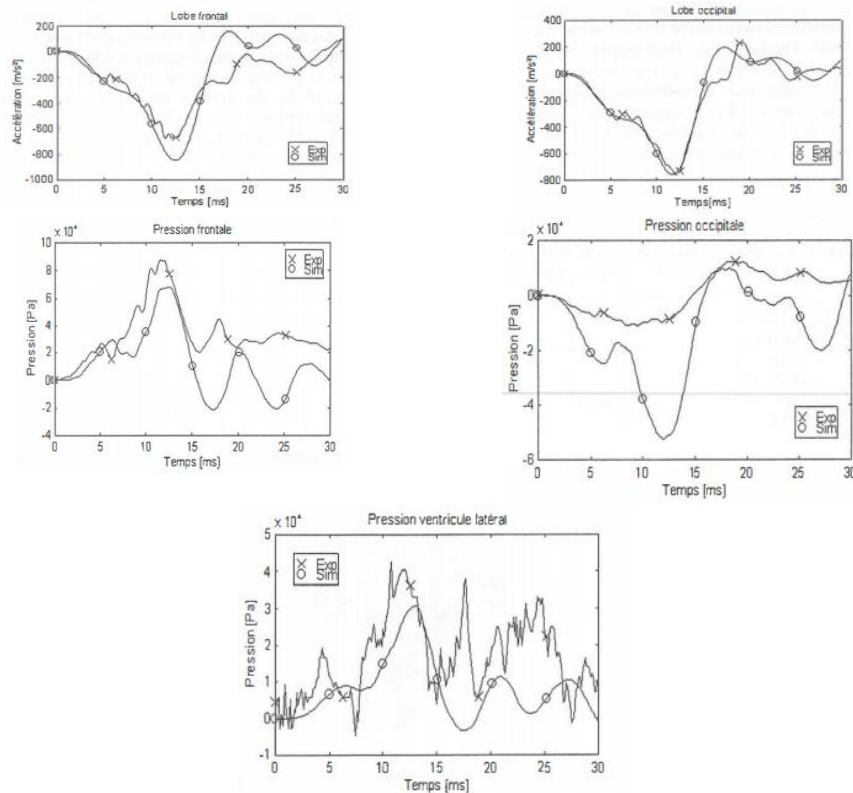


Figure 51. Validation results of the ULP model

Politecnico di Torino

In 2005 the professor Belingardi et al. [49], for the Politecnico di Torino, designed a FEHM with a geometry totally based on CT (160 scans corresponding to sections at a 1.25mm distances of a 31-year-old man) and MRI scanner images. The CAD phase was managed inside the softwares PARASOLID and Hypermesh 5.1, while the simulations were conducted with the software LS-DYNA.

The components of the model were, from outside to inside: scalp, compact bone, cancellous bone, facial bone, dura mater, falx, tentorium membranes, CSF, brain tissue and ventricles. The model contained 55264 elements and about 26000 nodes in total (Figure 52 and Figure 53).

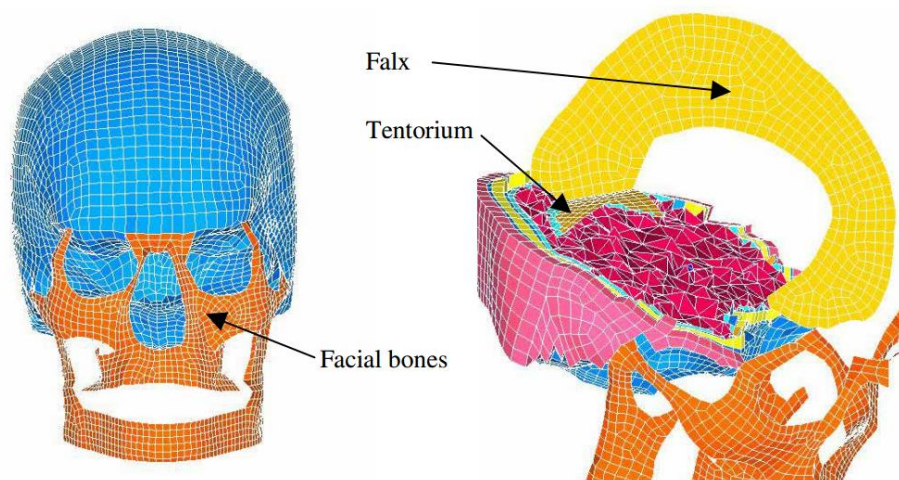


Figure 52. Frontal and perspective view of the Politecnico di Torino head model

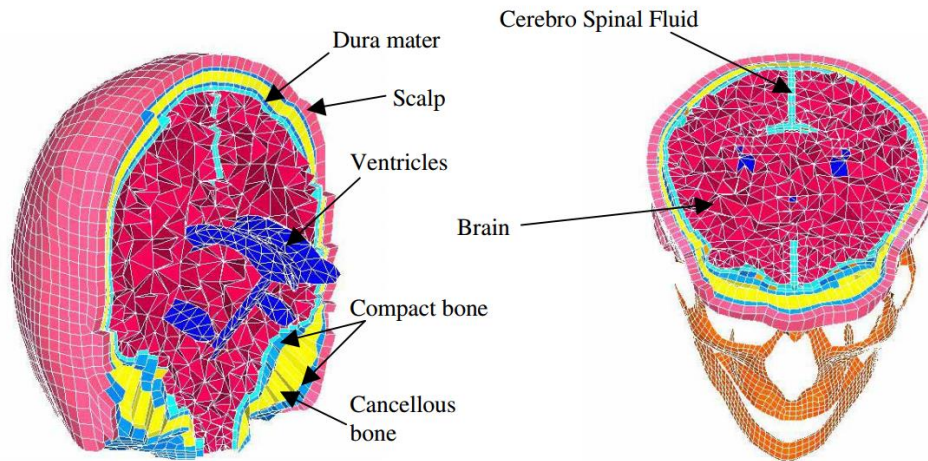


Figure 53. Coronal and axial section of the Politecnico di Torino head model

The mechanical properties of the materials were taken from literature and from previous FEHM: all the parts were modelled as a linear-elastic material, apart from the brain. In fact, the brain was modelled as a visco-elastic material, obtaining 1.4kg as weight of the brain. The characteristics of the materials are showed in Table 20.

Tissue	Material model	ρ (kg/m ³)	E (MPa)	ν
Compact bone	Linear elastic	1800	15000	0.21
Cancellous bone	Linear elastic	1500	4500	0.01
Facial bone	Linear elastic	4500	10000	0.3
Brain	Visco-elastic	1140		
CSF	Linear elastic	1040	0.012	0.49
Ventricles	Linear elastic	1040	0.012	0.49
Scalp	Linear elastic	1200	16.7	0.42
Dura mater	Linear elastic	1133	31.5	0.45
Tentorium	Linear elastic	1133	31.5	0.45
Faulx	Linear elastic	1133	31.5	0.45

Table 20. Characteristics of the materials of the Politecnico di Torino head model

The authors chose to keep the head model free from the neck, considering the impact too fast to be influenced by the constraint between neck and head.

The validation phase consisted in the comparison of the internal pressure trends of the results obtained by Nahum in 1977. The frontal region and the posterior fossa were the two areas considered for the

comparisons (Figure 54). The results obtained showed a reasonable agreement, above all in the frontal region.

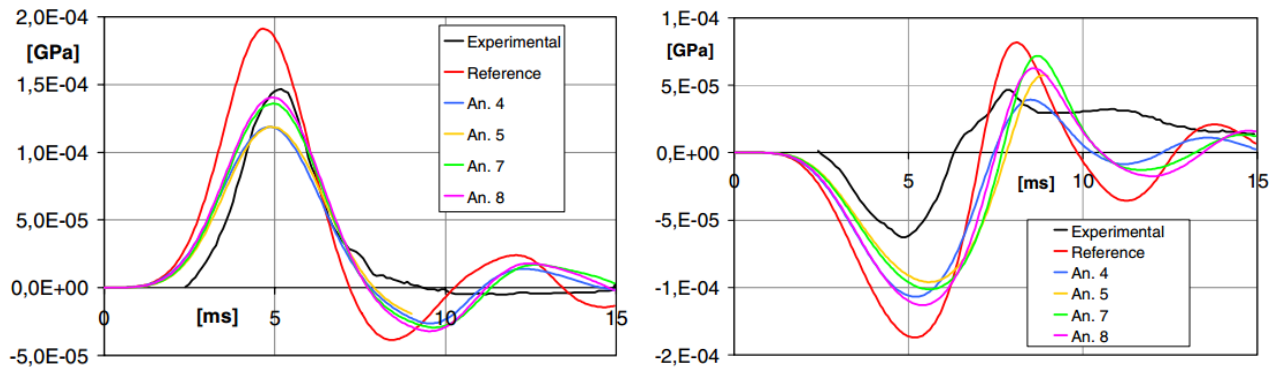


Figure Frontal pressure trend

Figure Posterior fossa pressure trend

Figure 54. Results of the comparison of the Validation phase of the Politecnico di Torino head model

Harvard Medical School

The Harvard FEHM was born from the collaboration between the Bioengineering Laboratory of the University of Salerno and the Harvard Medical School. The geometric basis of the model derived from Harvard's proprietary axial MRI images, contained in its anatomical atlases ('The Whole Brain Atlas') [50]. The software Mimics was used for the phase of image processing, necessary to obtain the 3D CAD.

The model was composed by the following components (Figure 55): (1) skull, (2) cerebrospinal fluid (CSF), (3) gray matter, (4) white matter, (5) cerebellum, (6) corpus callosum, (7) telencephalic nuclei, (8) brain stem and (9) ventricles, for a total of 39047 tetrahedral composite element. The level of details was comparable to the WSUHIM. The model did not contain the facial bones and the CSF was in the form of a 3-mm thick layer.

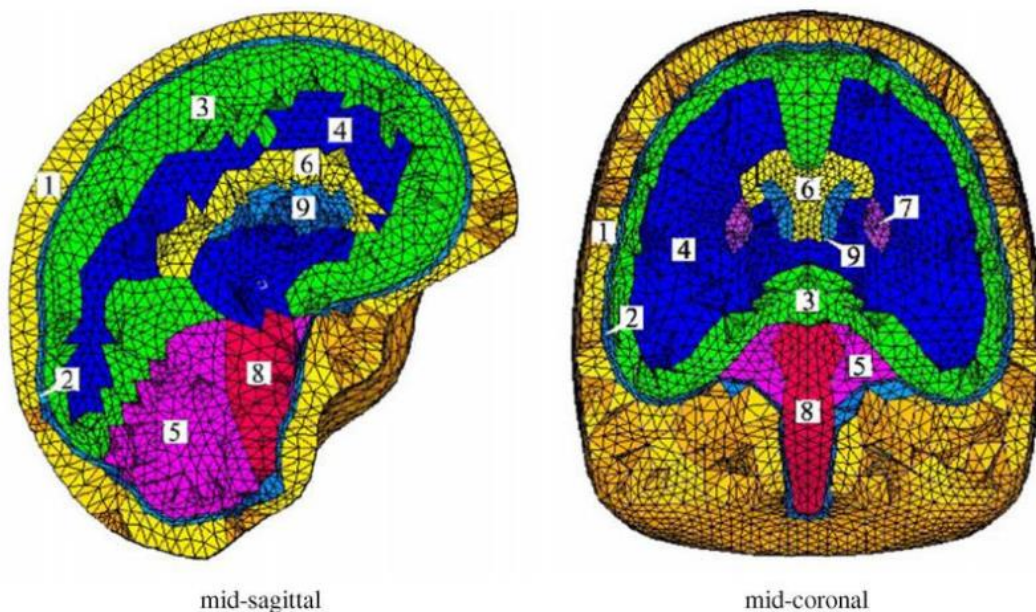


Figure 55. Sagittal and coronal sections of the Harvard FEHM

The brain inside the model weighted 1.4kg, with a volume of 1508 cm³, while the skull was 0.8kg for 678 cm³.

The characterization of the materials inside this model is the most complex: the model includes the linear elastic, the visco-elastic and the plastic behaviours, considering them in a thermodynamical point of view. All terms are summarized in the expression of free energy A:

$$A = W^e(e_j^e, \theta^e) + W^p(e^p, \theta^p) + \sum_{i=1}^M W_i^e(e_{ij}^e, \theta_i^e)$$

where i ($i=1 \dots M$) indicates the number of the viscoelastic element, W^e represents elastic strain energy densities, W^p represents the plastic stored energy, e^e is the elastic logarithmic shear strain, e^p the plastic logarithmic shear strain, θ^e is the elastic logarithmic volumetric strain, θ^p the plastic logarithmic volumetric strain. This law regulates the internal forces during the viscoelastic transient, when the thermodynamical equilibrium is reached the material behaviour is only elastoplastic.

The Harvard Medical School FEM was used for clinical purposes, both for the prediction of head injuries and for the detection of cavitations, physical phenomenon that occurs as a typical consequence of the coup–contrecoup injury (Figure 56). After a frontal impact, for example, a peak of positive pressure is observed in the hit region, or coup site, while, a valley of negative pressure is observed in the contrecoup site. These sudden changes in pressure, effect of the slapping of the skull hitting the brain, produce micro-bubbles of vapor inside the CSF (cavitation), that, when they collapse, they can cause damage to the local tissue.

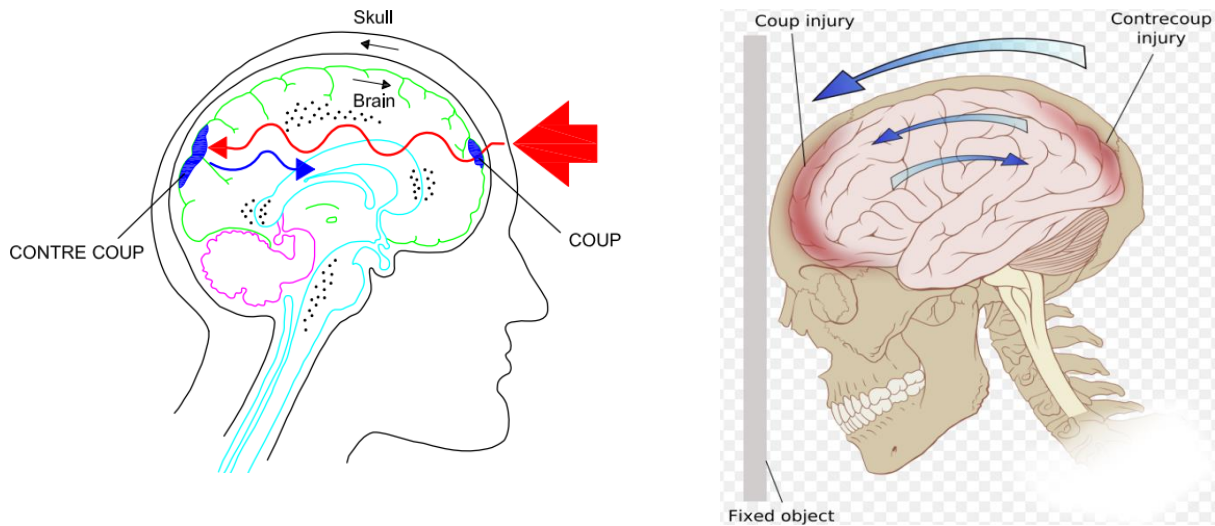


Figure 56. Coup–contrecoup injury

Shear stress contour, Intracranial pressure and HIC were the extractable parameters from this model. Moreover, the authors have developed a toolkit, starting from the processing of the aforementioned data, for the assessment of the damages caused by the cavitation.

The model was validated through the comparison with the results obtained by Nahum in 1977: the internal pressure was recorded in 3 different regions of the brain of the model, the frontal region, the posterior region and the parietal region, and the temporal trends were compared with the references inside the literature (Figure 57). In all 3 regions the correlation between model and reference was high.

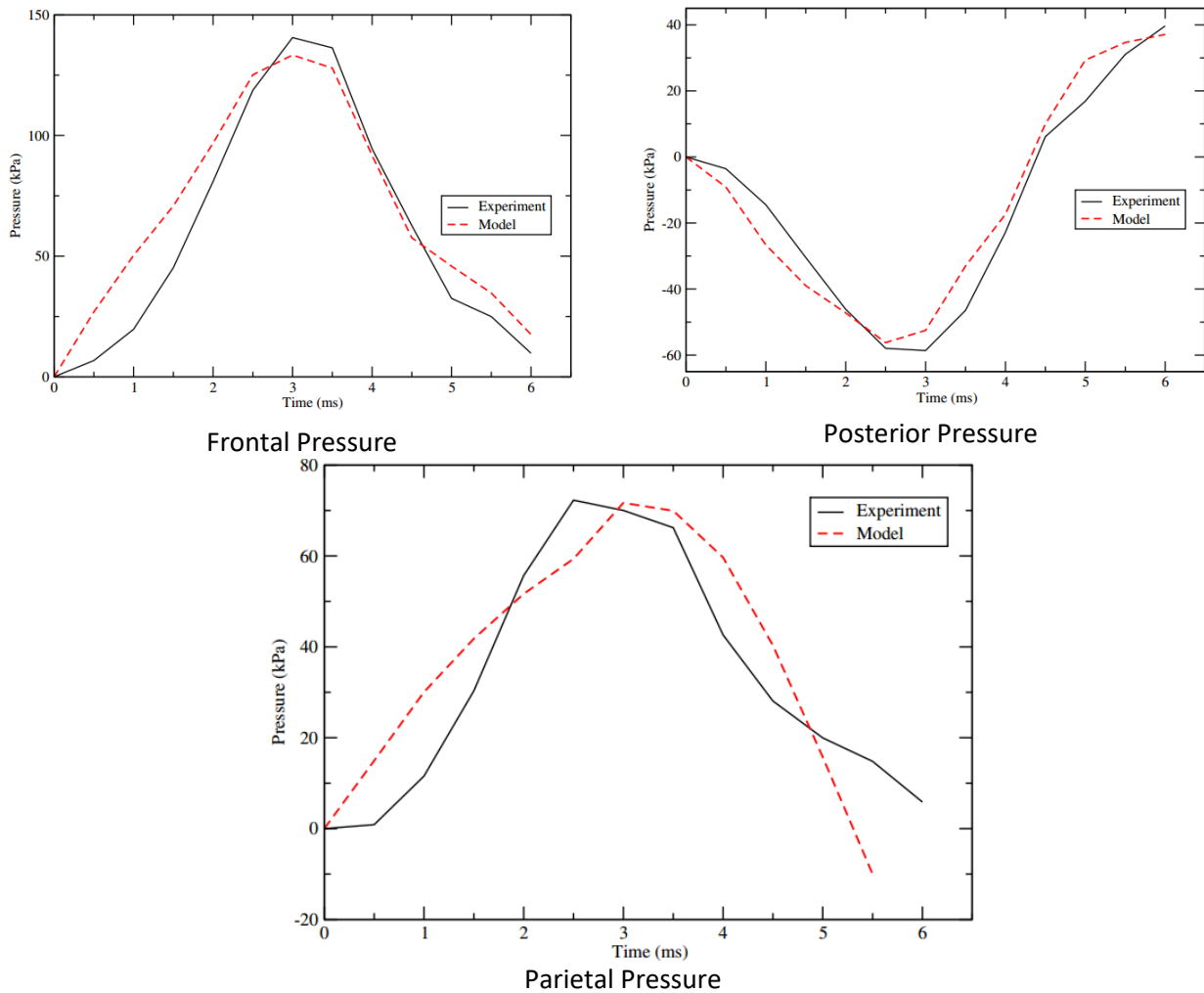


Figure 57. Results of the comparison of the Validation phase of the Harvard FEHM

Dublin College (UCDBTM)

In 2004 the Department of Mechanical Engineering of the University College Dublin developed a FEHM calling it UCDBTM [51], visible in Figure 58. The main parts of the model were: cerebrum, cerebellum, brainstem, intracranial membranes (falx and tentorium), pia, CSF layer, dura, a three-layered skull (cortical and trabecular bone layers) with varying thickness, scalp, and facial bone. 7318 hexahedral elements were used to model the brain, 2874 hexahedral to model the CSF. The brain weighted 1.42kg, for a total weight of the model of 4.01kg.

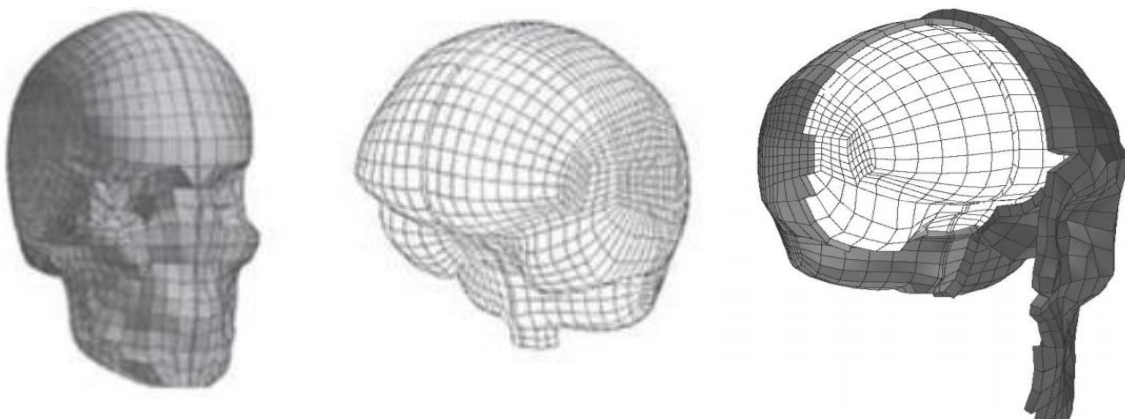


Figure 58. Three view of the UCDBTM

The software used to model the geometrical characteristics of the head-brain system was the commercial MSC/Patran. A phase of image processing of CT scans, through a VTK software, was at the base of the model [52].

All the parts inside the model were considered with a linear elastic behavior, except for the brain, which was considered with a hyper elastic behavior. The characteristics of the material used are shown in Table 21.

Material	Young's modulus (MPa)	Poisson's ratio	Density (kg/m ³)
Scalp	16.7	0.42	1000
Cortical Bone	15 000	0.22	2000
Trabecular Bone	1000	0.24	1300
Dura	31.5	0.45	1130
Pia	11.5	0.45	1130
Falx and Tentorium	31.5	0.45	1130
Brain	Hyper Elastic	0.499981	1040
Facial Bone	5000	0.23	2100

Table 21. Material properties of the UCDBTM

The simulation to validate the model, always with the same modalities of the experiments of Nahum and Trosseille, were conducted through the software Abaqus. The results of the validation were accepted by the scientific community.

From this model it was possible to extract the brain shear stress, the brain pressure, in certain regions, and the HIC. Moreover, it was possible to quantify the amount of the movement of the brain inside the skull [53].

ShockAccel FEHM

The anatomical basis of the model is exclusively geometric, unlike most models found in the literature, with a CAD drawn from scanned CT and/or MRI images of the brain. The reasons for this choice were many: first of all, the University (HES-SO of Fribourg) did not own CAD models of the human brain, the models found on the net, both free and paid ones, were difficult to adapt to the Abaqus software. Moreover, a lot of reticence was found in obtaining other models from other research institutes, not willing to sell their model. In conclusion, it was decided to design a simple and, at the same time, as accurate as possible model, calling it ShockAccel FEHM, trying to be accurate in the definition of materials, in the characterization of physical behavior and in the dimensions of the parts. The idea was therefore to have, at the end of the project, a model designed entirely by the student (with a view to a master's thesis) inexpensive and not very complex, as realistic as possible and with good computational time performances.

The Abaqus structure drives the user to work by modules, following a “waterfall decision tree”, without however removing the possibility of going back to the previous modules and making changes that affect the entire subsequent flow.

To create the model in Abaqus the user can proceed in two ways:

- following the model tree (on the left of the screen, red arrow in Figure 59);
- following the modules (above the blue screen, red arrow in Figure 59).

Following one of the two alternatives in the order proposed by Abaqus allows the user to minimize forgetfulness.

When the user accesses a module, Abaqus provides a toolkit (green box in Figure 59), different for each module.

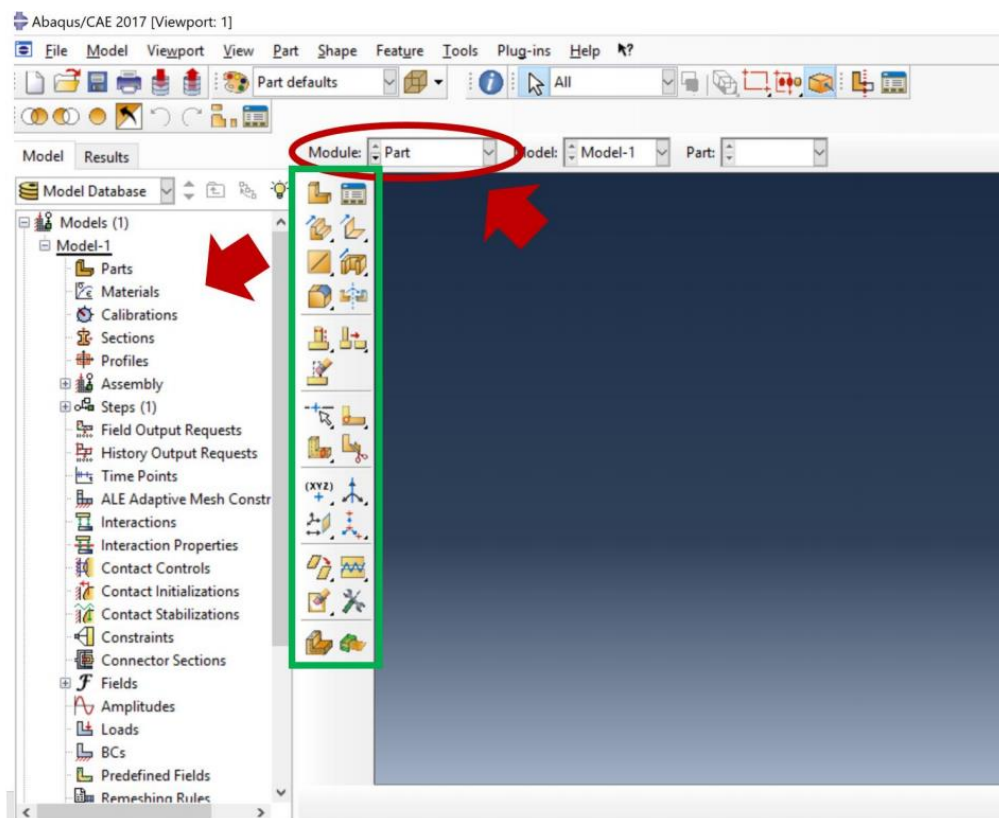


Figure 59. Abaqus Environment

In the latest version, Abaqus provides eleven modules:

1. Part module;
2. Property module;
3. Assembly module;
4. Step module;
5. Interaction module;
6. Load module;
7. Mesh module;
8. Optimization module;
9. Job module;
10. Visualization module;
11. Sketch module.

During the project, accessing all 11 modules was not necessary. The order just shown is the one recommended by Abaqus, and the most correct to follow during the design phase. During this project this was the order followed, but within this paper the order of presentation will be different, to facilitate the logic of the reader's understanding.

Part Module

The part module is a CAD tool, easier than other CAD software in the market, where the user can draw the parts (components) of his model. The model of the project was composed by 10 parts:

1. Left_hemisphere;
2. Right_hemisphere;
3. Corpus_callosum;
4. Cerebellum;
5. Cerebrospinal Fluid (CSF);
6. Column (Brainstem);
7. Neck;
8. Skull;
9. Padding;
10. Helmet

Abaqus software assumes that the user defines consistent units of measurement. The match between different units is figured in Table 22. The second column, SI (mm), was adopted in the project.

Quantity	SI	SI(mm)	SI	US Unit(ft)	US Unit(inch)
Length	<i>m</i>	<i>mm</i>	<i>m</i>	<i>ft</i>	<i>in</i>
Force	<i>N</i>	<i>N</i>	<i>kN</i>	<i>lbf</i>	<i>lbf</i>
Mass	<i>kg</i>	<i>tonne (10³kg)</i>	<i>tonne</i>	<i>slug</i>	<i>lbf s²/in</i>
Time	<i>s</i>	<i>s</i>	<i>s</i>	<i>s</i>	<i>s</i>
Stress	<i>Pa (N/m²)</i>	<i>MPa (N/mm²)</i>	<i>kPa</i>	<i>lbf/ft²</i>	<i>psi (lbf/in²)</i>
Energy	<i>J</i>	<i>mJ (10⁻³J)</i>	<i>KJ</i>	<i>ftlbf</i>	<i>inlbf</i>
Density	<i>kg/m³</i>	<i>tonne/mm³</i>	<i>tonne/m³</i>	<i>slug/ft³</i>	<i>lbf s²/in⁴</i>

Table 22. Unit protocols accepted by Abaqus

In this chapter each part will be briefly described, an image of it and its corresponding meshed version will be shown. In addition, the sketch and a table will summarize the geometric characteristics of each part.

Left_Hemisphere

The left and right hemispheres were drawn identical. No differentiation was made between white matter and gray matter. This is already a first important choice: not considering the white matter, and therefore ignoring the presence of axons, the model proposes a completely homogeneous view of the human brain. The literature [58] has shown how the direction of the axons, that make up the neural network, has a strong influence on the mechanics of the shock on the head. In order to properly analyze the deformations suffered by the axons, a very robust imaging base (Diffusion tensor imaging, or DTI) was necessary, but it was not involved in this project, as already explained. Moreover, the gyri and sulci typical of gray matter are not present.

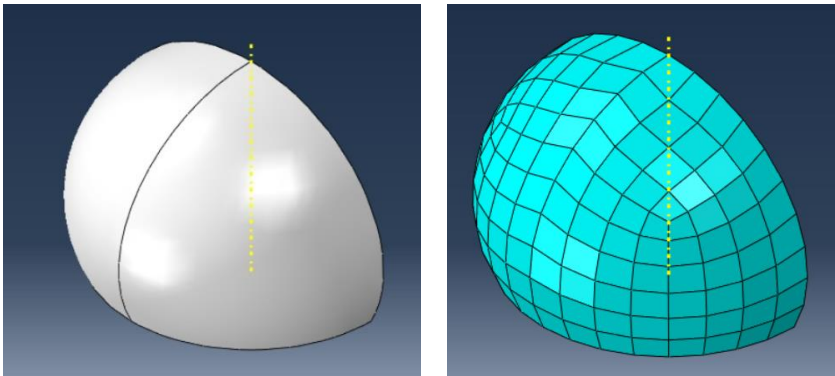


Figure 60. Left_Hemisphere: smooth and meshed version

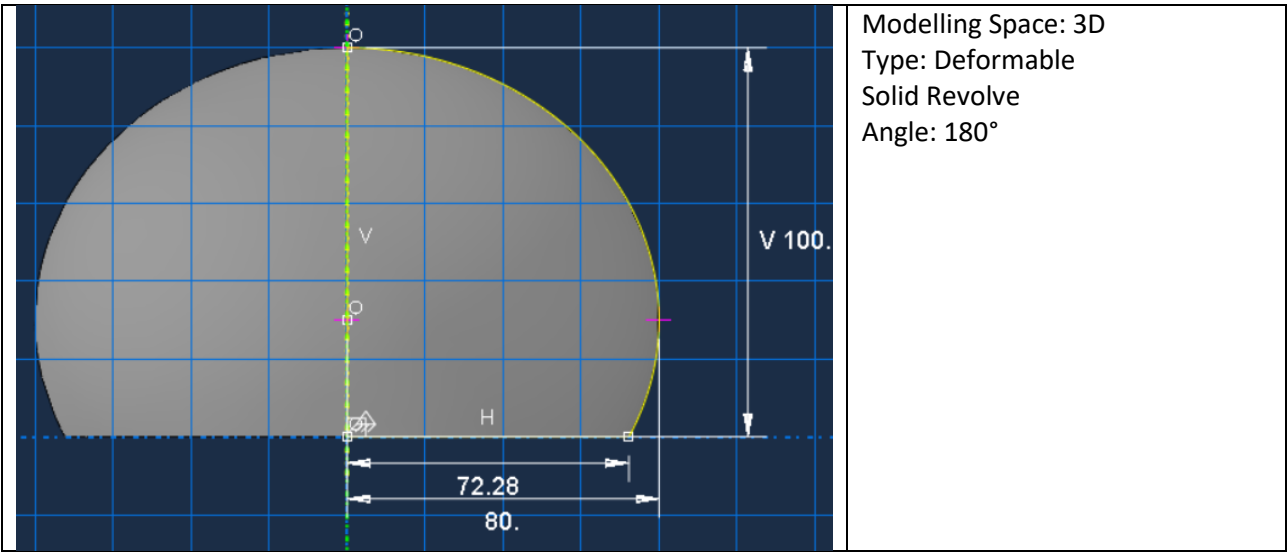


Table 23. Left_Hemisphere: geometric characteristics

Corpus_Callosum

Despite the lack of differentiation between white matter and gray matter, the modeling of the corpus callosum (white matter) was considered necessary, approximating it as a contact component between the two hemispheres.

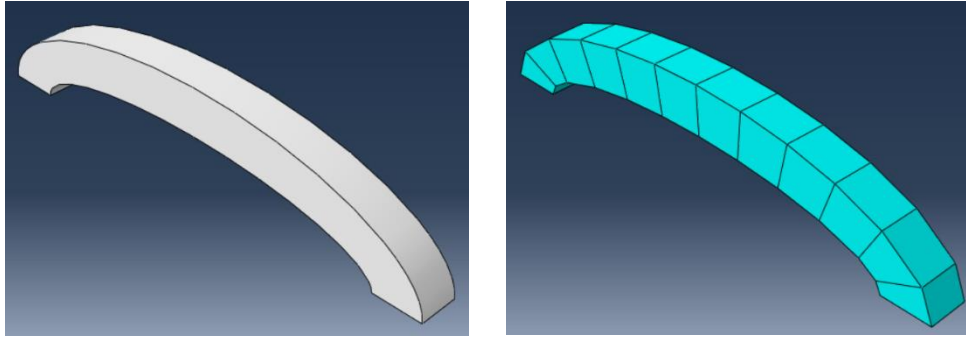


Figure 61. Corpus_Callosum: smooth and meshed version

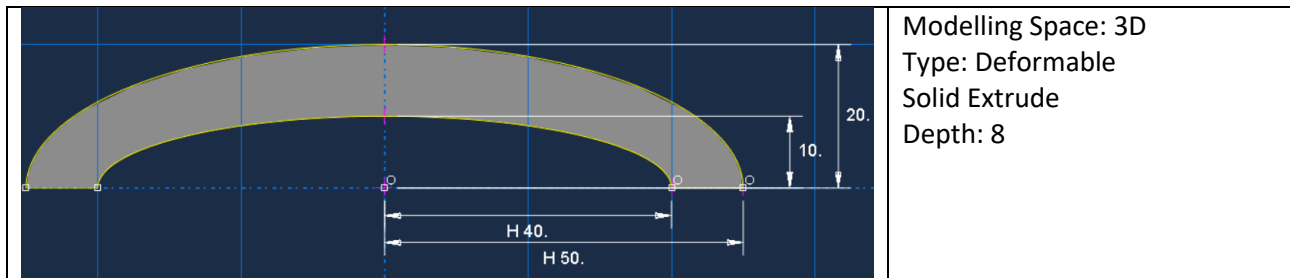


Table 23. Corpus_Callosum: geometric characteristics

Cerebellum

The cerebellum is a component present in nearly all FEHMs found in the literature. It was approximated with a very simple and symmetrical geometry, and placed at the base of the two hemispheres, under their occipital region.

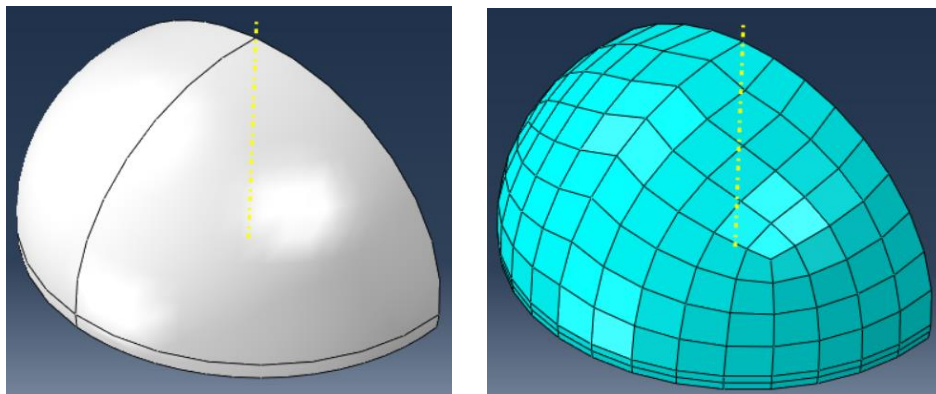


Figure 62. Cerebellum: smooth and meshed version

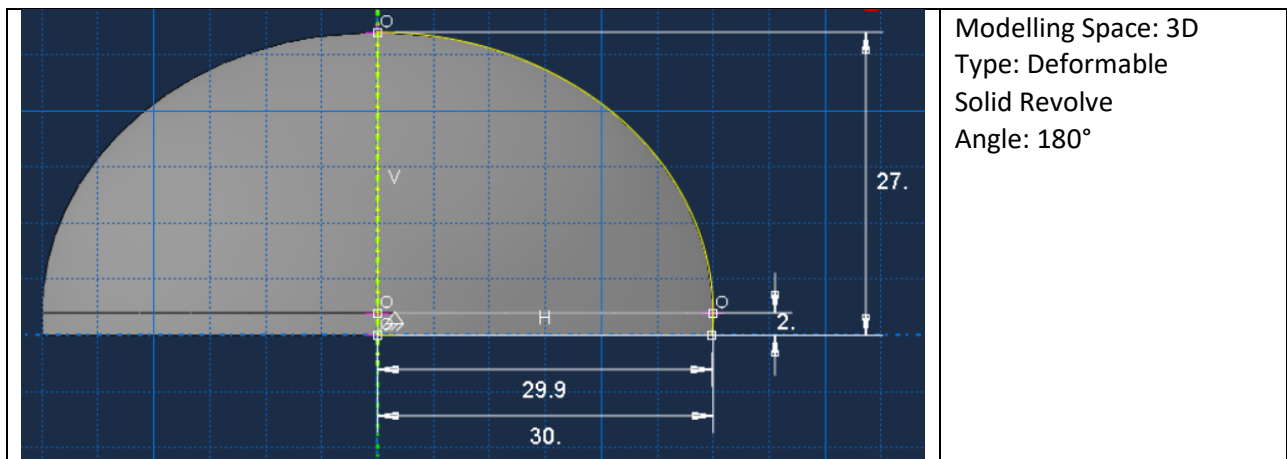


Table 24. Cerebellum: geometric characteristics

CSF

The CSF part was obtained through a cutting operation of the space inside the skull. As in sculpture, the starting 'block' was an ovoidal object, of the same size as the empty space inside the skull: from that block all the spaces that should subsequently contain the internal organs (the two hemispheres, the corpus callosum, the cerebellum and the brainstem) were eliminated.

The resulting geometric structure was the most complex, so much that it was the only part with a Tet (tetrahedral), and not Hex (hexahedral) element shape. Moreover, deriving from a cutting operation, there is no sketch of this component.

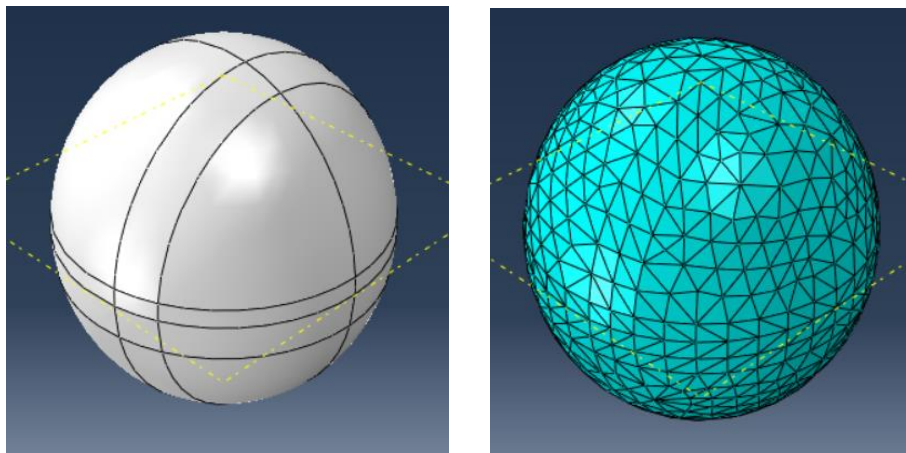


Figure 63. CSF: smooth and meshed version

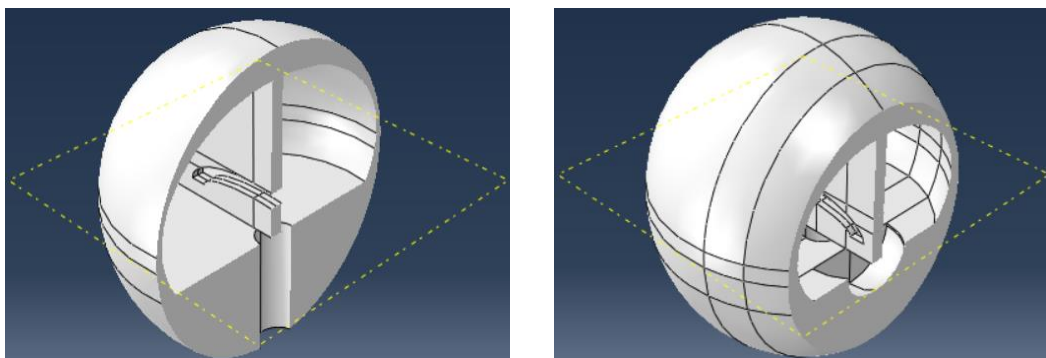


Figure 64. CSF: two frontal section of the part. It is possible to notice the gaps which the internal organs will fill: the two hemispheres, the cerebellum, the corpus callosum and the brainstem

Column

The column part has been modeled as a simple cylinder that "pierces" the skull parallel to its axis of symmetry, at a point a few mm behind its lower pole. Anatomically, this component represents, as a single body, the brainstem and the first part of the spinal cord, till to longitudinally cross the neck.

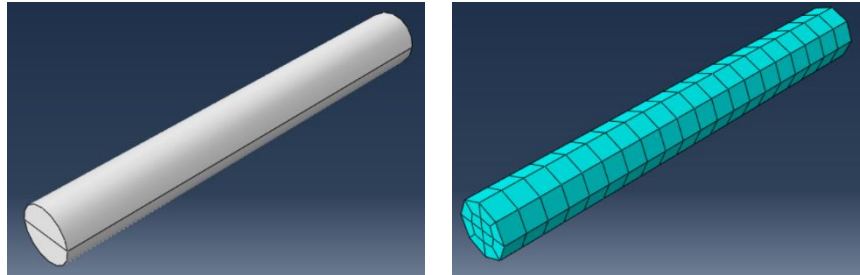


Figure 65. Column: smooth and meshed version

	<p>Modelling Space: 3D Type: Deformable Solid Extrude Depth: 200</p>
--	--

Table 25. Column: geometric characteristics

Neck

The neck part was originally designed as a cylindrical crown, inside which the vertebral column and therefore the brainstem and the spinal cord are placed. Subsequently, it was subjected to a cutting operation on its upper surface with the skull part, so as to allow the head to lean on the neck.

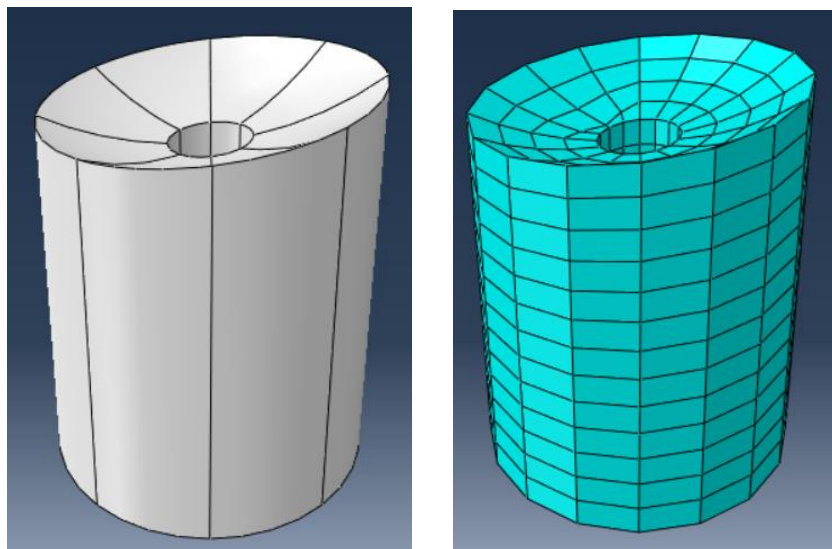


Figure 66. Neck: smooth and meshed version

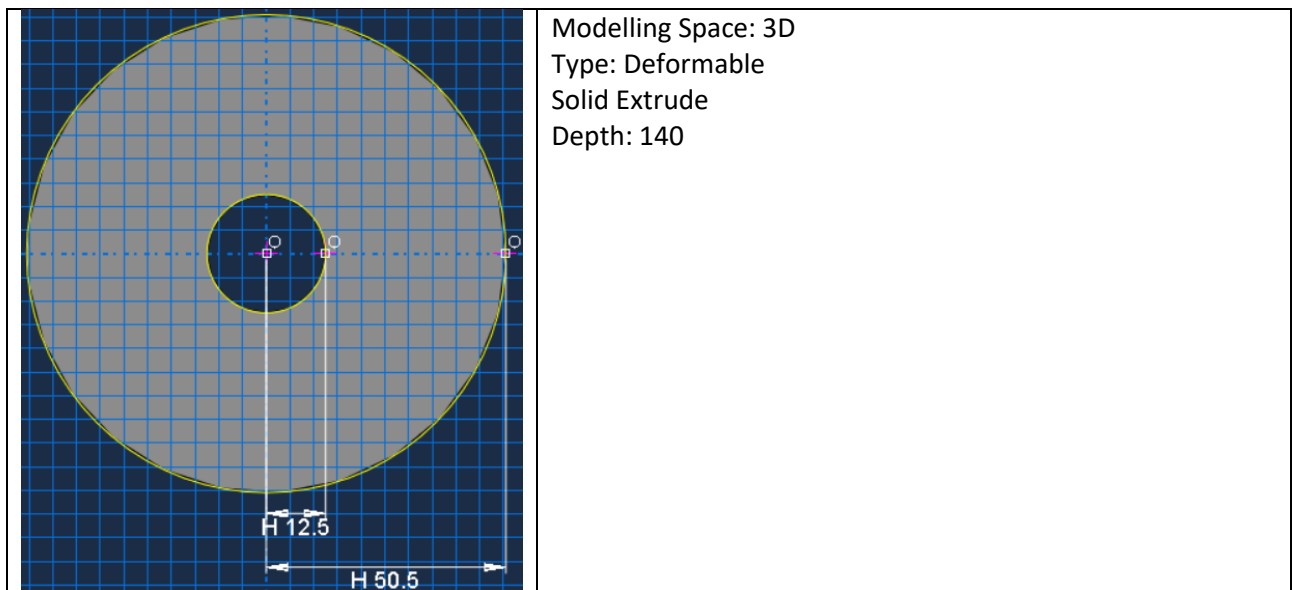


Table 26. Neck: geometric characteristics

Skull

The skull was modeled as a hollow ovoidal layer, which would then contain all the internal organs. Later it was "pierced", in a cutting operation, by the column part, a few mm behind the lower pole of the ovoid.

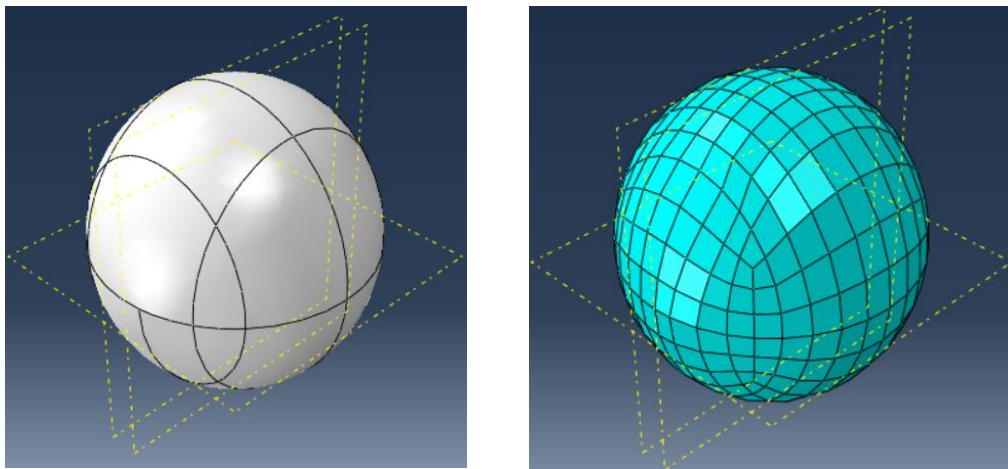


Figure 67. Skull: smooth and meshed version

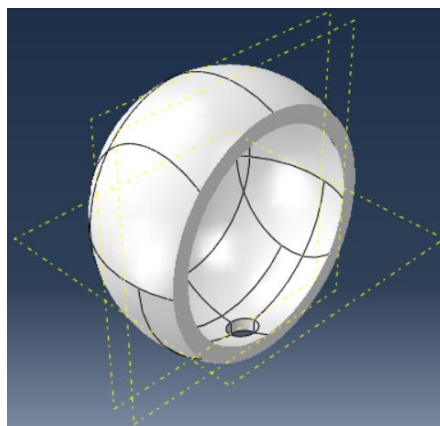


Figure 68. Skull: a frontal section of the part. It is possible to notice the gaps which the CSF and the brainstem will fill

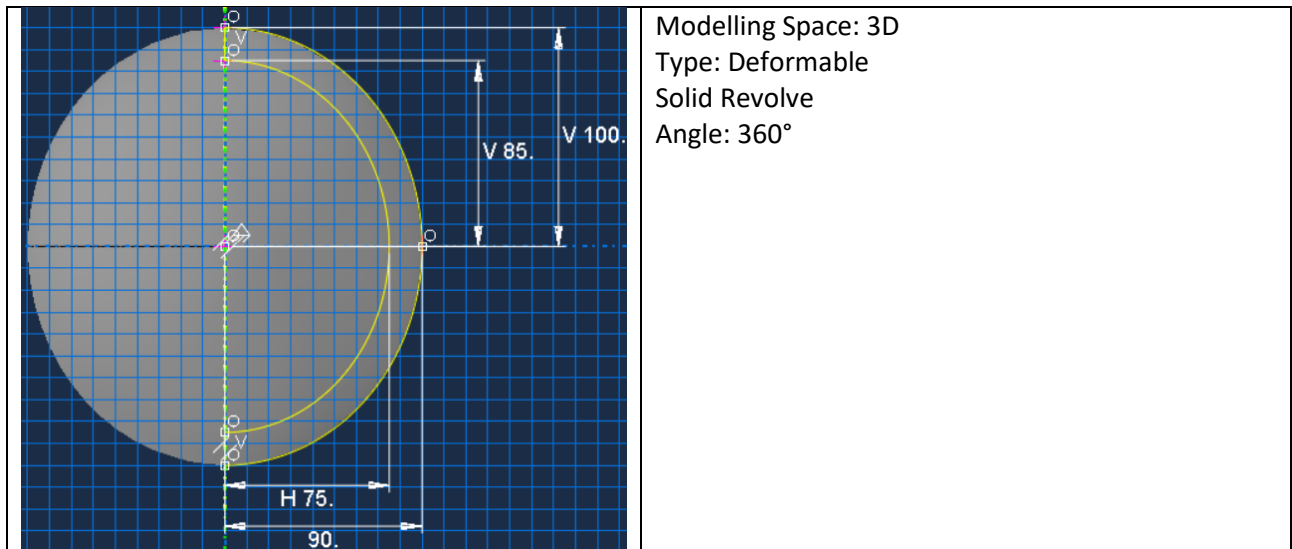


Table 27. Skull: geometric characteristics

Padding

The helmet designed in this project was represented in the simplest possible form: 2 components, the inner shell (called Padding in this model) and the outer shell (called Helmet), commons to all helmets on the market, for different types of sport competitions and for motorcycle use. The aerodynamic or aesthetic aspects, nor the characteristic shapes that distinguish a hockey helmet, compared to an American football or motorcycling one, were not considered.

The geometry of the padding layer was obtained simply by covering the shape of the skull.

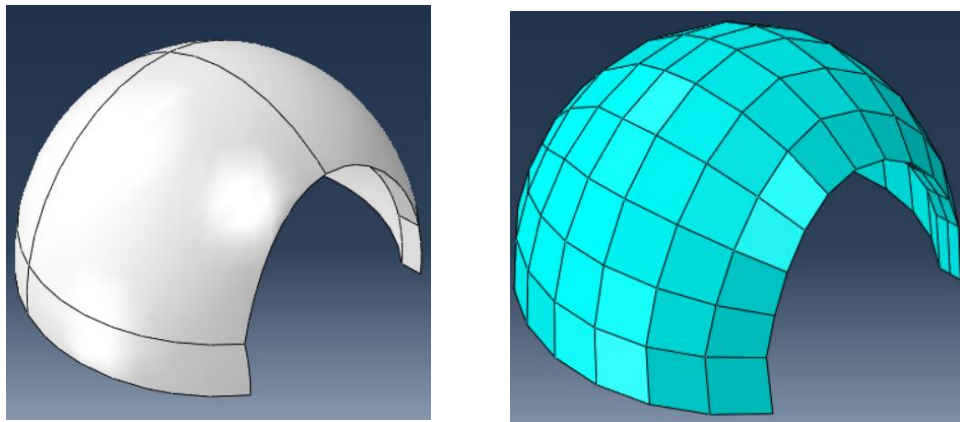


Figure 69. Padding: smooth and meshed version

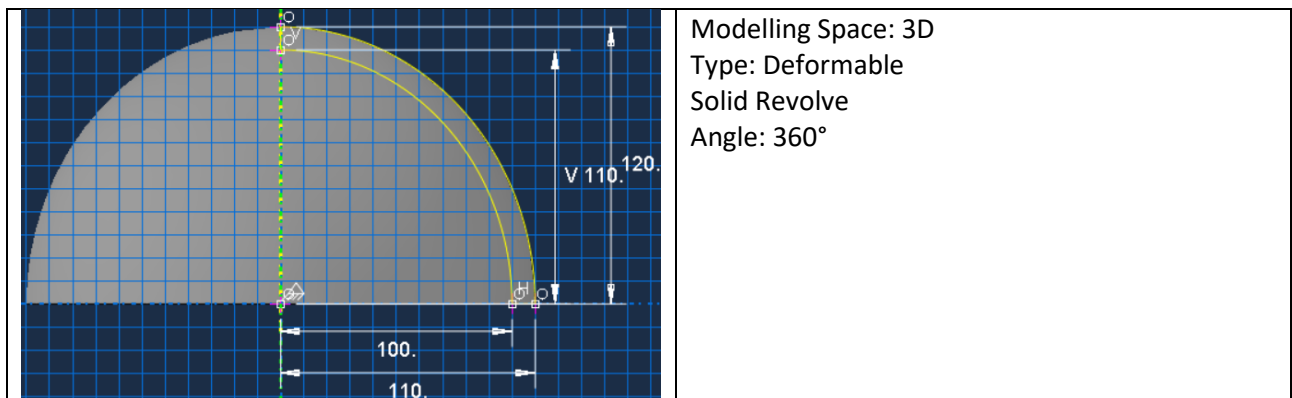


Table 28. Padding: geometric characteristics

Helmet

The geometry of the helmet layer was obtained simply by covering the shape of the padding.

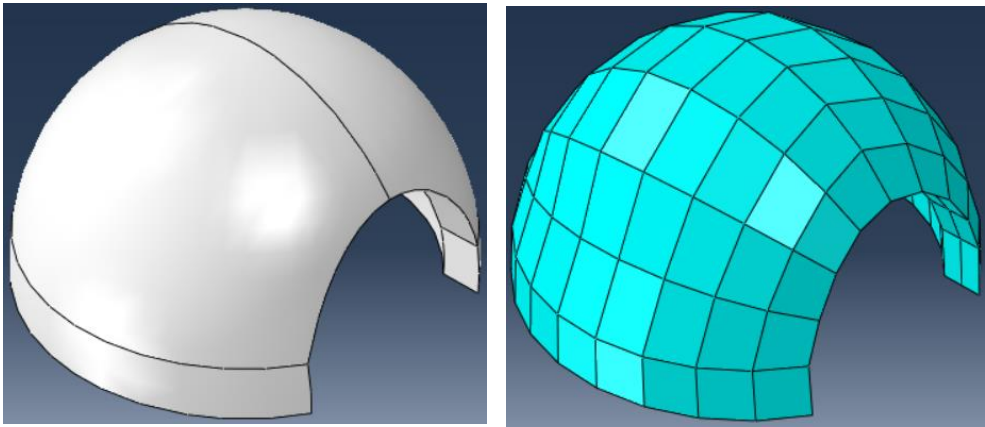


Figure 70. Helmet: smooth and meshed version

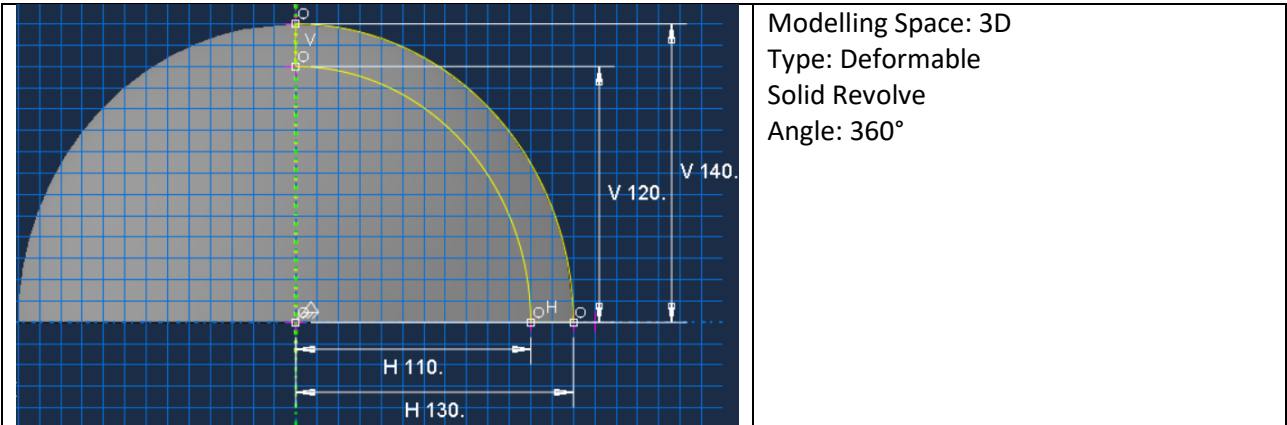


Table 29. Helmet: geometric characteristics

The helmet-padding structure covers the upper 50% of the ellipsoid representing the head. A cutting operation (Figure 71) with an eroding object allowed to create the space in which to place the visor of the helmet. The visor was not inserted in the model, its mechanical relevance during the impact was not considered.

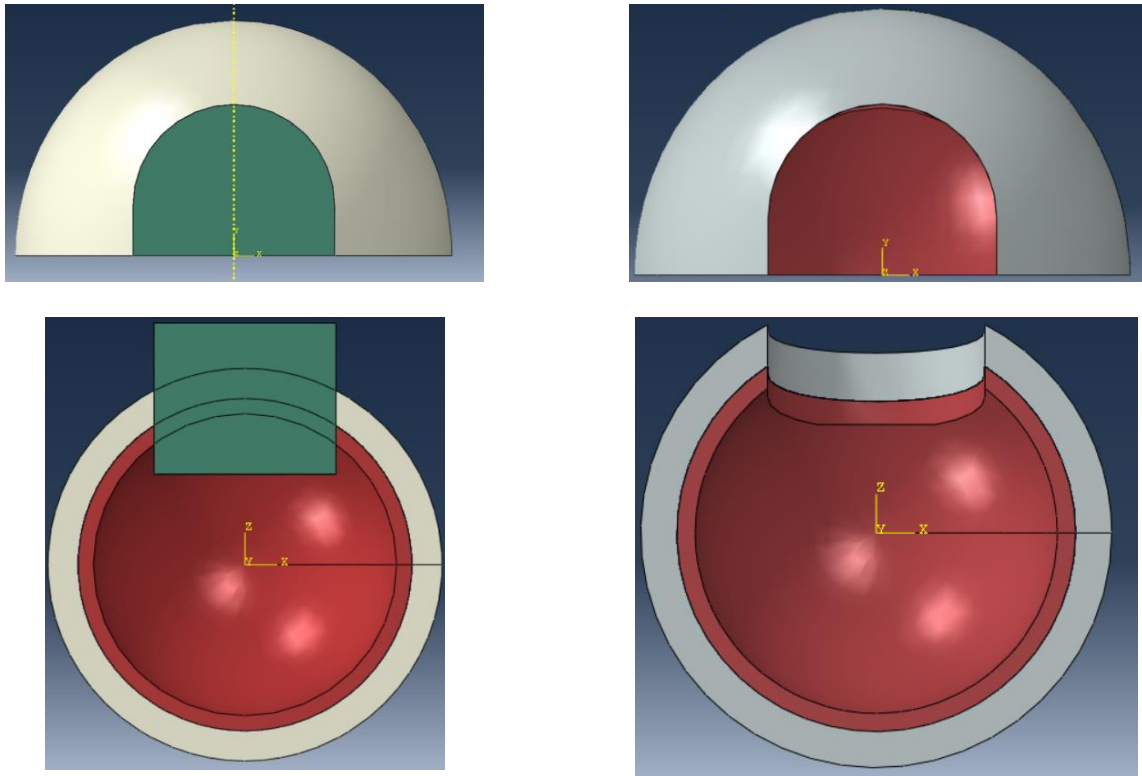


Figure 71. Cutting operation of the two layers of the helmet: the first row represents the frontal view, the second row represents the bottom view. The first column represents the pre-cutting structure, the second column represents the post-cutting structure

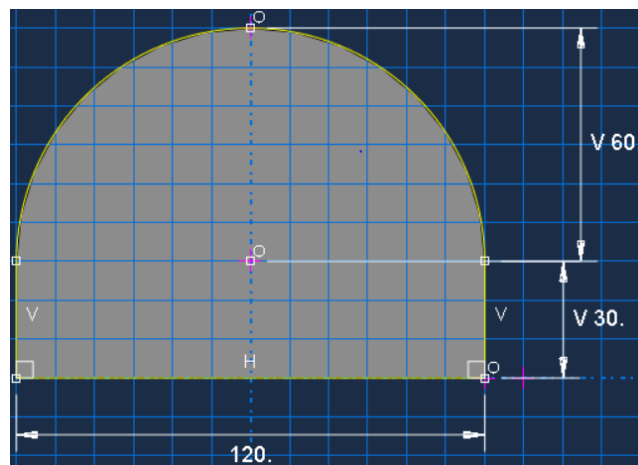


Figure 72. Geometrical characteristics of the object (green element in Figure 71) used to obtain the space for the visor

Property Module

In this module the material properties of the parts created in the previous module are defined. The characteristics of the materials adopted inside the model were 7:

1. Bone_material;
2. Brain_material;
3. CSF_material;
4. Corpus_callosum_material;
5. Neck_material;
6. Padding_material (EPS20);

7. Helmet_material (Polycarbonate).

To create a new Property, so to define the properties of the material to assign to the parts, Abaqus requires user to select a specific Material behavior, which can belong to various domains of physics, such as Mechanics, Thermodynamics, Electromagnetism, and the General domain and the domain called Other. For example, the material behavior assigned to almost all of the materials defined in each Abaqus model is the density, which belongs to the General domain.

Bone_material was the material assigned to the Skull. It was characterized by two material behaviors: density ($d = 2.12\text{E-}09$), and elastic ($E = 15000$, $\nu = 0.21$).

Brain_material was the material assigned to the Left_Hemisphere, Right_Hemisphere, Cerebellum and Column. It was characterized by three material behaviors: density ($d = 1.14\text{E-}09$), elastic ($E = 31.5$, $\nu = 0.45$) and viscoelastic (Prony Series).

CSF_material was the material assigned to the CSF. It was characterized by three material behaviors: density ($d = 1.15\text{E-}09$), Equation of State (Us-Up) and viscosity (Dinamic_Viscosity = $2.5\text{E-}07$).

Corpus_callosum_material was the material assigned to the Corpus_callosum. It was characterized by three material behaviors: density ($d = 1.14\text{E-}09$), elastic ($E = 31.5$, $\nu = 0.45$) and viscoelastic (Prony Series).

Neck_material was the material assigned to the Neck. It was characterized by two material behaviors: density ($d = 1.18\text{E-}09$), and elastic ($E = 6040$, $\nu = 0.38$) [54].

The insertion of the neck part in the model was necessary to have a damping effect following the shock suffered by the head. A free motion of the head, without friction or constraints, as a result of the input impact would not have been realistic, and therefore there was a need for a component that would dampen the shock, as the neck, integrated with a boundary condition, the encaster in the bottom surface.

The characterization of the neck was not easy: it is a highly heterogeneous anatomical part, it contains muscles, tendons and bones (vertebral column), all with very different material characteristics. Saboori et al. [54] were able to extrapolate an estimate of the properties of the materials that compose the neck, as if it were a homogeneous tissue, except for an internal corridor through which the spinal cord passes, which is the same approximation made on the model of this project.

Padding_material was the material assigned to the Padding. It was characterized by three material behaviors: density ($d = 2\text{E-}11$), elastic ($E = 11.5$, $\nu = 0.11$) and plastic (Yield stress vs Plastic strain). The material chosen, according to literature, for this component was Expanded Polystyrene (EPS).

The padding layer is the part that plays the most important role in protecting the head from impact. During the production phase of the helmet, the padding is made of particular materials (the most commonly used is EPS) which deforming absorb the energy of the impact preventing it from spread to the skull and brain. In the most critical areas that need more protection of the helmet, the density of the material is varied ensuring better absorption of the impact: this improvement was not imported into the realization of the project model.

It is also the most vulnerable layer of the helmet and it is therefore necessary to pay attention to helmet storage while not in use. The stress-strain diagram of EPS, from which the elasticity and plasticity parameters shown in the following tables are extracted, is represented in Figure 73.

Helmet_material was the material assigned to the Outer shell. It was characterized by three material behaviors: density ($d = 1.2\text{E-}09$), elastic ($E = 2300$, $\nu = 0.32$) and plastic (Yield stress vs Plastic strain). The material chosen, according to literature, for this component was Polycarbonate.

The outer shell is the most external part of the helmet, exposed to view. It is made with very resistant materials such as fiber-reinforced composite materials or with thermoplastic materials, because its main function is that of withstand the impact, preventing any foreign body from penetrating inside the helmet, it also has the task of redistributing the impact force over the entire surface of the helmet without breaking, however its presence alone is not enough to have complete protection.

The stress-strain diagram of Polycarbonate, from which the elasticity and plasticity parameters shown in the following tables are extracted, is represented in Figure 74.

ELASTIC BEHAVIOR	Density (10^3kg/mm^3)	E (MPa)	ν
Bone_material	2.12E-09	15000	0.21
Brain_material	1.14E-09	31.5	0.45
CSF_material	1.15E-09	/	/
Corpus_callosum_material	1.14E-09	31.5	0.45
Neck_material	1.18E-09	6040	0.38
Padding_material (EPS20)	2,00E-11	11.5	0.11
Helmet_material (Polycarbonate)	1.2E-09	2300	0.32

Table 30. Density and Elastic properties of the materials used in the model

PLASTIC BEHAVIOR	Yield Stress (MPa)	Plastic Strain
Padding_material (EPS20)	0.140	0.1
	0.160	0.2
	0.190	0.4
Helmet_material (Polycarbonate)	30	0
	90	0.1
	80	0.2
	70	0.3

Table 31. Plastic properties of the materials used in the model

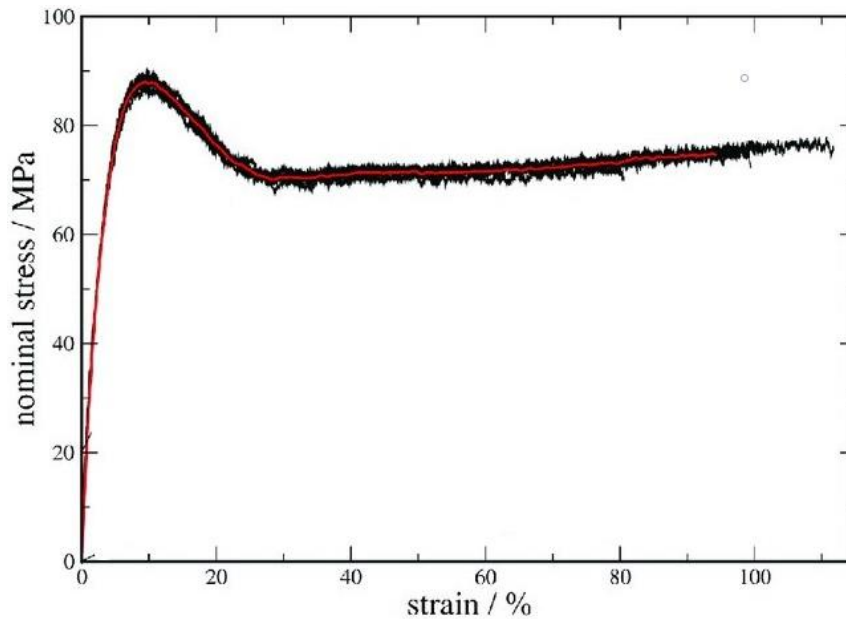


Figure 73. Stress-strain diagram of Polycarbonate. The averaged red curve was obtained starting from 6 experimental tests (black lines)

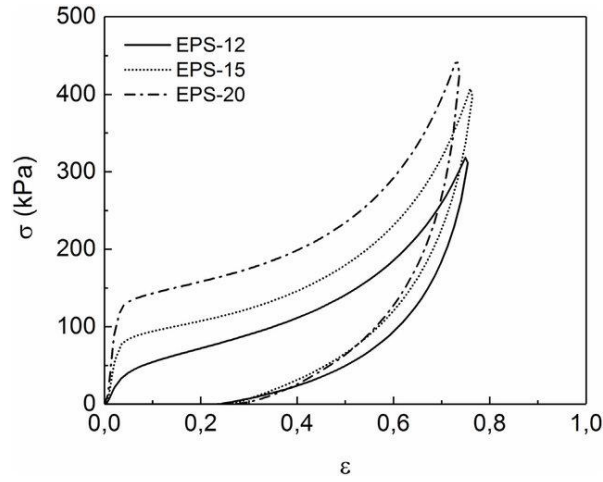


Figure 74. Stress-strain diagram of EPS for 3 different densities of the material. The EPS chosen during the project was EPS_20

Viscoelastic Behavior: Maxwell Model in Abaqus Prony Series

A viscoelastic material exhibits both viscous and elastic characteristics, depending on the type of deformation it is subjected: the response to an instantaneous strain or deformation is not instantaneous, but time dependent. Essentially, if the load is fastly applied, the material behaves as an elastic one, if the load is slowly applied, the material shows a fluidic behavior.

The Maxwell model is the simplest idealization of the viscoelastic behavior of materials, consisting of a connection in series of a dashpot and a spring (Figure 75).



Figure 75. Maxwell model

$$\sigma = E_1 \cdot \varepsilon$$

$$\sigma = \eta_1 \cdot \dot{\varepsilon}$$

The spring (elastic element) is characterized by the stiffness E , while the dashpot (viscous element) is characterized by the viscosity η .

Applying a load to the Maxwell model, the strain of the Maxwell model is equal to the sum of the strains of the two elements.

$$\varepsilon_M = \varepsilon_S + \varepsilon_D$$

Solving the differential equation through the use of the Laplace transform, i.e. passing through the frequency domain and then returning to the time domain, we obtain the differential equation of the Maxwell model:

$$\dot{\varepsilon} = \frac{\dot{\sigma}}{E_1} + \frac{\sigma}{\eta_1}$$

Applying a constant strain, ε_0 , in time to the Maxwell model, what is the response of the stress on time?

$$\varepsilon(t) = \varepsilon_0 \cdot u(t)$$

where $u(t)$ is the Heaviside step function. Using again the direct and the inverse Laplace Transformation we obtain:

$$\sigma(t) = \varepsilon_0 \cdot E_1 \cdot e^{-\frac{t}{\tau_1}}$$

where τ_1 is the relaxation time, defined as $\tau_1 = \frac{\eta_1}{E_1}$. The Eq. () defines the application of a constant deformation to the Maxwell model: that expression is called the stress-relaxation phenomenon [55]. The curves are reported in Figure 76.

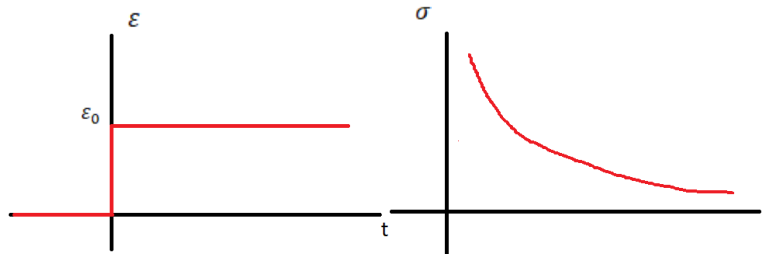


Figure 76. Stress-relaxation phenomenon

A single series of dashpot and spring is not enough to describe the true viscoelastic behavior of materials, so the Generalized Maxwell model (Figure 77) is composed by many Single Maxwell model, connected in parallel. In addition, the first element in parallel is a single spring characterized by the stiffness at time $t=0$, E_∞ .

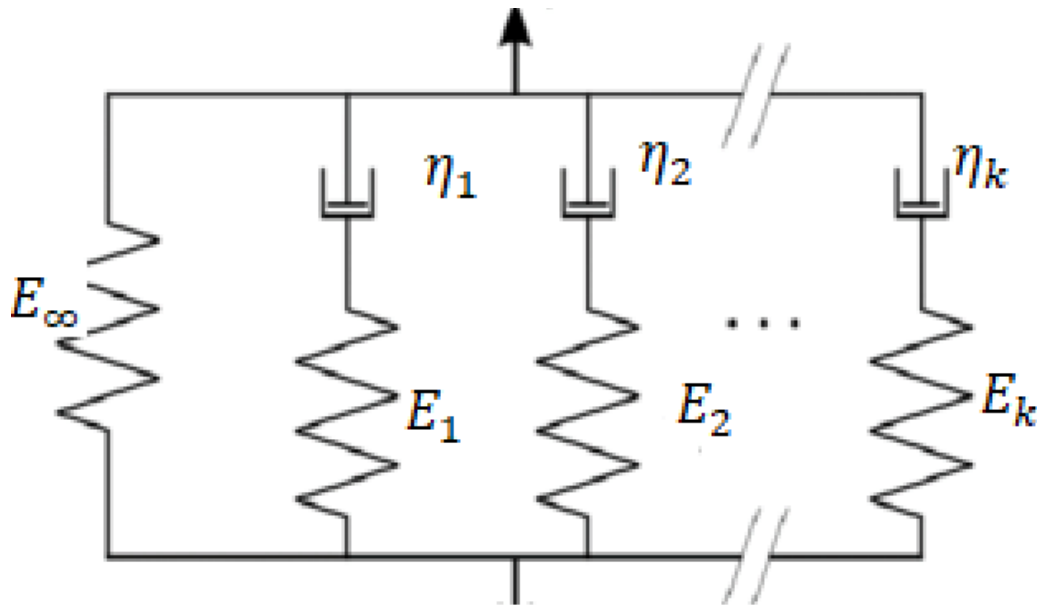


Figure 77. Generalized Maxwell model

The elements are placed in parallel, if a constant deformation ε_0 is applied, the total stress is given by the sum of the stresses of the individual Maxwell elements. If a constant strain ε_0 is imposed at $t=0$ the solution of a single element is:

$$\sigma_k = \varepsilon_0 \cdot E_k \cdot e^{-\frac{t}{\tau_k}}$$

Where τ_k , the relaxation time of the k^{th} element, is defined as $\tau_k = \frac{\eta_k}{E_k}$.

So, the total stress of all the Maxwell elements is given by a summation:

$$\sigma(t) = \varepsilon_0 \cdot E_\infty + \sum_{k=1}^n (\varepsilon_0 \cdot E_k \cdot e^{-\frac{t}{\tau_k}})$$

Obviously, in the element representing the time $t=0$, the response is that given by a simple spring, the Hook law. Collecting ε_0 :

$$\sigma(t) = \varepsilon_0 \cdot [E_\infty + \sum_{k=1}^n (E_k \cdot e^{-\frac{t}{\tau_k}})] = \varepsilon_0 \cdot E(t)$$

This series of exponentials was called Prony Series, developed by Gaspard Riche de Prony in 1795 and used in digital computer as an alternative to Fourier series for signal analysis.

Remembering the elastic properties of the materials:

$$G = \frac{E}{2 \cdot (1 + \nu)}$$

The definition of the shear modulus G is the ratio of shear stress to the shear strain. The parameter is dependent by the Young modulus E and the Poisson ratio ν .

$$K = \frac{E}{3 \cdot (1 - 2 \cdot \nu)}$$

The definition of the bulk modulus K is the ratio of the infinitesimal pressure increase to the obtained relative decrease of the volume. The parameter is dependent by the Young modulus E and the Poisson ratio ν .

In the viscoelastic view, since E is time dependent, so are G and K .

$$E = E(t) = E_\infty + \sum_{k=1}^n (E_k \cdot e^{-\frac{t}{\tau_k}})$$

$$G = G(t) = \frac{E(t)}{2 \cdot (1 + \nu)}$$

$$K = K(t) = \frac{E(t)}{3 \cdot (1 - 2 \cdot \nu)}$$

In literature the pair E_k and τ_k is called Prony pair; Abaqus, instead, asks other values as inputs.

For Abaqus:

- At $t=0 \rightarrow E(t=0) = E_\infty + \sum_{k=1}^n (E_k) = E_0$, the dashpot cannot immediately respond to a strain, at first only a rigid reaction occurs, typically elastic. E_0 represents the total stiffness of the system at $t=0$.
- At $t>0 \rightarrow E(t) = E_\infty + \sum_{k=1}^n (m_k \cdot E_0 \cdot e^{-\frac{t}{\tau_k}})$, where $m_k = \frac{E_k}{E_0}$.

Similarly, for Shear modulus and Bulk modulus:

$$G(t) = G_\infty + \sum_{k=1}^n (g_k \cdot G_0 \cdot e^{-\frac{t}{\tau_k}}) \quad g_k = \frac{G_k}{G_0}$$

$$K(t) = K_\infty + \sum_{k=1}^n (k_k \cdot K_0 \cdot e^{-\frac{t}{\tau_k}}) \quad k_k = \frac{K_k}{K_0}$$

Abaqus requires as input:

1. g_k
2. k_k
3. τ_k

Moreover, the user must decide how many elements consider in the Prony series. Papers in literature [56] suggests using two elements to model the viscoelastic behavior of four brain regions: the cortex, the corpus callosum, the basal ganglia and the corona radiata. In the model of the project only the data of the cortex (Brain_material) and of the corpus callosum (Corpus_callosum_material) were considered. Table 32 was composed processing (simple normalization operations, to obtain g_k from G_k) the data of the paper.

VISCOELASTIC BEHAVIOR	g_i Prony	k_i Prony	τ_i Prony
Brain_material			
1	0.6	0.6	1.63
2	0.28	0.28	20.28
Corpus_callosum_material			
1	0.62	0.62	1.26
2	0.24	0.24	31.62

Table 32. Parameters used as input in Abaqus for the Prony series of two viscoelastic materials inside the model

SPH Theory

In hydrodynamics theory there are two main methods to specify the flow field: the Eulerian and the Lagrangian method.

In summary, in the Eulerian method the flow properties (the velocity v , the density ρ , the pressure p) are functions of the space and of the time; the observer is integral to a fixed reference, he photographs the field at a certain instant. The space is divided in grids, the fluid dynamics parameters are observed in each grid.

In the Lagrangian method the flow properties are functions of the particular fluidic element and of the time; the observer is focused to a single particle.

The classical example to explain the difference between these two methods is that in the Eulerian method the observer is sit at the bank of the river and watches the flow, in the Lagrangian method the observer is sailing on a boat in the same river.

Smoothed Particle Hydrodynamics (SPH) is a meshfree Lagrangian method. Originally developed at the end of seventies with the aim of modeling astrophysical phenomena, then it was applied in other scientific fields that required computational calculations. It is a meshfree method, it does not require connection between nodes, but is based on interaction of each node with all its neighbors, that are particles. Moreover, the Lagrangian nature of the SPH method allows to have a local resolution of the particles, useful for representing the high-density contrasts often found in astrophysical problems.

The idea of the SPH is to approximate the continuum dynamics of fluids using particles, viewed as interpolation points.

The dynamics of the fluids is generally described by incompressible Navier-Stokes equations:

$$\rho \left[\frac{\partial v}{\partial t} + v \cdot \nabla v \right] = \rho g - \nabla p + \mu \nabla^2 v$$

$$\rho(\nabla \cdot \mathbf{v}) = 0$$

Considering incompressible fluids, so $\rho = \text{const}$, for a single particle i :

$$\frac{\partial v_i}{\partial t} = g - \frac{1}{\rho_i} \nabla p + \frac{\mu}{\rho_i} \nabla^2 \mathbf{v}$$

During the computational simulation, the Navier-Stokes equations are sensitive to scale: the software Abaqus chooses the optimum time scale of the simulation, but it is allowed to the user changing the settings.

As said before, the base of SPH method are the particles, considered interpolation points. The integral interpolant of a function $A(\mathbf{r})$ is:

$$A_I(\mathbf{r}) = \int A(\mathbf{r}') W(\mathbf{r} - \mathbf{r}', h) d\mathbf{r}'$$

In which the integration is over the space, \mathbf{r} is the position, h is the interaction radius. W is the interpolating Smoothing kernel with three properties:

- $W(\mathbf{r} - \mathbf{r}_b, h) = \frac{315}{64\pi h^9} (h^2 - \|\mathbf{r} - \mathbf{r}_b\|^2)^3$
 - $\int W(\mathbf{r} - \mathbf{r}', h) d\mathbf{r}' = 1$
 - $\lim_{h \rightarrow 0} W(\mathbf{r} - \mathbf{r}', h) = \delta(\mathbf{r} - \mathbf{r}')$

Effectively the introduction of a smoothing kernel, ideated by Monaghan in 1992 [57] is a way to evaluate a quantity by sampling a neighborhood space and weighting points according to how they are close to our sample point (Figure 78).

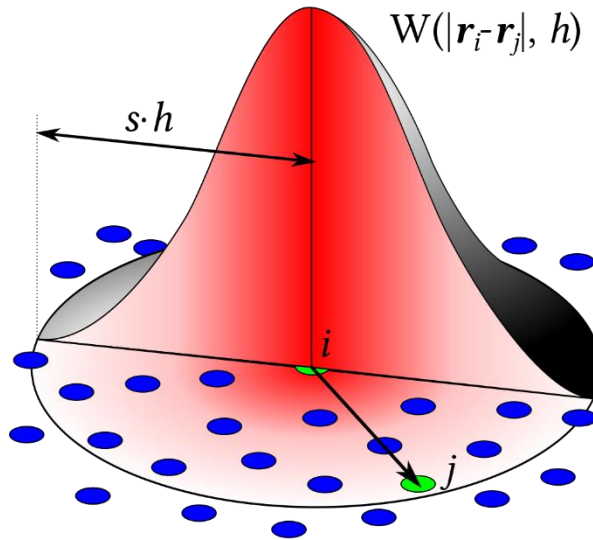


Figure 78. SPH convolution

In numerical computation the integrals can be approximated as summations:

$$A_I(\mathbf{r}) = \int A(\mathbf{r}') W(\mathbf{r} - \mathbf{r}', h) d\mathbf{r}' \cong \sum_b A(\mathbf{r}_b) W(\mathbf{r} - \mathbf{r}_b, h)$$

Where the index b indicates a single particle.

Also, the density can be approximated as the sum of the masses of nearby points weighted by the kernel W :

$$\rho_i \approx \sum_j m_j W(\mathbf{r} - \mathbf{r}', h)$$

And so, the pressure gradient:

$$\frac{\nabla p_i}{\rho_i} = \sum_j m_j \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} \right) \nabla W(r - r_j, h)$$

And the Laplacian of the velocity v :

$$\frac{\mu}{\rho_i} \nabla^2 v \approx \frac{\mu}{\rho_i} \sum_j m_j \left(\frac{v_j - v_i}{\rho_j} \right) \nabla^2 W(r - r_j, h)$$

Where $v_j - v_i$ is the difference of velocity between two points, i and j .

The steps leading from Navier-Stokes to the SPH formulation are shown below:

$$\begin{aligned} \rho_i &\approx \sum_j m_j \frac{315}{64\pi h^9} (h^2 - \|r - r_b\|^2)^3 \\ \frac{\nabla p_i}{\rho_i} &\approx \sum_j m_j \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} \right) \frac{-45}{\pi h^6} (h - \|r - r_b\|)^2 \frac{r - r_b}{\|r - r_b\|} \\ \frac{\mu}{\rho_i} \nabla^2 v_i &\approx \frac{\mu}{\rho_i} \sum_j m_j \left(\frac{v_j - v_i}{\rho_j} \right) \frac{45}{\pi h^6} (h - \|r - r_b\|) \end{aligned}$$

Once computed these three equations, strictly in this order (density, gradient of pressure and viscous term), all the terms are inserted by the program in Navier-Stokes:

$$\frac{dv_i}{dt} = g - \frac{1}{\rho_i} \nabla p + \frac{\mu}{\rho_i} \nabla^2 v$$

And conclude through a simple time stepping numerical integration scheme to advance, firstly the velocity of the particles, and then their positions.

In the SPH application the definition of the equation of state, that regulates the behavior of the material under certain physical conditions, becomes fundamental. Abaqus allows the user to set the Material Behaviors EOS (Equation of state), and to choose the type:

- Ideal gas;
- JWL, from the Jones-Wilkins-Lee equation of state;
- Us-Up;
- Ignition and growth;
- Tabular.

For the CSF a linear Equation of state (type Us-Up) was chosen, approximating its physical behavior to the same as water: the approximation was justified by the fact that CSF consists of 99% water. The equation of EOS type Us-Up is:

$$U_s = c_0 + sU_p$$

Where U_s is the shock wave velocity, c_0 is the speed of sound through the material, Γ_0 is Grüneisen's gamma at the reference state, $s = \frac{dU_s}{dU_p}$ is the linear Hugoniot slope coefficient and U_p is the particle velocity. This linear form is a derivation of the complete Mie – Grüneisen equation of state, used in the field of shock wave propagation. The parameters chosen for the water in the literature relating to this EOS, and therefore transferable to the CSF, are those indicated in Table 33.

EQUATION OF STATE	c_0 (mm/s)	s	Gamma0
CSF_material	1.45	0	0

Table 33. Parameters used as input in Abaqus for the Us-Up Equation of State of the CSF

A third Material Behavior to the CSF_material has also been added, the viscosity. Remaining consistent with the fluid dynamic association between water and CSF, a Newtonian dynamic viscosity similar to that of water was set (Table 34).

VISCOSITY	Dinamic viscosity
CSF_material	2.5E-07

Table 34. Parameter used as input in Abaqus for the Viscosity of the CSF

Characterized with these physical properties and meshed with the Conversion to particle technique (explained in the chapter of the Mesh module), the CSF will behave like a sea in which the brain will “navigate” (Figure 79).

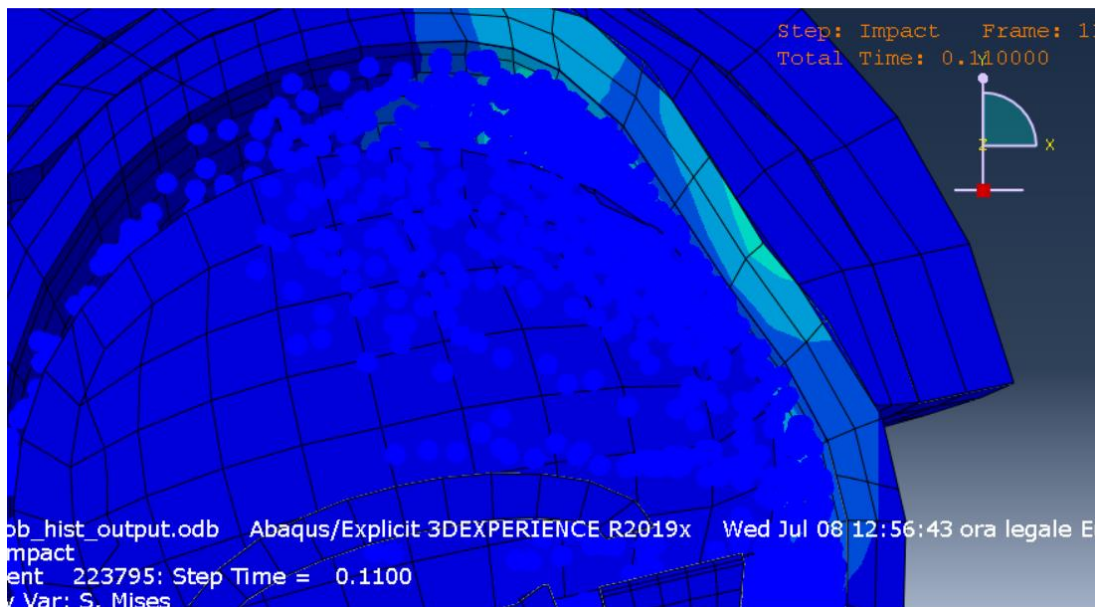


Figure 79. Section of the model: detail of the particles, visible during a deformation phase of the simulation

Among the physical properties of the model we must also mention the interaction properties: a tangential behavior has been set in the contact property options throughout the model, defining it with an isotropic friction coefficient of 0.3.

Assembly Module

During the assembly phase, each part was inserted in its correct anatomical position (Figure 80). The internal organs were inserted inside the skull and immersed in the CSF fluid. The skull was then placed on the neck and then surrounded by the two layers that composed the helmet. The displacement operations, necessary to place the parts correctly, consisted in translations and rotations. The perfect adhesion between the various components was facilitated by Abaqus: during a translation the user can indicate the starting point of the object, which must obviously belong to the object itself, and then indicate the point of arrival, where the user wants the starting point go to finish, which can easily be a point belonging to another object, which wants to put in contact with the starting object.

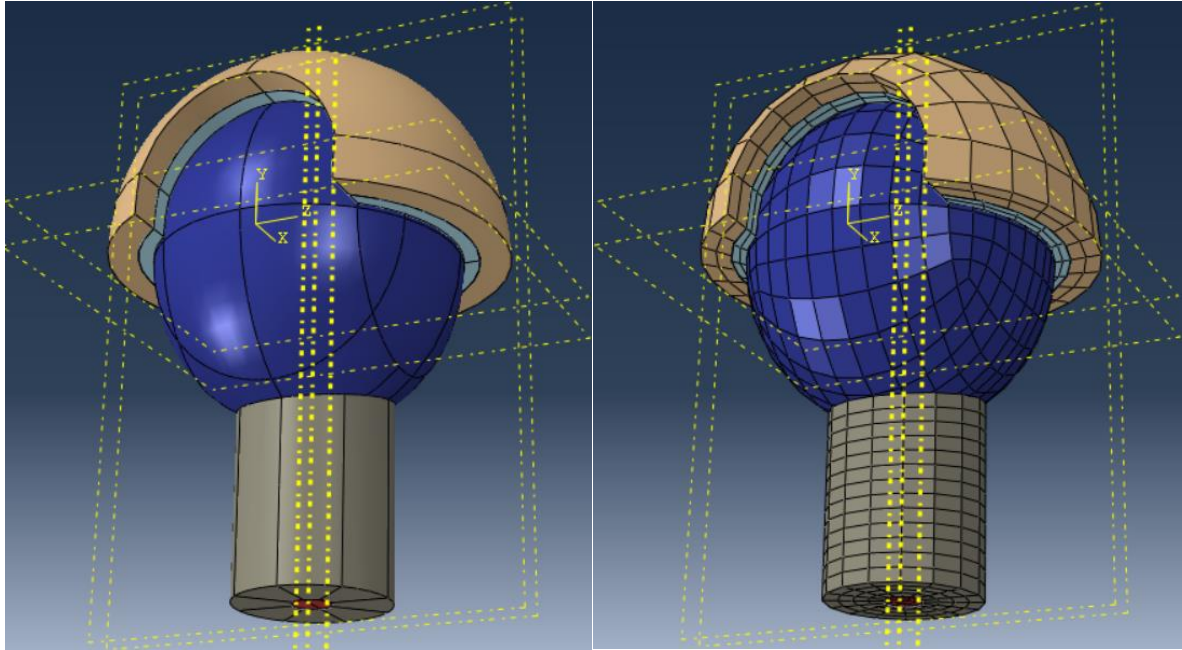


Figure 80. Assembly of the whole model, normal and meshed version

All cutting operations of the parts (such as those carried out on the skull, neck, CSF, or helmet structure), described in the chapters of the individual parts, were performed using the tool Merge/Cut Instances in the Assembly module (Figure 81).

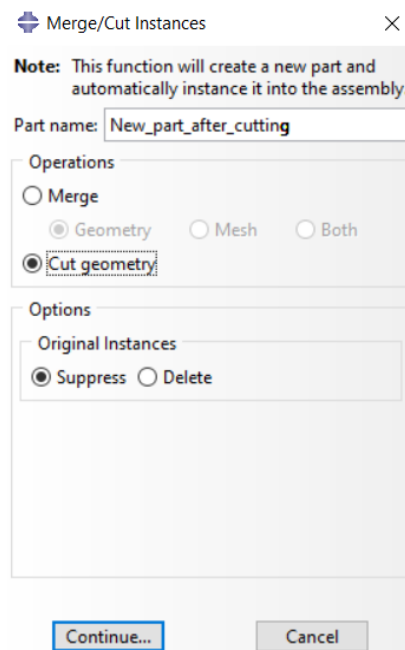


Figure 81. Procedure screen for the merging/cutting operation

This tool asks the user to indicate two parts, to decide whether to merge or cut them, and the result is always the generation of a new part (New_part_after_cutting or New_part_after_merging). Abaqus allows the user to decide whether to suppress or permanently eliminate the parts at the origin of the operation. Within the project, no merging operations were carried out, but only cutting.

Mesh Module

In the mesh module the user creates the finite element meshes of all parts: this is the core process of all FEM theory, the discretization. The meshing operations were the most complex and took a long time of work. The goal was to mesh the model components as regular as possible. In a finite element simulation software, the computation times are constrained by the smallest element of the whole model: the smaller the smallest element of the model, the more processing time increases. Therefore, the meshing phase becomes crucial. Moreover, the meshes must be regular in order to have realistic responses after the simulation.

All elements used to mesh the model were taken from the Explicit Library. The visible lines drawn on the individual parts are called partitions in Abaqus. The partition is the operation to divide parts or independent part instances into smaller regions, and this toolset is placed in the meshing step. There are two main reasons to create partitions in the Mesh module: to assign different element types to different regions (created by the partition), or, as in the case of this project, to divide complex parts into simpler regions, easier to mesh using primarily hexahedral elements. So, the goal of the partition is to have more control over mesh generation. Abaqus provides the user with several tools to make a partition, all of which are ways to define a plane that cuts the object. The plane can be defined by selecting 3 points on the object, or by fixing a plane parallel to one of the 3 Cartesian axes (x , y , z). The planes defined during the partition are then no longer erasable, otherwise the partition itself is deleted and they remain visible (dashed yellow lines) in the previous figures.

In the window of the Element Controls of all the parts the meshing was set to a Second-order accuracy and the distortion control was activated, to avoid excessive distortion of the elements during simulation shocks.

The partition phase made it possible to assign Hex elements to all parts, except the CSF, the geometrically most complex part, to which, despite the partition, it was possible to assign only Tet elements. Furthermore, to characterize the fluidic behavior of CSF, according to the SPH theory, the option Conversion to particle was set for it.

Assigned all elements the entire structure, Abaqus allows a verification of the Mesh module: it highlights in yellow the Warnings, the most unstable elements, those can be subject to the greatest deformations during the simulation, which can therefore slow down the calculations; moreover, it highlights in violet the Errors, the elements that will surely interrupt the running of the job. In the Abaqus vocabulary the term “job” indicates simulation. The various evolutions and changes made during the meshing phase were then checked with the Abaqus Verify tool, till to reach a minimum percentage of warnings (around 5%) and no errors (Figure 82). The statistics in details are summarized in Table 35.

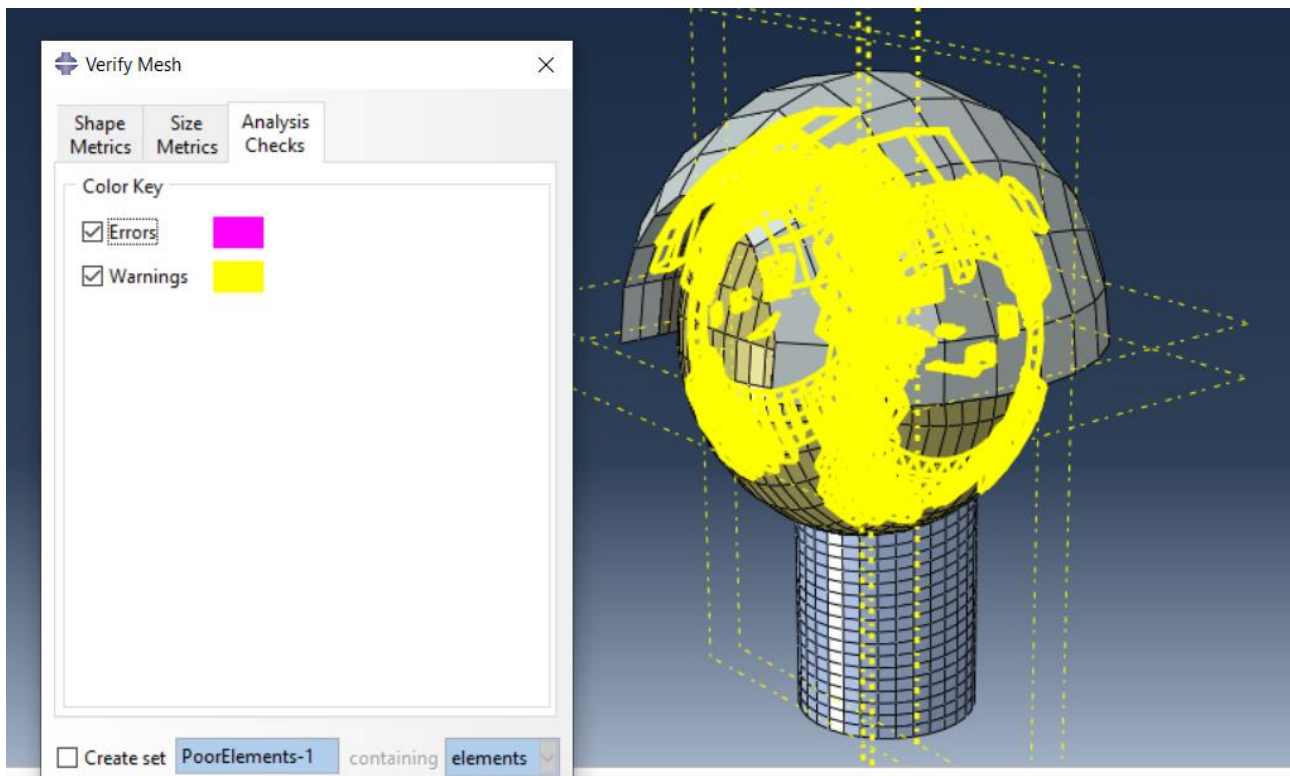


Figure 82. Tool Verify mesh on the assembly

	Number of elements	Analysis errors	Analysis warnings
Corpus_Callosum	12	0 (0%)	0 (0%)
Left_Hemisphere	320	0 (0%)	0 (0%)
CSF	7443	0 (0%)	94 (1,263%)
Right_Hemisphere	320	0 (0%)	0 (0%)
Cerebellum	388	0 (0%)	0 (0%)
Helmet	204	0 (0%)	26 (12,745%)
Column	240	0 (0%)	0 (0%)
Padding	204	0 (0%)	5 (2,451%)
Skull	1936	0 (0%)	445 (22,986%)
TOTAL	11067	0 (0%)	570 (5,150%)

Table 35. Mesh statistics of the model

Load Module

The next module in Abaqus modeling is the Load module, where the user can define Loads, Boundary conditions or Predefined fields to apply to the model.

In the project a boundary condition, type Encaster, was imposed on the lower surface of the neck and brainstem. An Encaster boundary condition forced the elements inside the selected region to have a linear velocity and a rotational velocity equal to zero, in all 6 axes ($U1 = U2 = U3 = UR1 = UR2 = UR3 = 0$, as Abaqus reminds). With the approximation of the Encaster the idea was to represent the attachment of the neck to the rest of the body. Once the encaster has been applied, Abaqus pictures small cones on the elements involved, the two different colors indicate the different nature of speed, linear or angular, which is set equal to zero (Figure 83).

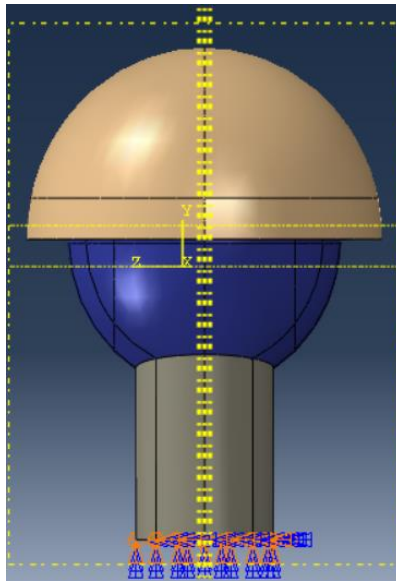


Figure 83. Encaster Boundary Conditions below the surface of the neck and of the brainstem

Moreover, the Load module was the place of the connection between the two worlds of the project: the data acquisition with the monitoring system and the finite element modeling. Once a shock is detected, therefore a potential danger for the athlete, the accelerations acquired by the monitoring system can be imposed as boundary conditions on the head model.

The imposition of accelerations was done consistently on where the accelerations were measured, that is where the monitoring system is implanted, within the padding layer of the helmet. Because of the discretization of the finite elements, concentrating the accelerations in a single point would not have been realistic, therefore a square, 9 points, of dimensions comparable to the space occupied by the monitoring system was selected (Figure 84). These 9 points were collected together by creating a Set in the Part module, referring to the Padding component. A Set is a special spatial region to which a specific material can be assigned or a Load or a Boundary condition (as in this case) can be imposed. Moreover, as will be described later, a Set can be useful to indicate the spatial region in which to make a measurement (Definition of a History Output Request).

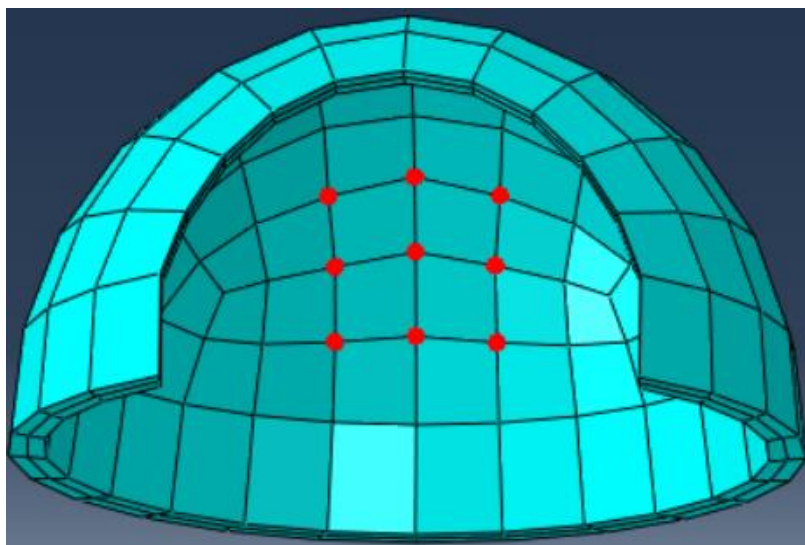


Figure 84. The 9 points that approximate the location of the monitoring system on the padding layer of the helmet

6 accelerations were imposed, 3 linear and 3 angular (Figure). Abaqus allows the user to create an Amplitude profile over the time of the set quantities, also for acceleration: the 6 temporal profiles acquired by the monitoring system have been inserted into the model.

The duration of the amplitude profile must be compatible with the duration of the impact: in the example simulation the duration was set to 0.2 s. Abaqus requires the user to enter percentage amplitude values along the amplitude profile, with a range from -1 to 1, and to also indicate the reference value of 100%, the maximum amplitude.

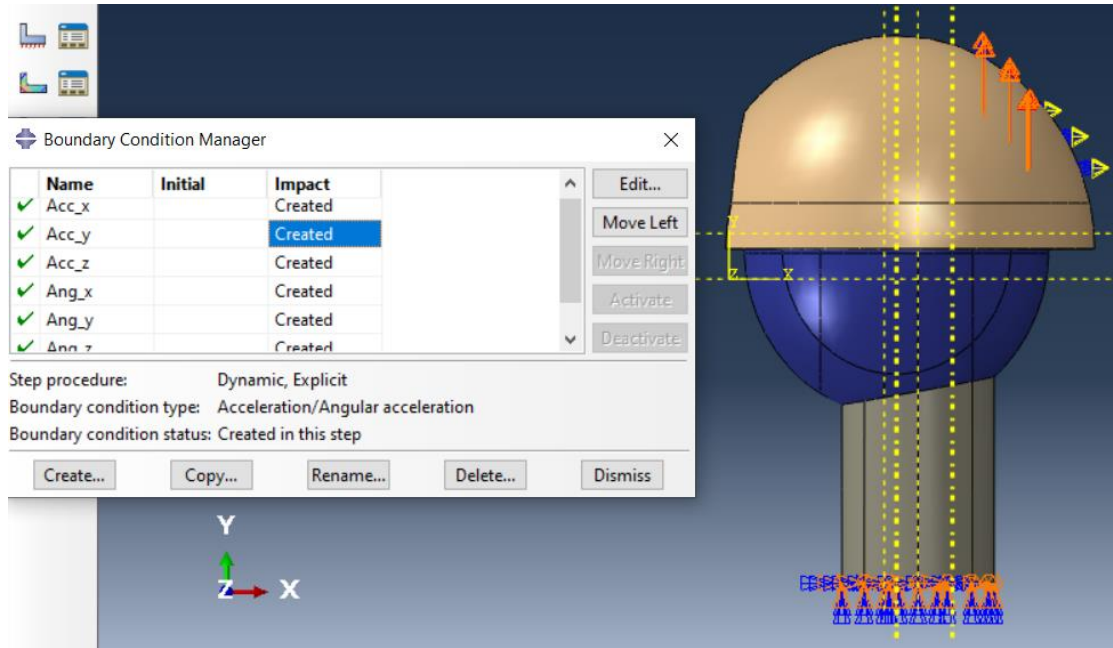


Figure 85. Example of Abaqus representation, through arrows, of the imposed accelerations

The telemetry data tested in this simulation (3 linear accelerations and 3 rotational accelerations) was measured with the ShockAccel electronic system, implemented in this project. The data were obtained by sharply moving the electronic system, to mimic a shock suffered by an athlete's head, and the wave peak was recorded in these 0.2s.

To respect the units of measurement required by Abaqus, the linear acceleration data was converted from g to $\frac{mm}{s^2}$. Moreover, again due to the request of the Abaqus system, a zero acceleration had to be imposed at time $t = 0$. As described in the chapters concerning the electronics, the sampling frequency was 40 Hz, so data was collected every 25ms (Table 36).

TIME	acc_x	acc_y	acc_z	ang_x	ang_y	ang_z
0	0	0	0	0	0	0
0,025	-37000	0	11040	157,9572	248,8068	853,65
0,05	24050	7050	-24000	418,122	-145,766	-836,577
0,075	-25900	-4950	12960	-603,954	203,5692	657,3105
0,1	15170	3000	-19920	557,496	-198,543	-682,92
0,125	-8140	-12000	11040	-566,788	133,1996	256,095
0,15	2960	-10950	-1920	929,16	125,66	-110,975
0,175	9990	-1050	-6000	-919,868	-251,32	-93,9015
0,2	-21090	15000	16080	306,6228	120,6336	623,1645

Table 36. Telemetry data imposed on the simulation: acc values are expressed in $\frac{mm}{s^2}$, ang values are expressed $\frac{rad}{s^2}$

The imposition of a load in Abaqus takes place in two steps: in the first, the maximum load is defined (the absolute values of the data in pink in Table 36), in the second the amplitude profiles are defined (Table 37), which, as already explained, are percentage values of the respective maximum loads.

TIME	acc_x	acc_y	acc_z	ang_x	ang_y	ang_z
0	0	0	0	0	0	0
0,025	-1	0	0,46	0,17	0,99	1
0,05	0,65	0,47	-1	0,45	-0,58	-0,98
0,075	-0,7	-0,33	0,54	-0,65	0,81	0,77
0,1	0,41	0,2	-0,83	0,6	-0,79	-0,8
0,125	-0,22	-0,8	0,46	-0,61	0,53	0,3
0,15	0,08	-0,73	-0,08	1	0,5	-0,13
0,175	0,27	-0,07	-0,25	-0,99	-1	-0,11
0,2	-0,57	1	0,67	0,33	0,48	0,73

Table 37. Amplitude profiles, percentage values of the maximum loads.

A next step, which was not done in this project, would be the calibration between the axes of the internal sensors in the ShockAccel system and the reference axes of the Abaqus model, to have a correct representation of the shock suffered by the athlete on the finite element model.

Interaction Module

In the interaction module the user defines the desired interaction between two different parts. As already explained in the assembly module, all the parts have been placed in the correct position, in contact with each other. Abaqus proposes different types of constraints to keep in contact two different components during the simulation: the tie constraint was chosen in the project. To define a tie constraint the user needs to select the two surfaces that he wants to connect, declaring which of them is the master and which the slave. 9 tie constraints were defined in the model:

- Tie corpus callosum – left_hemisphere;
- Tie corpus callosum – right_hemisphere;
- Tie cerebellum – left_hemisphere;
- Tie cerebellum – right_hemisphere;
- Tie column – left_hemisphere;
- Tie column – right_hemisphere;
- Tie helmet – padding;
- Tie skull – neck;
- Tie skull – padding.

The 3 sections of the model are shown in Figure 86, alongside the respective versions with a different degree of translucency, in order to better see the internal details.

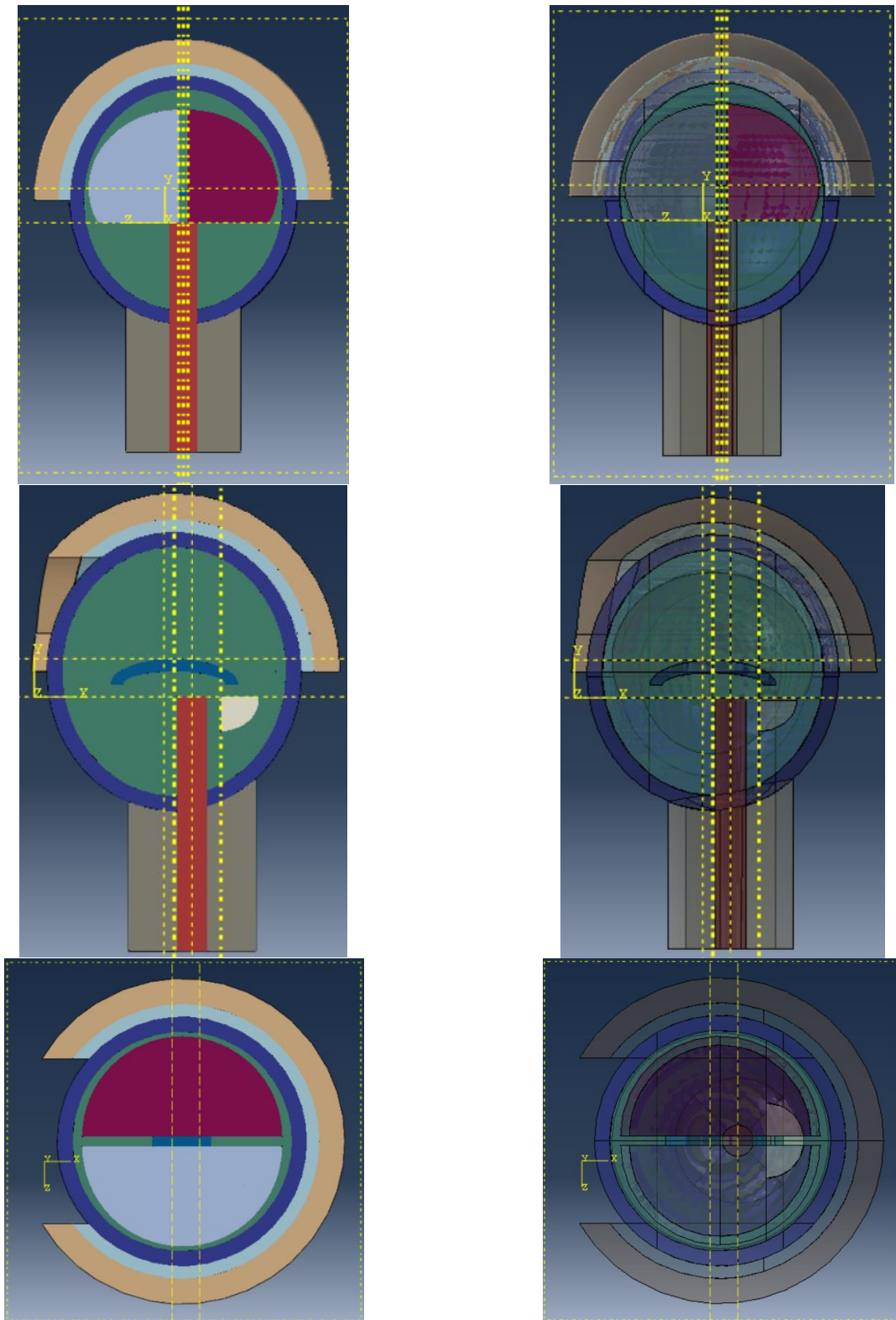


Figure 86. Left: frontal, lateral and top section view of the model. Right: the same sections are represented with a translucency level of 30%

Step module

In the Step module the user can define a sequence of analysis steps in which the entire simulation is developed. In each step the user can set different parameters, such as the interactions between the various components, or the loads/boundary conditions to be applied to the structure or thermodynamical conditions, in order to give a dynamic sense (a variation of the conditions over time) to the simulation.

Abaqus by default creates a first step, called Initial step, without giving the user the possibility to rename, modify, copy or delete it: in the Initial step the user can define all the parameters of the simulation at time $t=0$.

Abaqus allows to create further steps, of which the Procedure type must first be defined: it can be General or Linear Perturbation. The main difference between the two types is that the General also includes non-linearity effects in the simulation, so it is more complete but also heavier from a computational point of view.

Once the General type has been chosen, Abaqus proposes various subtypes, to define the physical nature of the simulation to be carried out, whether static, dynamic or thermodynamic: there are 15 specific subtypes, all belong to the three domains mentioned above. The choice of one of these subtypes influences the nature of the subsequent steps.

The model of the project showed 2 steps, the Initial (by default) and the Impact_step, created by the student (Figure 87). The Impact_step was defined as General, of Dynamic Explicit type, with a time period of 0.2 s (Figure 88). Being the Impact_step the only step, except the Initial, the duration of the entire simulation assumed the same value as the time period of the Impact_step. Therefore, all the parameters previously described (in the Load and Interaction modules) are to be considered valid during the Impact_step.

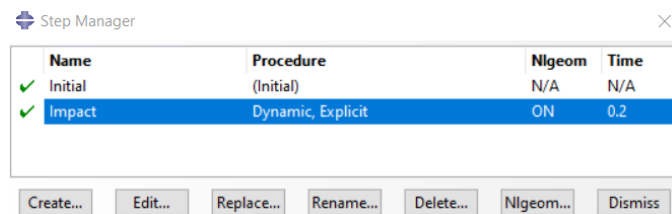


Figure 87. Step Manager of the model, containing 2 steps

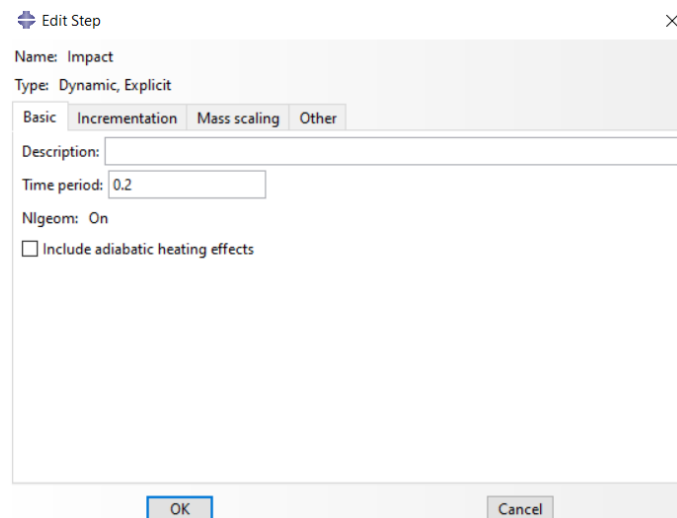


Figure 88. Editing of the Impact_step

For the most computationally demanding simulations, above all those characterized by very small elements that will slow down the calculations, Abaqus provides a tool, inside the Step module, called Mass scaling. The idea behind this technique is that Abaqus “scales” the mass of the smallest elements to make them more homogeneous with the other elements of the structure and to gain computational efficiency.

The Mass scaling technique is more effective as smaller the population of small elements that slows down the calculations. Mass scaling mode has not been set in the model of this project.

Job Module

Once finished the procedures to define the model (defining the geometry, assigning the material properties, assembling and meshing all the part, defining contacts and imposing the loads), Abaqus analyzes the simulation through the Job module.

The user can create the job, submit it to the analysis process and, at the same time, monitor the progress of the analysis: Abaqus communicates, during the monitoring, any errors, which cause the simulation to abort, or warnings, which slow down the calculation or which they can lead to distortions of some elements. Each warnings or errors is explained to help the user to understand where the problem is and how to solve it. The monitoring screen is updated every step time, the minimum unit of the simulation calculation, which, as mentioned in the chapter of the Mesh module, is influenced by the minimum size of the model elements.

One of the advantages of the Job module is the possibility to build multiple models and launch the job of each of them at the same time, saving a lot of time.

When creating a new Job Abaqus requires to name it, then to indicate the type of simulation (if Full analysis, like all the simulations made in the project, or Recover analysis, which gives the possibility to recover jobs killed prematurely), to decide the output format of the simulation (if .ODB, chosen by the student, or .SIM), to indicate the number of processors to be used by the software and the level of accuracy of the calculation (if Single or Double). The laptop with which the simulations were conducted (equipped with an Intel i7 processor of 8th generation, 16 gb of RAM and 512 gb of SSD) allowed the use of a maximum of 4 cores, which, coupled to a Double level of precision, resulted in a duration of the simulation described in the report of less than an hour (52 minutes). The same simulation was performed by a computer of iPrint Center with 64 cores and completed it in less than 20 minutes (19 minutes).

An example of a Job manager display is shown in Figure 89. The box indicates, for each Job, the model to which it refers, the type of simulation conducted and the status of the run, which can be:

- successfully completed (Completed);
- killed by the user (Terminated);
- not launched yet (None);
- terminated with error (Aborted);
- still running (Running).

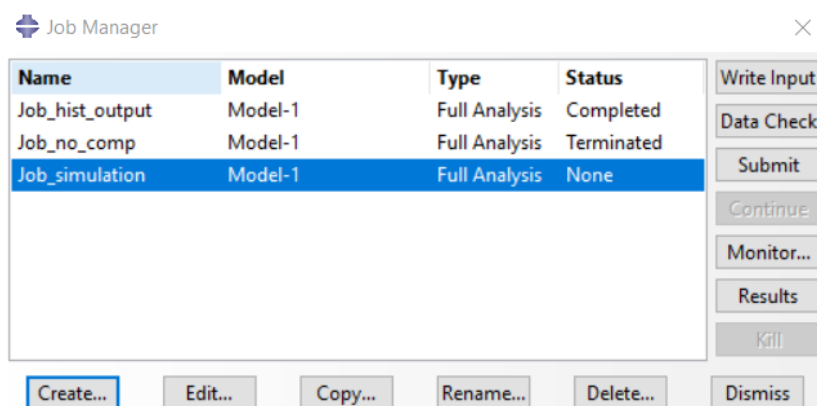


Figure 89. Job manager display with 3 different jobs

The options on the right allow to start the running of the simulation (Submit implicitly includes Write Input and Data Check), to monitor the progress of the Job (Monitor) and, once completed successfully, with Results it is possible to access the telemetry of the measured parameters and graphic animation of the model.

At each submitting of the Job (which includes the execution of the Write Input), Abaqus generates the input file (.inp) by default. This file is the ASCII representation of the model subjected to simulation; it is accessible with a text editor.

Visualization Module

In the Visualization Module the user can read all the outputs that Abaqus generated during the Job phase and graphically view the results of the simulation. This module is accessible in two ways:

- selecting Visualization module in the drop-down menu, where it is possible to find all the other modules provided by Abaqus;
- opening the .ODB file, which is the format chosen, during the Job creation phase, for the analysis output file.

By creating a History Output Request, the user can ask Abaqus to make measurements on the model simultaneously with the simulation calculation. The History Output Request section is external to all Abaqus modules and can be defined for each step of the model, except for the Initial step.

Once that the History Output Request is named and temporally placed in a specific step (in this model the only step besides the Initial was the Impact_step), Abaqus asks to define:

- the Domain, the spatial region where to measure the variable;
- the Frequency, how to sample the measurement of variables;
- the Output Variables, which variables must be monitored.

The Abaqus screen to set a new History Output Request is shown in Figure 90.

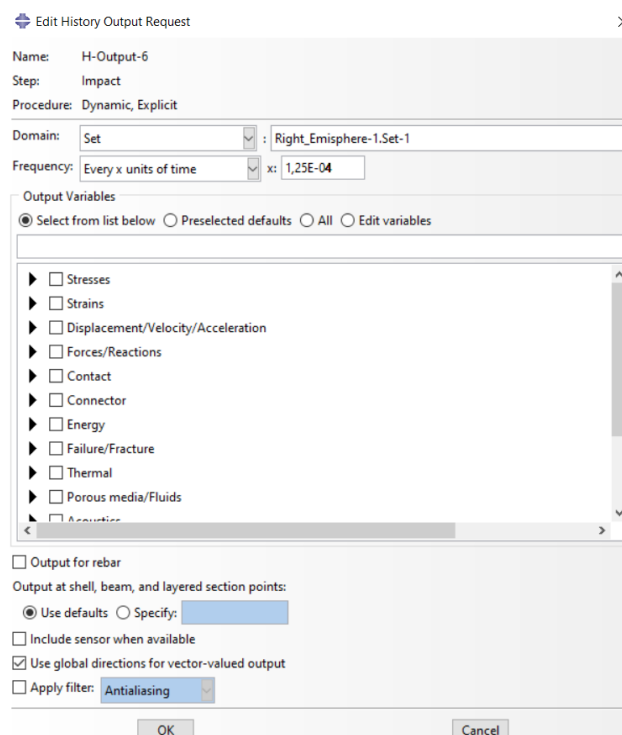


Figure 90. Screen to set the History Output Request

Among the various Domain options, the most useful is to choose a previously created Set: a specific brain region that the user (the clinician in this case) wants to monitor during the evolution of the shock, such as the frontal, temporal, or occipital region.

To obtain a time series of the desired variable with a fixed sampling frequency, it is necessary to set "Every x units of time" in the Frequency item: with an x of 1,25E-04 s, as set in the figure, the output time series will have a sampling frequency of 8kHz.

The output variables that can be monitored are many. The main categories are:

- Stress;
- Strains;
- Displacement/Velocity/Acceleration;
- Forces/Reaction;
- Contact;
- Energy;
- Thermal;
- Acoustics.

The more the variables Abaqus must monitor the more the computational time of the simulation increases.

The most interesting measurements to do, for this specific case, are found in the first three packages: the clinician can assess the temporal trend of physiological parameters (such as the stress suffered by the axons, the intracranial pressure, the Global strain energy of the brain/skull interface, or the inertial forces) of different brain regions. As already specified in the anatomical description, this model does not distinguish between gray matter and white matter and represents the brain as an isotropic body: some parameters mentioned above, such as the stress of the axons, cannot be calculated.

Having the possibility to monitor the acceleration of a specific set (the category Displacement / Velocity / Acceleration), this model allows the calculation of the HIC value, like other models described in the bibliographic search: an .rpt file, containing the accelerations of a point in the brain, such as the center of gravity, can be read by a simple Matlab script that calculates the HIC of the shock.

In addition to the parameters requested by the user when defining the history output for a given set, Abaqus by default also calculates energetic parameters valid for the whole model, such as (in alphabetical order, as shown by Abaqus):

- the artificial strain energy (ALLAE);
- the creep dissipation energy (ALLCD);
- the damage dissipation energy (ALLDMD);
- the external work (ALLWK);
- the frictional dissipation (ALLFD);
- the internal energy (ALLIE);
- the kinetic energy (ALLKE);
- the plastic dissipation (ALLPD);
- the strain energy (ALLSE);
- the total energy of the output set (ETOTAL);
- the viscous dissipation (ALLVD).

An example of one of these energies plotted by Abaqus, at the end of simulation, is shown in Figure 91. Abaqus makes sure that all these parameters respect the law of conservation of energy:

$$E_I + E_V + E_{FD} + E_{KE} - E_W = E_{total} = const$$

This equation indicates that the sum of the internal, the viscous, the frictional, the kinetic energies and the opposite of the work of an external load must be constant.

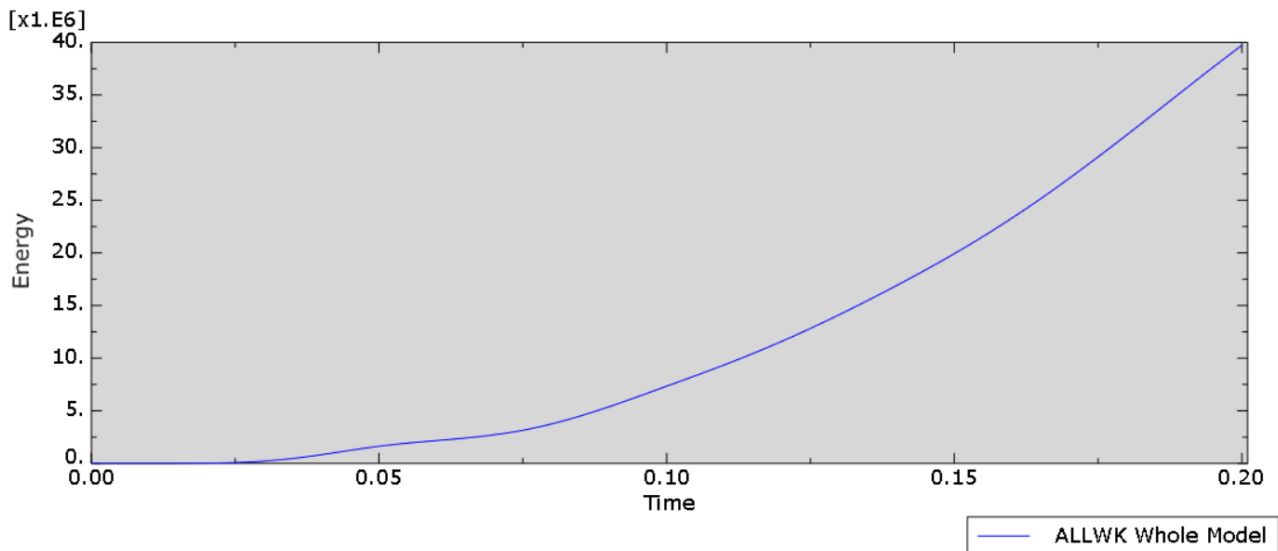


Figure 91. External work, plotted by Abaqus (in mJ), made by the external load imposed on the head model

The gold standard for ICP measurement is a lumbar puncture, followed by intraventricular drainage performed with a pressure transducer (Figure 92): it is a very invasive procedure because it requires reaching the CSF, exposing it and therefore the whole brain to the risk of infections during the measurement. ICP is usually monitored in patients who have suffered head trauma: the nature of this signal is very dynamic, and therefore it is necessary to monitor it for a few hours, preferably with the conscious subject, to then perform an averaging operation, obtaining a mean-ICP [59].

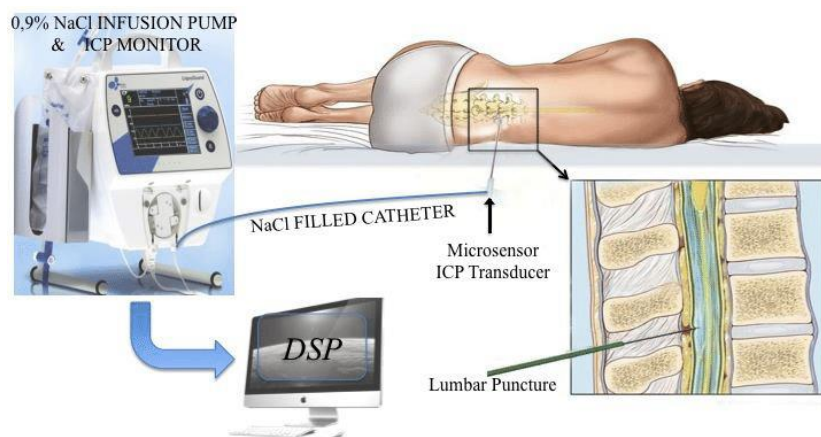


Figure 92. ICP monitoring scheme

With the Abaqus model of this project it was possible to measure the trend of ICP during the shock suffered by the head. A set in the CSF_part (anatomical location where ICP should be monitored) was created, behind the occipital region of the brain; subsequently a History Output Request (PPress, in Stresses category) was created for that set. A file in .rpt format was saved from the ODB, and later read by Matlab, with which the curve was plotted (Figure 93).

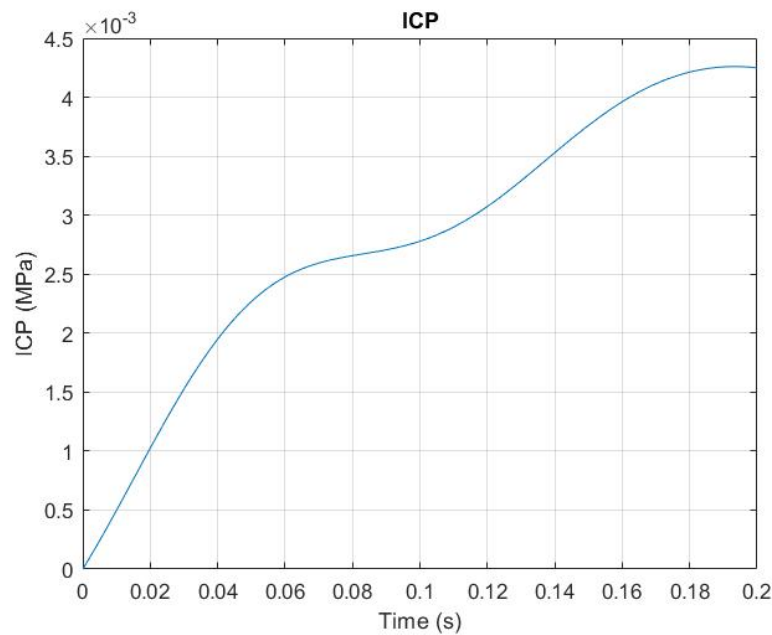


Figure 93. ICP trend measured by Abaqus at the CSF set

Following the shock, imposed as acceleration input in the Load module, ICP increased in the occipital region. As already mentioned, ICP monitoring should be much longer, 0.2s are not enough for such a dynamic signal, but they can give the clinician an idea of the scale of values of the case in question.

Figure 94 shows two views of the model at the end of the simulation (0.2 s, as indicated by Abaqus in the Step Time). The input set in the Load module resulted in a bending of the head, and therefore of the neck, forward. It is possible to notice the result of the Boundary condition (Encaster) imposed at the base of the neck: the base remained anchored to the “virtual ground” and the typical tension distribution is generated around the joint. The color code of the Mises Stresses is placed on the left of the screen.

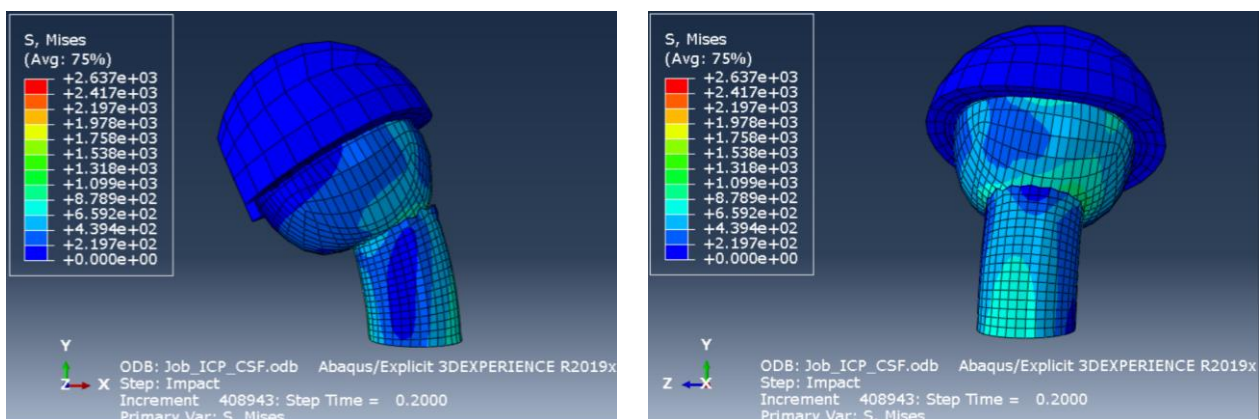


Figure 94. Lateral and back view of the model at t=0.2s

It is also possible to notice deformations of the skull and a non-perfect adhesion between the lower surface of the skull and the upper one of the neck, due to the imperfect compatibility of the meshes of the two different parts. This is a structural and graphic defect, but for the purpose of the simulation, i.e. to monitor brain damage following head shocks, it is negligible.

To graphically monitor the internal structures of the head-brain system, a section-view tool, provided by Abaqus in the Visualization module, is required. Moreover, the user can also select the desired frame within the video-loop (Animate time history tool). Abaqus decides by default, based on the duration of the simulation, how many frames to generate and make available to the user. Alternatively, the user can set the

number of frames. In the simulation of the project the default option has been left: Abaqus made available 21 (20 plus the initial one) frames for a total duration of 0.2s, i.e. one frame every 0.01s. In Figure 95 four different frames are represented.

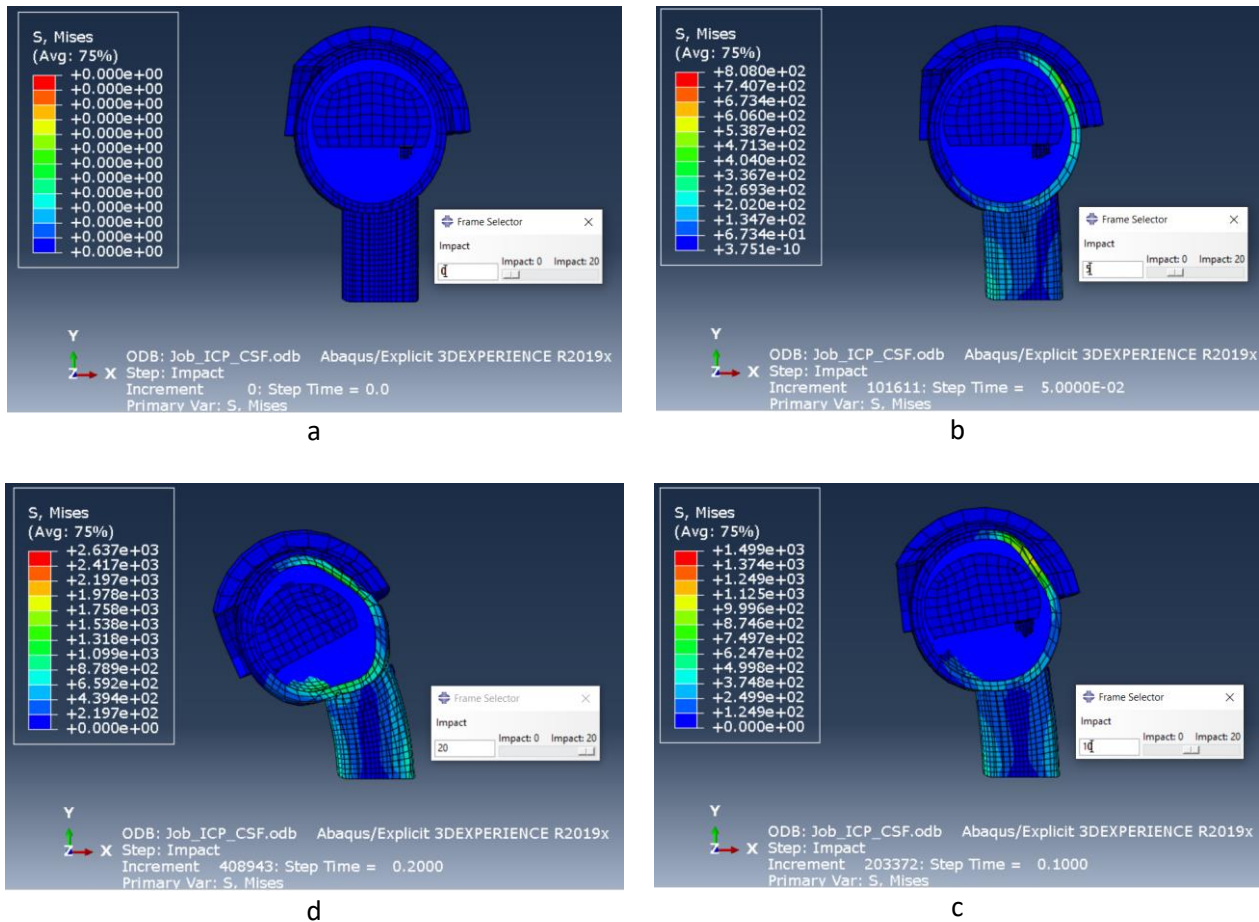


Figure 95. Four frames of lateral view taken from the Animate time history tool. a) $t=0$ s, b) $t=0.05$ s, c) $t=0.1$ s, d) $t=0.2$ s

Unfortunately, the graphic display does not allow to recognize which areas of the brain undergo the greatest stress: the color code highlighted above all the flexion of the neck and the deformation of the skull, so the only way to monitor the clinical parameters of the brain regions are the History Output Requests.

However, one can notice a progressive forward flexion of the neck, as well as of the head, while the brain (in this section only the left hemisphere is visible) "navigates in the sea" of CSF particles, making possible a qualitative vision of coup-contrecoup phenomenon, although not yet complete due to the short duration of the simulation (0.2s).

Limits of the model and possible improvements

The main limitation of the model designed in this project is the lack of the validation phase with any other reference model. As already explained in the introduction of the description of the model, a lot of reticence was found in obtaining other models from other research institutes, not willing to sell their products: the same reticence was found to enter into a collaboration to validate the ShockAccel FEHM with another already validated. The validation phase would have consisted in imposing the same inputs to the two models (the one designed in this project and the reference one) and then comparing the results. One of the reasons for the failure of collaboration with other research teams was the lack of time: in future developments of the project there will be other opportunities to face the topic of validation.

The geometric simplifications of the anatomy of the head-brain system are another clear limitation of the model: as already explained, no CT or MRI image database was used to build the geometry of the model. The consequence of this choice (due to the inaccessibility of these databases) is an inaccuracy of the simulation results. The shape of the skull perfectly oval, and not similar to that of a human skull, the brain that “navigates” in the CSF, the lack of distinction between white and gray matter, the absence of meninges and axons, these are all characteristics which make the model more anatomically inaccurate than others described in the literature search. A possible development of the project could be the design of a new model, with the same physical and material characteristics adopted in this one, the same boundary condition, but with a geometry based on medical images, and then comparing the results of the two simulations.

The input of the boundary conditions is a crucial point of the project: it represents the conjunction between the two parallel topics dealt with during the development of the final project, the design of the electronic device and the imposition of the data acquired by that as input in the FEHM. However, even this crucial point contains inaccuracies. The ShockAccel system measures the accelerations in a point of the helmet (posterior region), which are then applied in the same point in the finite element model, without having any information on the real location and on the direction of the impact. This is a limit, because the shock could have occurred, for example, in the anterior region of the head, but the system, as it is structured, would apply the accelerations measured in the posterior region of the model.

This is a problem that cannot be solved with a single accelerometer and a single gyroscope, as included in ShockAccel system: an optimization problem of the number of sensors and their location to be installed on the helmet could be implemented in a future development of the project.

ShockAccel FEHM could be useful in solving this optimization problem, since, through the History Output Requests of Abaqus, the user can simultaneously monitor numerous acceleration curves of different locations (virtual inertial sensors). The literature [63] suggests the simulated annealing as optimization method.

The model also has strengths:

- the properties of the materials were taken from the literature;
- the results obtainable from the model are similar to those obtainable from other models already validated;
- the production cost of this model was very low, being based on geometric shapes and not on images;
- the total number of elements of the model, lower than that of the others, makes it less computationally demanding.

Comparisons

The main characteristics of the models described in this paper (those of the bibliographic research plus the ShockAccel FEHM, designed in this project and highlighted in rose) are summarized in Table 38.

Model	Shape geometry	Layer & components	Materials	Boundary conditions	Employment	Results that can be obtained
EEVCWG17 headform FE model	Simple	3 layers with covers, 14045 nodes, 12162 shells	Skin: vinyl and aluminum pants	Lifting speed 4rad/s, Maximum lifted height 107mm, time 20ms	Engineering field, pedestrian head	HIC
Wayne State University Head model	Very complex	245000 elements	All linear elastic material properties	Max principal strain 0,2, time 5ms	Engineering and medical field	HIC, Brain pressure, Brain Von Mises stress, Max Principal strain, MPS contour, DAI map
Human body head model (HBM)	Very complex	5 layers, 32 components, 66624 nodes, 49607 elements	Skin: vinyl (viscoelastic brain foam) and aluminum parts	Impactor mass 5,59kg, speed 9,94m/s	Engineering and medical field	HIC, Brain pressure
Simpleware's model	Very complex	All tetrahedral mesh, 3,72M volumetric elements	Simple linear elastic and equations of state, as well as more advanced hyper-viscoelastic models.	Frontal impact to the head at 45° by a padded impactor of 5,6kg mass travelling at 9,94m/s	Engineering and medical field	Stress wave sharp, Intracranial pressure
TNO Head Model	Simple	22 parts, 15637 nodes, 13868 elements	All elements by linear elastic	Lower collision speed 10-15km/h, High collision speed 30km/h	Engineering field, pedestrian head	HIC, Stress wave sharp, Intracranial pressure
THUMS FE Model	Very complex	4 layers, 24100 solids, 25200 shells, 49700 elements	Skull: solid element and spongy bone, Brain: hexagonal solid	Impactor mass 23,4kg, speed 7m/s	Engineering field, passenger and pedestrian head	Brain intracranial pressure, acceleration, HIC
ULP FEM Human Head	Complex	13208 elements (10395 brick, 2813 shell)	Brain/skull interface: linear elastic, Brain and cerebellum: elastic plastic, skull: elastic plastic brittle	Impactor speed 7,5m/s, Max compression 350kPa	Engineering field, pedestrian head	HIC, Brain pressure, Brain Von Mises stress, Global strain energy of the brain/skull interface
Politecnico di Torino Head Model	Complex	4 layers, 26000 nodes, 55264 elements	All elements by linear elastic. Brain: visco-elastic	Impactor mass 5,6kg, speed 9,6m/s, force 7,56kN	Engineering and medical field	HIC, Brain pressure, Brain shear stress, DAI
Harvard Medical School Head Model	Complex	39047 elements	All elements by Polymers	Impactor force 7,9kN	Engineering and medical field	HIC, Brain intracranial pressure, Cavitation damage, Shear stress contours
Dublin College (UCDBTM)	Simple and complex	3 layers, 7318 hexahedral brain element, 2874 element for CSF	All elements by hyperelastic for some models and others. Brain: linear elastic, tissue: non-linear elastic	Impactor mass 23,4kg, speed 7m/s, semisinusoidal force 7kN, time 6ms	Engineering field, pedestrian head	HIC, Brain shear stress, Frontal brain pressure, amount of movement of brain (hydrostatic and Von Mises)
ShockAccel FEHM	Very Simple	11067 elements (7443 tetrahedral, 3624 hexahedral), 7 parts + 2 parts for the helmet	All elements by linear elastic (except for CSF). Brain: visco-elastic. Helmet: plastic behavior	3 linear accelerations + 3 rotational accelerations imposed by ShockAccel detection	Engineering and medical field	HIC (postprocessing in Matlab), Brain pressure, Brain Von Mises stress, ICP

Table 38. Comparison among the human head models investigated in this study and ShockAccel FEHM

Conclusions

This project had two fundamental targets: the design of a portable monitoring system (ShockAccel system) capable of detecting shocks to the head of hockey players, and to transmit an alert signal to the device of the medical staff at the pitch side, and the design of a Finite Element Model of a human head (ShockAccel FEHM), with which to perform the multiphysics simulations of the shocks detected with the monitoring system. Both targets were achieved, even if with some limitations.

ShockAccel system showed comparable performance with other devices on the market: the detection algorithm is more complete than others, considering an additional variable, thus an additional threshold, such as angular acceleration, which is clinically relevant; the sampling frequency, the measurement ranges and resolutions chosen are suitable for the assessed problem; the estimated selling cost is competitive, even if the analysis carried out was incomplete.

Concerning the ShockAccel FEHM, it showed good performances in computational terms and reasonable results that may help the clinician to better analyze the dynamics of the athlete's head shock.

The biggest step forward in this project, compared to the monitoring devices on the market and compared to the FEHM designed by other research centers, was the ex-novo design of these two elements which were then combined into a single system: this was the core of the whole project. The system is presented as a support tool for diagnostics and clinical analysis, not a replacement for them.

This was a pilot project, so the aim was to develop as many things as possible during the five months of Erasmus experience and then to continue them in the future with other types of work, such as other master's theses of future students or as a topic of PHD or as papers for scientific literature.

References

- [1] Fernandes, F.A.O. and Alves de Sousa, R.J. (2015), "Head injury predictors in sports trauma—a state-of-the-art review", *Proceedings of the Institution of Mechanical Engineers. Part H, Journal of Engineering in Medicine*, Vol. 229 No. 8, pp. 592-60
- [2] Giza CC, Hovda DA. The new neurometabolic cascade of concussion. *Neurosurgery*. 2014;75(suppl 4):S24–S33.
- [3] McCrory, P., K. Johnston, W. Meeuwisse, M. Aubry, R. Cantu, J. Dvorak, T. Graf-Baumann, J. Kelly, M. Lovell, and P. Schamasch. Summary and agreement statement of the 2nd international conference on concussion in sport, Prague 2004. *Clin. J. Sport Med*. 15:48–55, 2005
- [4] Kwong Ming Tse, Siak Piang Lim, Vincent Beng Chye Tan, Heow Pueh Lee. A Review of Head Injury and Finite Element Head Models. *American Journal of Engineering, Technology and Society*. Vol. 1, No. 5, 2014, pp. 28-52.
- [5] K. L. O'Connor, S. Rowson, S. M. Duma and S. P. Broglio, "Head-Impact–Measurement Devices: A Systematic Review," *Journal of Athletic Training*, vol. 52, no. 3, pp. 206-227, 2017.
- [6] J.A. Langois, W. Rutland-Brown, and M.M. Wald. The Epidemiology and Impact of Traumatic Brain Injury - A Brief Overview. *Journal of Head Trauma Rehabilitation*, 21 (5):375{378, 2006.
- [7] A.W. Selassie, D.A. Wilson, E.E. Pickelsimer, D.C. Voronca, N.R. Williams, and J.C. Edwards. Incidence of sport-related traumatic brain injury and risk factors of severity: a population-based epidemiologic study. *Annals of Epidemiology*, 23:750{756, 2013.
- [8] Greenwald RM, Gwin JT, Chu JJ, Crisco JJ. Head impact severity measures for evaluating mild traumatic brain injury risk exposure. *Neurosurgery*. 2008; 62:789–798. [PubMed: 18496184].
- [9] Gurdjian, ES.; Lange, WA.; Patrick, LM.; Thomas, LM. Impact injury and crash protection. Thomas, CC., editor. Springfield; Illinois: 1970.
- [10] T. B. Hoshizaki, A. Post, M. Kendall, J. Cournoyer, P. Rousseau, M. D. Gilchrist, S. Brien, M. Cusimano and S. Marshall, "The Development of a Threshold Curve for the Understanding of Concussion in Sport," *Trauma*, vol. 19, no. 3, pp. 196-206, 2016.
- [11] A Test Methodology for Assessing Demining Personal Protective Equipment (PPE).
- [12] Versace, J. 1970. A review of the severity index. In *Proceedings of the 15th Stapp Car Crash Conference*, pp. 771–796.
- [13] David C. Viano. 1988. "Biomechanics of head injury: Toward a theory linking head dynamic motion, brain tissue deformation and neural trauma". SAE n° 881708.
- [14] Kathryn L. O'Connor, MS; Steven Rowson, PhD; Stefan M. Duma, PhD; Steven P. Broglio, PhD. Head-Impact–Measurement Devices: A Systematic Review. *Journal of Athletic Training* 2017;52(3):206–227 doi: 10.4085/1062-6050.52.2.05.
- [15] Prins ML, Alexander D, Giza CC, Hovda DA. Repeated mild traumatic brain injury: mechanisms of cerebral vulnerability. *J Neurotrauma*. 2013;30(1):30–38.
- [16] Rowson S., Duma S., Beckwith J., Chu J., Greenwald R., Crisco J. Rotational Head Kinematics in Football Impacts: An Injury Risk Function for Concussion, 2011.

- [17] "Simbex: HIT System." [Online]. Available: <http://www.simbex.com/hitsystem.html>
- [18] "History - Riddell." [Online]. Available: <http://www.riddell.com/history>
- [19] Crisco, Chu, Greenwald, "An Algorithm for Estimating Acceleration Magnitude and Impact Location Using Multiple Nonorthogonal Single-Axis Accelerometers", *Journal of Biomechanical Engineering* DECEMBER 2004, Vol. 126.
- [20] "ADXL193 datasheet and product info | Single-Axis, High-g, iMEMS® Accelerometers | MEMS Inertial Sensors | Analog Devices." [Online].
- [21] "MSP430 Ultra-Low Power 16-bit MCUs - 1 Series - MSP430F148 - TI.com." [Online].
- [22] Semtech Corporation, XE1203F 433 MHz / 868 MHz / 915 MHz Low-Power, Integrated UHF Transceiver
- [23] T. M. Talavage, "Interview with Tom Talavage: Real-Time Telemetry-Based Modeling for Minimization of Head Injury Risk," 11-Feb-2014.
- [24] E. L. Breedlove, M. Robinson, T. M. Talavage, K. E. Morigaki, U. Yoruk, K. O'Keefe, J. King, L. J. Leverenz, J. W. Gilger, and E. A. Nauman, "Biomechanical correlates of symptomatic and asymptomatic neurophysiological impairment in high school football," *J. Biomech.*, vol. 45, no. 7, pp. 1265–1272, Apr. 2012.
- [25] S. Foreman and D. B. Crossman, "A Comparative Analysis for the Measurement of Head Accelerations in Ice Hockey Helmets using Non-Accelerometer Based Systems," presented at the ASTM Concussion Mechanisms Symposium, 2013.
- [26] BlueGiga, WT11i_Datasheet.pdf.
- [27] MC10 Reshaping Electronics, Reebok CHECKLIGHT Overview, MC10 CONFIDENTIAL
- [28] <http://iot.appinventor.mit.edu/#/bluetoothle/bluetoothleintro>
- [29] Horgan and Gilchrist, 2003; Kleiven, 2007; Ruan et al., 1994; Sahoo et al., 2014; Takhounts and Ridella, 2008; Yang, 2011; Zhanget al., 2011.
- [30] Zhang, L., Bae, J., Hardy, W.N., Monson, K.L., Manley, G.T., Goldsmith, W., Yang, K.H. and King, A.I. "Computational study of the contribution of the vasculature on the dynamic response of the brain", *Stapp Car Crash Conference*, p. 46:145-64, 2002-22-0008 (2002).
- [31] Sunan Huanga, Jikuang Yanga, —Optimization of a reversible hood for protecting a pedestrian's head during car collisions|| paper No.42 (2010) 1136–1143 <http://www.sciencedirect.com/science/article/pii/S000145750900342X>.
- [32] Directive of European parliament relating to the protection of pedestrians, amending directive 70/156/EEC Brussels, 2003
- [33] Franklyn, Fildes, Zhang, Yang - Analysis of Finite Element Models for Head Injury Investigation: Reconstruction of Four Real-World Impacts - *Stapp Car Crash Journal*, Vol. 49 (November 2005), pp.1-32.
- [34] Yasuhiro Dokko, Jim Manavis, Peter Blumburghs, Jack McLean, Liyn Zhang, King H. Yang, Albert I. King — VALIDATION OF THE HUMAN HEAD FE MODEL AGAINST PEDESTRIAN ACCIDENT AND ITS TENTATIVE APPLICATION TO THE EXAMINATION OF THE EXISTING TOLERANCE CURVE||, Paper No. 322.
- [35] A.M. Nahum, R. Smith, C.C. Ward, — Intracranial pressure dynamics during head impact||, *Proceedings of the 21st Stapp Car Crash Conference*, Vol. SAE Paper No. 770922, 1977. <http://www.mendeley.com/research/intracranialpressure-dynamics-during-head-impact/>

- [36] YANG, XU, OTTE - BRAIN INJURY BIOMECHANICS IN REAL WORLD VEHICLE ACCIDENT USING MATHEMATICAL MODELS - CHINESE JOURNAL OF MECHANICAL ENGINEERING Vol. 21, a No. 4, a 2008.
- [37] WILLINGER R, BAUMGARTNER D. Numerical and physical modeling of the human head under impact- Towards new injury criteria [J]. International Journal of Vehicle Design, 2003, 32: 94-115.
- [38] A.M. Nahum, R. Smith, C.C. Ward, — Intracranial pressure dynamics during head impact||, Proceedings of the 21st Stapp Car Crash Conference, Vol. SAE Paper No. 770922, 1977.
<http://www.mendeley.com/research/intracranialpressure-dynamics-during-head-impact/>
- [39] P.G. Young, S.M. Qidwai, A. Bagchi, N. Kota, C. Pearce, R.T. Cotton, Developing a Finite Element Head Model for Impact Simulation in Abaqus, 2015 SIMULIA Community Conference.
- [40] 2008 International Workshop on Modelling, Simulation and Optimization, Lei Gang, Bao Chunyan, Liu Xianjue. — Finite Element Analysis of Head- Ground Collision||
- [41] Voo K, Kumaresan S, Printar FA et al. Finite element models of the human head[J]. Med Biol Eng Comput, 1996, 34(5): 375-381.
- [42] Oshita, F. et al., "Development of a Finite Element Model of the Human Body", 7th International LS-DYNA Users Conference, Detroit, 2002.
- [43] Masami Iwamoto, Yuko Nakahira, Atsutaka Tamura, Hideyuki Kimpara, Isao Watanabe, Kazuo Miki. — DEVELOPMENT OF ADVANCED HUMAN MODELS IN THUMS|| Keynote 7, 6th European LSDYNA Users Conference.
- [44] Galford, J.E., and McElhaney, J.H., "A viscoelastic study of scalp, brain and dura". Journal of Biomechanics 3, 1970, pp. 211-222.
- [45] Trosseille, X., Tarriere, C., Lavaste, F., Guillon, F., and Domont, A., "Development of a FEM. of the human head according to a specific test protocol", Proc. of 36th Stapp Car Crash Conference, SAE Paper No.922527, 1992.
- [46] Daniel Baumgartner, Daniel Marjoux, Remy Willinger, Emma Carter, Clive Neal-Sturgess, LuisGuerra, Luis Martinez, Roger Hardy: —PEDESTRIAN SAFETY ENHANCEMENT USING NUMERICAL METHODS|| paper2007.
- [47] R. Willinger, D. Baumgartner, B. Chinn, M. Neale - HEAD TOLERANCE LIMITS DERIVED FROM NUMERICAL REPLICATION OF REAL WORLD ACCIDENTS - IRCOBI Conference, Montpellier (France), September 2000.
- [48] Turquier F., Kang H.S.-Troseille X.-Willinger R.-Tarriere C.-Lavaste F., Validation study of a 3D finite element head model against experimental data, Proceed. of the 40st Stapp Car Crash Conference, Albuquerque, 1996, pp283-294
- [49] Giovanni Belingardi, Giorgio Chianidussi, Ivan Gaviglio, Politecnico di Torino, Dipartimento di Meccanica. Italy: —DEVELOPMENT AND VALIDATION OF A NEW FINITE ELEMENT MODEL OF HUMAN HEAD||. Paper Number 05-0441
- [50] Tamer El Sayed, Alejandro Mota, Fernando Fraternali, Michael Ortiz,|| Biomechanics of traumatic brain injury||, 2008http://aero.caltech.edu/~ortiz/Pubs/2008/ElSayedMotaFraternaliOrtiz2008_Biomechanics.pdf
- [51] Horgan, T. J. & Gilchrist, M. D., 2003, — The creation of three-dimensional finite element models for simulating head impact biomechanics||, International Journal of Crash worthiness, Inpress.
- [52] Anderson R.W.G.; A Study on the Biomechanics of Axonal Injury,2000

- [53] T. J. Horgan, M. D. Gilchrist, — Influence of FE model variability in predicting brain motion and intracranial pressure changes in head impact simulations|| paper No.0299,2004
- [54] P. Saboori, A. Sadegh, Material modeling of the head's subarachnoid space - *Scientia Iranica* (2011), 18 (6), 1492–1499.
- [55] The effect of viscoelasticity on fatigue behaviour of polymer matrix composites - J.A. Epaarachchi, in *Creep and Fatigue in Polymer Matrix Composites*, 2011.
- [56] Fifty Shades of Brain: A Review on the Mechanical Testing and Modeling of Brain Tissue - Silvia Budday · Timothy C. Ovaert · Gerhard A. Holzapfel · Paul Steinmann · Ellen Kuhl - *Archives of Computational Methods in Engineering*, 2019.
- [57] Smoothed Particle Hydrodynamics - Monaghan J. J. - *Annual Review of Astronomy and Astrophysics* Vol. 30:543-574 (Volume publication date September 1992).
- [58] Brain injury tolerance limit based on computation of axonal strain - Debasis Sahoo, Caroline Deck, Rémy Willinger - *Accident Analysis and Prevention* 92 (2016) 53–70.
- [59] Monitoring and interpretation of intracranial pressure - M. Czosnyka, J. D. Pickard - *J Neurol Neurosurg Psychiatry* 2004;75:813–821. doi: 10.1136/jnnp.2003.033126.
- [60] Joseph T. Gwin, Jeffery J. Chu, Thomas A. McAllister , and Richard M. Greenwald. In situ Measures of Head Impact Acceleration in NCAA Division I Men's Ice Hockey: Implications for ASTM F1045 and Other Ice Hockey Helmet Standards, *Journal of ASTM International*, Vol. 6, No. 6, 2009.
- [61] Head acceleration is less than 10 percent of helmet acceleration during a football impact – Manoogian S., McNeely D., Duma S. – ISB XXth Congress, ASB 29th Annual Meeting July 31 August 5, Cleveland, Ohio.
- [62] Analog Device, ADXL372 Datasheet
- [63] An Algorithm for Estimating Acceleration Magnitude and Impact Location Using Multiple Nonorthogonal Single-Axis Accelerometers – Crisco J., Chu J., Greenwald M. – *Journal of Biomechanical Engineering* DECEMBER 2004, Vol. 126 pag. 849-854