

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Development of speed measurement system for Split Hopkinson Pressure bar

José Pedro Ferreira de Sousa



Master's Degree in Mechanical Engineering - Automation Specialization

Supervisor: Pedro José da Silva Carvalho Pereira de Sousa

Co-supervisor: Gonçalo Filipe Pina Cipriano

Second Co-supervisor: Pedro Miguel Guimarães Pires Moreira

September 29, 2023

Development of speed measurement system for Split Hopkinson Pressure bar

José Pedro Ferreira de Sousa

Master's Degree in Mechanical Engineering - Automation Specialization

September 29, 2023

Resumo

A caracterização de materiais é uma área de engenharia que oferece os dados necessários para se fazer a melhor escolha de que material usar, dependendo da aplicação pensada, sendo necessário ter em conta condições do meio ambiente, rotina de uso, cargas aplicadas e longevidade da estrutura. Para fazer esta caracterização, os materiais têm de ser testados para perceber os limites de cada um. Para isto pode-se usar vários aparatos dependendo das características que se queiram testar.

A barra de Hopkinson é um aparato com o objetivo de testar a capacidade de resposta de um material a altas taxas de deformação, consistindo num ensaio de impacto, onde o provete pode ser comprimido ou tensionado de forma brusca, dependendo da configuração usada no aparato. Para analisar este ensaio, pode-se usar extensómetros colados às barras, detetando, assim, a deformação elástica, devido a um impacto inicial, que se propaga pelo aparato, e pode-se usar, também, uma câmara de alta velocidade para filmar o comportamento do provete ao longo do ensaio, analisando-se, depois, as imagens gravadas. Este processo de análise é relativamente extenso e demorado, o que causa gastos de tempos necessários para a correta e aprofundada compreensão do ensaio. No entanto, se os resultados dessa análise concluírem que o ensaio foi mal realizado, será necessário realizar de novo os mesmos testes e fazer uma nova análise após os mesmos. Isto causa um gasto de tempo muito grande, e faz com que não se tenha muita confiança no ensaio realizado.

Para agilizar este processo, é necessário obter mais informações aquando da realização do ensaio, de maneira a se conseguir perceber se o mesmo foi bem realizado ou se é necessário repeti-lo. Assim, precisa-se de escolher alguma variável do ensaio que possa ser calculada e confirmada expeditamente. Uma possível escolha é a velocidade das barras, mais em particular, a velocidade da barra que inicia o ensaio. Sabendo que esta barra é disparada por um sistema pneumático, não se pode considerar que a relação entre a pressão do sistema e a velocidade da barra disparada seja objetiva. Com esta informação, consegue-se ter uma ideia se o ensaio foi bem realizado ou não. Desta maneira, torna-se interessante desenvolver um sistema para monitorizar a velocidade linear a que a barra de início é disparada.

Assim, este trabalho tem como objetivo principal o desenvolvimento de um sistema de monitorização de velocidade linear da barra de início de ensaio da barra de Hopkinson. Neste trabalho, descreve-se o processo de desenvolvimento do sistema, discutindo métodos e equipamentos a serem utilizados e explicando programas implementados para um correto funcionamento do sistema, assim como se analisa resultados de testes executados com o aparato em causa de maneira a validar o funcionamento do sistema.

Abstract

The characterization of materials is an engineering area that offers the necessary data to make the best choice of which material to use, depending on the intended application, being necessary to consider environmental conditions, routine use, applied loads and longevity of the structure. To make this characterization, the materials must be tested to understand the limits of each one. For this, you can use various devices depending on the characteristics one would want to test.

The Split Hopkinson Pressure bar is an apparatus with the objective of testing the response of a material to high strain rates, consisting of an impact test, where the specimen can be compressed or tensioned abruptly, depending on the configuration used in the apparatus. To analyze this test, strain gauges glued to the bars can be used, thus detecting the elastic deformation, due to an initial impact, that propagates through the apparatus. Also, a high-speed camera can be used to film the behavior of the specimen throughout the test, followed by an analysis of the recorded images. This analysis process is relatively extensive and time-consuming, which causes expenditure of time necessary for the correct and in-depth understanding of the test. However, if the results of this analysis conclude that the test was poorly performed, it will be necessary to perform the same tests again and make a new analysis after them. This wastes a lot of time and makes the confidence in the test performed to be lower.

To speed up this process, it is necessary to obtain more information when carrying out the test, in order to be able to understand if it was well performed or if it is necessary to repeat it. Thus, one needs to choose some variable that can be calculated and confirmed expeditiously. A possible choice is the speed of the bars, more in particular, the speed of the bar that starts the test, the striker bar. Knowing that this bar is triggered by a pneumatic system, it is not possible to consider that the relationship between the pressure of the system and the speed of the bar is a direct one. With this information, it is possible to have an idea if the test was well performed or not. In this way, it becomes interesting to develop a system to monitor the linear velocity at which the striker bar is shot.

Thus, this work has as its main objective the development of a system for monitoring the linear velocity of the striker bar of the Split Hopkinson Pressure bar test. In this work, the system development process is described, discussing methods and equipment to be used and explaining programmed and incorporated codes for the correct functioning of the system, as analyzing results of tests performed with the apparatus in question in order to validate the functioning of the system.

Acknowledgments

The authors acknowledge the project POCI-01-0247-FEDER-072260, financed by European Funds, through program COMPETE2020, under the Eureka smart label S0318-STREAM -Surface Treatment for Additive Manufacturing.

I would like to begin by thanking Professor Pedro José Sousa, and Gonçalo Cipriano, for making this work opportunity available to me as the subject of my thesis dissertation. I want to thank all the support, guidance, patience, and willingness to work together that have been provided along this path of mine. I also want to thank the UMAI team for the occasional support and effort to integrate me as a student in this unit where I worked.

I would like to express my gratitude to all the professors I encountered during my academic journey, each of whom taught me what they had to teach, whether on a technical, professional, political or even personal level.

I would like to thank all my colleagues and friends from other FEUP courses, with whom I have made many memories in this long journey of five years, where we have been studying, celebrating, and sharing life stories.

I want to thank my companions in the master's specialization, who accompanied me these two years. During this time, I often found myself stepping out of my comfort zone, as I was the only one among my closest group of friends who chose this particular field. I believe it was a fluke that provided me with challenges that forced me to discover new horizons and create tools for teamwork.

I want to thank my high school friends who accompanied me since the beginning of my journey, long before I entered college. To those with whom I continued to create memories, thank you very much for your support.

I want to thank the friends I made in the first moments of starting the course, who accompanied me throughout my academic journey. I had many good times with them that I will never forget. I created friendships that I know will accompany me for the rest of my journey. It was with them that I managed to get the strength to move forward on my journey, because I knew we were all in the same boat. It was a pleasure and a privilege to have shared moments with them, like the ones I had.

Finally, my most heartfelt and profound gratitude goes to my family, who consistently made every effort to assist, support, and educate me. I want to express my thanks for the patience you've shown me over these five years and for providing the ideal conditions for me to excel to the best of my abilities.

Contents

Contents	iv
List of Figures	vi
List of Tables	ix
Abbreviations	x
1 Introduction	1
1.1 Background and motivation	1
1.2 Objectives	1
1.3 Thesis structure	2
2 State of the art	3
2.1 High strain rate tests	3
2.2 Split Hopkinson Pressure bar	5
2.2.1 Historical background	5
2.2.2 Working principle	7
2.3 Projectile speed sensing methods	13
2.3.1 Electromagnetic field effect	14
2.3.2 Doppler effect	15
2.3.3 Optical sensor	17
2.3.4 Proposed solution	20
2.3.5 Applied Digital Image Correlation	21
3 System Development	24
3.1 System Requirements	24
3.2 Optic sensor	25
3.3 Timing hardware	27
3.3.1 Method and components selection	27
3.3.2 Microcontroller Input Capture Unit	28
3.4 Firmware Development	30
3.5 Board Validation	35
3.5.1 Pulse Width Modulation	35
3.5.2 PWM testing	37
3.5.3 Signal Smoothing	44
3.6 User interface and support components	45
3.6.1 User Interface	45
3.6.2 Support components	49

4	System Validation	51
4.1	Validation results with 40 mm bands	52
4.1.1	DIC results	55
4.1.2	Oscilloscope results	60
4.1.3	Sensing system results	65
4.1.4	Discussion	67
4.2	Validation results with 5 mm bands	68
4.2.1	DIC results	68
4.2.2	Oscilloscope results	73
4.2.3	Sensing system results	78
4.2.4	Discussion	81
5	Conclusion	83
5.1	Conclusions	83
5.2	Future work	84
	References	85
A	Developed codes	90
A.1	Final Arduino code	90
A.2	Windows Forms code	92

List of Figures

2.1	High strain rates compression tests: a) Taylor impact test setup (adapted from [7]); b) Expected Taylor impact test specimen shape after impact [4]; c) Gas gun test setup (adapted from [9]).	4
2.2	High strain rates tensile tests: a) Flyer plate principle [10]; b) Expanding ring simplified setup [11].	5
2.3	B. Hopkinson's apparatus [14]	6
2.4	Kolsky's apparatus [14]	6
2.5	Split Hopkinson Pressure bar schematic: a) Compression setup; b) Tensile setup (adapted from [16])	7
2.6	SHPB tensile setup impact wave propagation representation (adapted from [13]) .	9
2.7	Example of a graph of the SHPB strain gauges responses over the time of a test .	9
2.8	Representation of an infinitesimally small body at impact [17]	10
2.9	Representation of the impact between two bars (adapted from [2])	11
2.10	Strain and stress behavior representation in apparatus' bars and specimen (compression setup example) [17]	12
2.11	Electromagnetic coils setup representation (adapted from [18])	14
2.12	Doppler effect representation (adapted from [20])	15
2.13	Doppler continuous wave radar representation (adapted from [29])	16
2.14	PDV setup representation (adapted from [33] and [31])	17
2.15	Through-beam setup representation (adapted from [37])	18
2.16	Retro-reflective setup representation (adapted from [37])	18
2.17	Diffuse-reflection setup representation (adapted from [37])	18
2.18	REL Inc mounting on apparatus speed sensor (adapted from [38])	20
2.19	Proposed solution: a) method representation; b) expected signals examples. . . .	21
2.20	Schematic representation of a reference subset and a target subset following deformation (adapted from [39])	22
2.21	Speckle applied on specimen and DIC results example (adapted from [13])	22
2.22	DIC planar specimen setup example (adapted from [13])	23
2.23	DIC example: point tracking and respective speed calculation (GOM Correlate software)	23
3.1	a) TCRT5000 sensor; b) Schematic top view of TCRT5000 sensor; c) Test conditions (adapted from [41]).	26
3.2	TCRT5000 module with analog pin [42]	26
3.3	Sensor module's simplified circuit representation	27
3.4	Arduino Mega2560 from JOY-iT [52]	29
3.5	Connection between Arduino board and sensor module circuit representation . .	30
3.6	Work pattern and firmware's behavior to transitions	31

3.7	Working methods of transitions detection	32
3.8	Arduino firmware main flowchart	33
3.9	Arduino firmware secondary flowcharts: a) Preparation for interrupts routine; b) Input capture interrupt routine; c) Overflow counter routine.	34
3.10	PWM digital signal and average output representation [59]	35
3.11	Duty cycle examples [55]	36
3.12	PWM testing circuit representation	38
3.13	PWM testing circuit with button representation	40
3.14	PWM test with button oscilloscope result	41
3.15	PWM test with button ICU result	41
3.16	Continuous PWM test oscilloscope result	42
3.17	Continuous PWM test oscilloscope result detail	42
3.18	Continuous PWM test ICU result	43
3.19	Final circuit after signal smoothing	44
3.20	Developed interface	46
3.21	User's interface software code main flowchart	47
3.22	User's interface software code secondary flowcharts: a) Connection routine; b) Processing data routine; c) Exporting data to a .csv file routine.	48
3.23	Developed support components: a) Sensor support CAD model; b) Printed sensor support; c) Arduino support CAD model; d) Printed Arduino support.	50
3.24	Supports for the sensor module and Arduino mounted on the profile of the SHPB structure	50
4.1	Pattern and sensor setup on SHPB striker bar	52
4.2	Test setup with high speed camera and high power LED light	52
4.3	Sensor system results for the first executed test, without camera nor LED, at 2.5 bar with 40 mm bands	53
4.4	Connection between oscilloscope and sensor module circuit representation	54
4.5	DIC results for the first test at 10 bar with 40 mm bands	56
4.6	DIC results for the first test at 10 bar with 40 mm bands	56
4.7	Filtered DIC results for the first test at 10 bar with 40 mm bands	57
4.8	DIC results for the second test at 10 bar with 40 mm bands	58
4.9	Filtered DIC results for the second test at 10 bar with 40 mm bands	58
4.10	DIC results for the first test at 2.5 bar with 40 mm bands	58
4.11	Filtered DIC results for the first test at 2.5 bar with 40 mm bands	59
4.12	DIC results for the second test at 2.5 bar with 40 mm bands	59
4.13	Filtered DIC results for the second test at 2.5 bar with 40 mm bands	59
4.14	DIC data analysis routine	60
4.15	Oscilloscope results for the first test at 10 bar with 40 mm bands	61
4.16	Processed oscilloscope results for the first test at 10 bar with 40 mm bands	62
4.17	Oscilloscope results for the second test at 10 bar with 40 mm bands	62
4.18	Processed oscilloscope results for the second test at 10 bar with 40 mm bands	62
4.19	Oscilloscope results for the first test at 2.5 bar with 40 mm bands	63
4.20	Processed oscilloscope results for the first test at 2.5 bar with 40 mm bands	63
4.21	Oscilloscope results for the second at 2.5 bar with 40 mm bands	63
4.22	Processed oscilloscope results for the second test at 2.5 bar with 40 mm bands	64
4.23	Oscilloscope data analysis routine	65
4.24	Sensor system results for the first test at 10 bar with 40 mm bands	65
4.25	Sensor system results for the second test at 10 bar with 40 mm bands	66

4.26	Sensor system results for the first test at 2.5 bar with 40 mm bands	66
4.27	Sensor system results for the second test at 2.5 bar with 40 mm bands	66
4.28	DIC results for the first test at 10 bar with 5 mm bands	69
4.29	Filtered DIC results for the first test at 10 bar with 5 mm bands	69
4.30	DIC results for the second test at 10 bar with 5 mm bands	70
4.31	Filtered DIC results for the second test at 10 bar with 5 mm bands	70
4.32	DIC results for the first test at 2.5 bar with 5 mm bands	70
4.33	Filtered DIC results for the first test at 2.5 bar with 5 mm bands	71
4.34	DIC results for the second test at 2.5 bar with 5 mm bands	71
4.35	Filtered DIC results for the second test at 2.5 bar with 5 mm bands	71
4.36	DIC results for the third test at 2.5 bar with 5 mm bands	72
4.37	Filtered DIC results for the third test at 2.5 bar with 5 mm bands	72
4.38	Oscilloscope results for the first test at 10 bar with 5 mm bands	73
4.39	Processed oscilloscope results for the first test at 10 bar with 5 mm bands	74
4.40	Oscilloscope results for the second test at 10 bar with 5 mm bands	74
4.41	Processed oscilloscope results for the second test at 10 bar with 5 mm bands . . .	74
4.42	Oscilloscope results for the third test at 10 bar with 5 mm bands	75
4.43	Processed oscilloscope results for the third test at 10 bar with 5 mm bands	75
4.44	Oscilloscope results for the first test at 2.5 bar with 5 mm bands	75
4.45	Processed oscilloscope results for the first test at 2.5 bar with 5 mm bands	76
4.46	Oscilloscope results for the second test at 2.5 bar with 5 mm bands	76
4.47	Processed oscilloscope results for the second test at 2.5 bar with 5 mm bands . .	76
4.48	Oscilloscope results for the third test at 2.5 bar with 5 mm bands	77
4.49	Processed oscilloscope results for the third test at 2.5 bar with 5 mm bands	77
4.50	Sensor system results for the first test at 10 bar with 5 mm bands	78
4.51	Sensor system results for the second test at 10 bar with 5 mm bands	78
4.52	Sensor system results for the third test at 10 bar with 5 mm bands	79
4.53	Sensor system results for the first test at 2.5 bar with 5 mm bands	79
4.54	Sensor system results for the second test at 2.5 bar with 5 mm bands	79
4.55	Sensor system results for the third test at 2.5 bar with 5 mm bands	80

List of Tables

2.1	Types of strain tests and testing techniques (adapted from [1])	3
3.1	Corresponding resolution times for different transition lengths at 10 m/s	25
3.2	Table of requirements and limitations	25
3.3	Input Capture Pins in Arduino Mega2560 [44,51,52]	29
3.4	PWM pins [44,51,52]	37
3.5	Microcontroller's timers and respective PWM pins [44,51,52]	38
3.6	Prescalers and respective clock speeds	39
4.1	Lowpass filter conditions	57
4.2	DIC analysis maximum speeds from each test with 40 mm bands	57
4.3	Oscilloscope analysis maximum speeds from each test with 40 mm bands	61
4.4	Sensor system analysis maximum speeds from each test with 40 mm bands	65
4.5	DIC analysis maximum speeds from each test with 5 mm bands	69
4.6	Oscilloscope analysis maximum speeds from each test with 5 mm bands	73
4.7	Sensor system analysis maximum speeds from each test with 5 mm bands	78

Abbreviations and Symbols

Acronyms

3D	Three Dimensional
ARM	Advanced RISC Machine
AVR	Advanced Virtual RISC
CAD	Computer-aided Design
CPU	Central Processing Unit
CS	Clock Selections
DAQ	Data Acquisition
DIC	Digital Image Correlation
GPIO	General Purpose Input/Output
ICP	Input Capture pin
ICRn	Input Capture Register from timer "n"
ICU	Input Capture Unit
IDE	Integrated Development Environment
LED	Light-emitting Diode
PDV	Photon Doppler Velocimetry
PWM	Pulse Width Modulation
RAM	Random Access Memory
ROM	Read Only Memory
SHPB	Split Hopkinson Pressure bar
TCCR	Timer/Counter Control Register
TDC	Time-to-digital converters
TOF	Time-of-flight
T _{on}	Time that is on
USB	Universal Serial Bus
V _{avg}	Average voltage
VCC	Common Collector Voltage
VISAR	Velocity Interferometer System for Any Reflector
V _{max}	Maximum voltage

Symbols

b	Apparatus' bar
C_0	Elastic stress wave velocity in the apparatus' striker bar material
C_b	Elastic stress wave velocity in an apparatus' bar material
C_s	Elastic stress wave velocity in the specimen material
E	Young's modulus
F	Acting force
f	Dynamic stress per unit length
l	Apparatus' striker bar length
l_s	Specimen length
n	Amount of wave reflection inside the specimen
T	Time period for the specimen to attain the state of stress equilibrium
t	Time variable
Δt	Time interval
u	Axial displacement
v_1	Speed of the incident bar tip
v_2	Speed of the transmission bar tip
v_b	Speed of an apparatus' bar
v_i	Speed of the incident bar
$v_{\max}$	Maximum speed
v_{sb}	Speed of the striker bar
w	Bar's speed after an impact
x	Length variable
Δx	Length interval
$\dot{\epsilon}$	Strain rate
ϵ	Strain
ϵ_b	Strain of an apparatus' bar
ϵ_i	Initial strain of the incident bar
$\epsilon_i^{\text{calculated}}$	Calculated strain of the incident bar
$\epsilon_i^{\text{measured}}$	Measured strain of the incident bar
ϵ_r	Strain of the incident bar due to the reflected wave
ϵ_s	Specimen strain
ϵ_{s1}	Specimen strain on the incident bar tip
ϵ_{s2}	Specimen strain on the transmission bar tip
ϵ_t	Strain of the transmission bar
ρ	Density
σ	Stress

Chapter 1

Introduction

In this chapter, the subject of this dissertation is introduced. Therefore, this chapter contains a brief background and motivation of developed work, its goals and the organization of the present document.

1.1 Background and motivation

The Split Hopkinson Pressure bar test is very important for the characterization of materials since it allows to obtain strain rate values of materials in an easy way. In addition, it allows several configurations of tests, thus giving the possibility of carrying out tests in different conditions. However, this test is dependent on external methods to analyze its results. This analysis is usually done after the tests are carried out and is quite extensive, given the parameterization it needs.

This means that a lot of time is spent in making sure that the tests were done correctly, and if any of them fails, it is required a new repetition of the process, implying an even greater waste of time. This way, it would be interesting to have a system that captures information from this test in an expeditious way to check if the tests are being carried out correctly, at the time of testing.

A parameter that can be easily captured and analyzed is the speed of the apparatus' bars, namely the bar that starts the test, the striker bar. This way, it is possible to see if the test was executed well, that is, at the necessary speed, or if an error occurred, and if this error is due to the analyzed bar. Also, with more sources of information, the perception of the need for some maintenance on the equipment becomes more effective, making the whole process more efficient.

1.2 Objectives

This work intends to develop a system for measuring the speed of the striker bar of the apparatus in question. This system should be compact and easy to replicate, so that it can be installed in places other than the striker bar.

This system must calculate the speed of the analyzed bar without involving methods that disturb its normal functioning, so any physical contact between the speed system and the apparatus

must be avoided. This system should analyze the velocity of the projected bar in a way to calculate its real displacement and velocity, avoiding indirect correlations or inferences.

This system should be fast enough to analyze and calculate the speed of the striker bar and provide its values in an expeditious and intuitive way for the user.

1.3 Thesis structure

This document is divided into five main chapters.

The first chapter contains an introduction to the problem to be addressed, presenting the background and motivation of the project and its objectives. It also explains the structure of the document and the work carried out.

In the second chapter, the state of the art is presented in the subjects of the various types of high strain rate tests, presenting more in depth the Split Hopkinson Pressure bar and its configurations for each intended purpose, and calculations that are involved in order to understand the strain that the apparatus specimen suffers. Yet, projectile sensing methods were studied in order to understand which should be used in this work, to achieve the defined objectives, and calculate the speed of the apparatus' striker bar in a efficient way. Also, in this part, a dedicated commercial solution was compared to the proposed one.

The third chapter describes the entire development process of the sensing system for calculating the projected bar speed, setting the requirements for the system, discussing the available options for the sensor to be used, and also the acquisition and processing hardware and methods, to choose the combination that best suits this project. Developed algorithms, firmware codes for the acquisition stage and graphical interface are also explained, and support components are shown across the modelation and assembly phases.

In the fourth chapter, the results of the sensing system are compared and discussed to reference values, in order to validate the developed sensing system. The reference values were obtained from the use of a high frame rate camera, and a oscilloscope to ensure the sensor's values are correct. The process routines of the reference values and test execution are explained in order to be possible to compare the results of the three methods.

In the fifth chapter, conclusions made throughout the work are summarized and discussed, as an enumeration of suggestions for possible future work.

Chapter 2

State of the art

2.1 High strain rate tests

For planning and predicting the performance of structures under diverse loading scenarios, an in-depth knowledge of engineering materials is always essential [1]. Most materials' responses might vary depending on the loading conditions. As a result, strain rate sensitivity is a key issue that concerns the yield strength, failure modes, and fracture toughness of structures in numerous industries. [1]

Regarding loading scenarios, we can categorize those scenarios in three groups regarding to the strain rate of the test. Table 2.1 indicates those categories, how they differ from one another, and what apparatus are commonly used in each scenario. [1]

Table 2.1: Types of strain tests and testing techniques (adapted from [1])

Testing technique	Applied Strain Rate (s^{-1})		
	Low Strain Rate ($\leq 10^1$)	Medium Strain Rate ($10^1 - 10^3$)	High Strain Rate ($\geq 10^3$)
Compression Tests	Conventional load frame and servo-hydraulic machine	Special Servo hydraulic load frames, drop tower, Fly-wheel device	Split Hopkinson pressure bar, Gas gun, Taylor impact test
Tensile Tests			Split Hopkinson bar in tension, Flyer plate, expanding ring

For high strain rates compression tests, all the three examples function in similar ways.

In the Split Hopkinson Pressure bar (SHPB) for compression test, the striker bar is shot by, for example, a pneumatic system onto the incident bar. The latter then crushes the specimen that is in between this incident bar and a transmission one. The impact between the striker bar and the incident bar generates a wave that propagates through the entire apparatus. This mechanical wave is detected through strain gauges. This method will be discussed in more depth in section 2.2. [2, 3]

On another side, the Taylor impact test consists on shooting a specimen, with the shape of cylinder, onto a solid immovable object, and understand how the impact deforms the specimen. It is expected that the specimen ends up with a shape similar to the object in figure 2.1.b). Then, curves of the specimen last shape are correlated to extract the needed data. This test setup is represented in figure 2.1.a). [4–7]

The gas gun is more a tool than a method, since it can be used to power a Split Hopkinson Pressure bar, to shoot a Taylor specimen [8] or to execute a different test, like shooting steel balls with 500 μm of diameter onto a cubic specimen to study its cavities and ductility [9]. A gas gun setup is represented in figure 2.1.c).

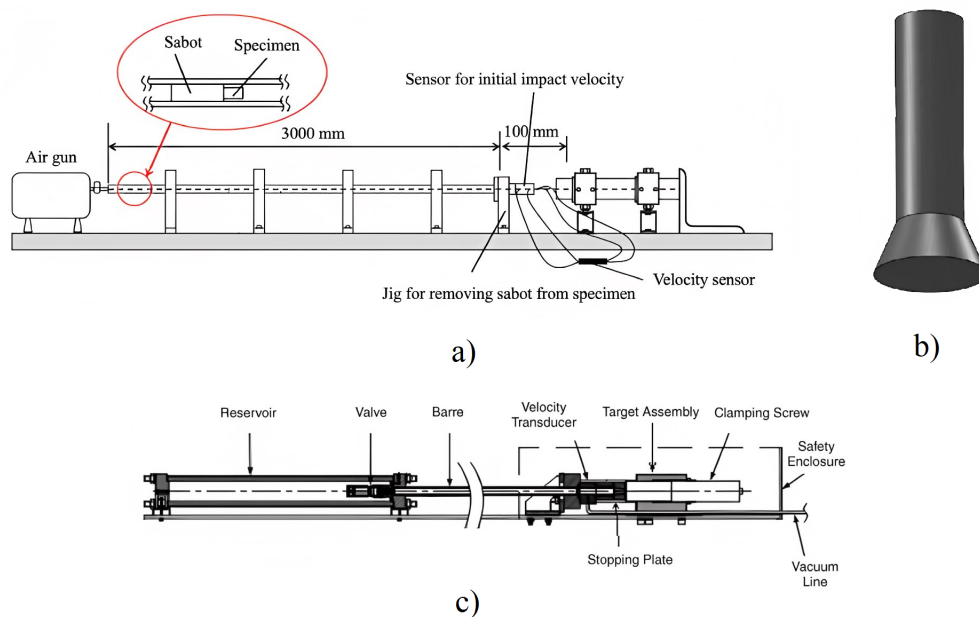


Figure 2.1: High strain rates compression tests: a) Taylor impact test setup (adapted from [7]); b) Expected Taylor impact test specimen shape after impact [4]; c) Gas gun test setup (adapted from [9]).

For tensile tests, the presented methods have less similarities between each other, than compression tests had. The SHPB tensile test works with same principle as in the compression mode, but instead of shooting the striker bar onto the incident one, so the specimen was crushed, the first bar shoots the incident bar away from the specimen. Since the specimen is attached to the incident bar, and to a third one which is at rest, called transmission bar, it will follow the incident bar movement and be stretched and/or broken . [2, 3]

On the flyer-plate method, a thin layer of the material in study is in between two plates and are positioned in a certain angle. Then, a thinner third plate is shot against the plates that are protecting the material. This principle is exemplified in figure 2.2 a). Two compressive stress waves are produced during impact. Since the flyer plate is thinner than the front plate, the reflected

unloading waves from the flyer and target plates' respective free surfaces collide at the "spall" plane in the target plate, putting tensile stress on the material there. [10]

On the expanding ring method, a ring of the material in study, usually an alloy with magnetic properties, is placed around a coil, as exemplified in figure 2.2 b), which will impose an electromagnetic field. The interaction between the specimen's current and its magnetic field, as well as the solenoid's current, generates a uniform outward force on the specimen ring, causing it to expand. The expansion is visually captured using flashlamps and a framing camera, while the speed of expansion is measured with a Velocity Interferometer System for Any Reflector (VISAR). [11, 12]

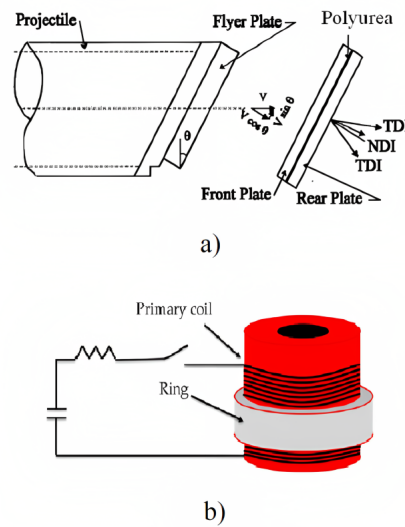


Figure 2.2: High strain rates tensile tests: a) Flyer plate principle [10]; b) Expanding ring simplified setup [11].

2.2 Split Hopkinson Pressure bar

As it was seen, the Split Hopkinson Pressure bar is very versatile for high strain rates testing, since it can be set up for compression and tensile tests. Allied with strain gauges and high frame rates cameras, it turns a very reliable method for these types of tests, being one of the most widely used techniques for determining the dynamic material characteristics under high strain rates. [2]

As previously mentioned, the SHPB, also often referred to as Kolsky bar, is a method for characterizing materials using mechanical wave propagation in an elastic medium at strain rates between 10^2 s^{-1} and 10^4 s^{-1} . By using the SHPB approach, a variety of engineering materials, including metals, composites, polymers, foam, and others, may be characterized. [2, 13]

2.2.1 Historical background

J. Hopkinson, in 1872, conducted studies on the failure of iron wires under different loading conditions. His research laid the foundation for understanding the behavior of materials under

dynamic loading. [14, 15]

Building upon his father's research, B. Hopkinson made significant contributions to the field. In 1914, he developed the Hopkinson Pressure bar technique, represented in figure 2.3, as a means to measure pressure resulting from collisions or explosions. The technique involved studying pressure pulses traveling through cylindrical rods and reflecting as tension pulses. This enabled the analysis of various factors influencing pressure measurements. [14, 15]

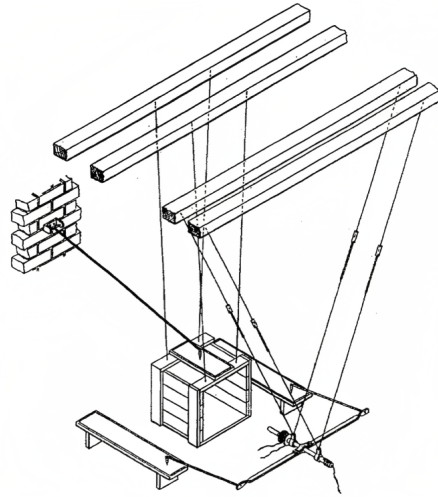


Figure 2.3: B. Hopkinson's apparatus [14]

Nearly four decades later, in 1949, Kolsky further advanced the field by introducing the "Kolsky bar", represented in figure 2.4, a modified version of the Hopkinson Pressure bar. Kolsky's focus was on studying the mechanical properties of materials under high-rate loading. The Kolsky bar apparatus consisted of a pressure bar and an extension bar connected by a brass collar. Thin specimens were placed between the bars, and measurements of axial and radial displacements were taken using condenser microphones. [14, 15]

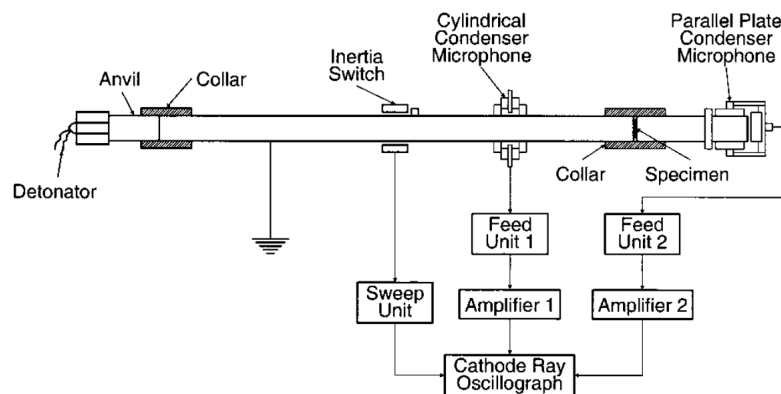


Figure 2.4: Kolsky's apparatus [14]

Kolsky's work included deriving equations to convert experimental data into stress and strain values, considering the effects of inertia and relaxation in materials. The theory behind the Kolsky bar assumed a plane compression pulse without distortion in a linear cylindrical bar, with uniform axial pressure and linear radial stress distribution. Kolsky emphasized the importance of using sufficiently thin specimens to ensure accurate measurements. [14, 15]

These techniques have shaped the field and provided valuable insights into the behavior of materials under high-rate loading, paving the way for further advancements in experimental analysis and understanding of material properties. [14, 15]

2.2.2 Working principle

The SHPB primarily consists of two relatively long solid bars, a loading mechanism, that is often a pressurized air chamber, and a data collection system. A test specimen must be put between two solid bars to allow the propagation of stress waves in order to evaluate the dynamic material characteristics, as represented in figure 2.5. In this method, the specimen experiences plastic deformation as a result of a mix of elastic strain energy and kinetic energy conveyed by stress waves. [2, 3, 13, 14]

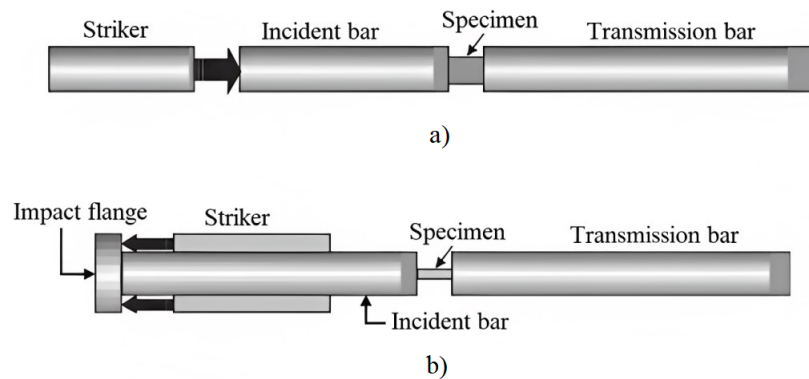


Figure 2.5: Split Hopkinson Pressure bar schematic: a) Compression setup; b) Tensile setup (adapted from [16])

The working principle of compression and tensile setups is the same. Each has a pressurized air-powered gas gun that shoots the striker bar at a defined air pressure, that will correspond to a shooting speed. The shot striker bar will collide with an incident bar, causing the latter to move in the same direction as the former. This collision causes the flange to generate a longitudinal compressive stress pulse, which is immediately reflected as a tension wave. This mechanical wave will propagate throughout the entire apparatus. [2, 3, 13, 14]

The specimen under investigation is connected to both the incident and transmission bars. The latter is stationary with non-negligible inertia. As a result, once the impact is applied to the incident bar, the specimen movement begins to influence its shape. The impact can be calculated so that the specimen either breaks or deforms. Because the specimen is also attached to the transmission bar,

the latter will experience impact-induced motion and will serve as mean to continue the generated mechanical wave propagation. [2, 3, 13, 14]

Wave propagation begins with the striker's impact and propagates towards the specimen. When the wave passes through the specimen, deforming or breaking it, the wave direction alters. Part of the wave's energy will continue to propagate in the same direction, passing through the transmission bar. The remaining energy will cause a wave to propagate backwards through the incident bar. This is the result of the recoil from compression towards the specimen, which in the compressive setup, it is caused by the impact of the incident bar on the specimen, whereas in the tensile setup, it is caused by the resisting force from the specimen when breaking. [2, 3, 13, 14]

The SHPB does not require a load cell to quantify the stress value within the test specimen or any direct feedback from the deforming sample. Strain gauges are positioned on the surface of the pressure bars, so the wave behavior interacting at specimen/bar interfaces is captured to determine the material's dynamic reaction. As the stress waves move along the elastic bars, it is intended to have minimal wave dispersion. [2, 3, 13, 14]

The impact wave propagation can be analyzed in figures 2.6, where its time evolution through the apparatus is described. It is, also, represented the strain gauges response to the waves, where the elastic deformations on the incident and transmission bars are seen. In the first bar, it is represented the incident strain, ϵ_i , that is the initial strain that is detected in the test, and the reflected strain, ϵ_r , that is the strain associated with the rebound wave. In the second bar, it is represented the transmitted strain that is detected after part of the initial wave passes the specimen [2, 3, 13, 14]. Figure 2.7 is an example of a graph of the SHPB strain gauges responses over the time of a test. The green curve represents the response of the strain gauge of the incident bar, where you can see the initial deformation after the striker impact, as soon as it starts to rise, and the deformation of the energy reflected by the transmission bar, represented by the negative part of the curve. The yellow curve represents the response of the transmission bar strain gauge, where you can see the deformation of the transmission bar from the moment the green curve starts to descend.

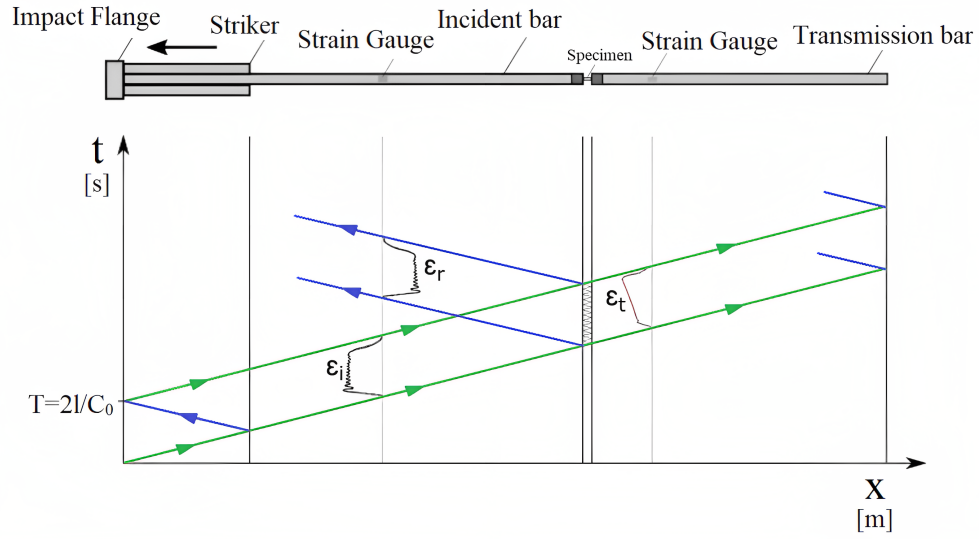


Figure 2.6: SHPB tensile setup impact wave propagation representation (adapted from [13])

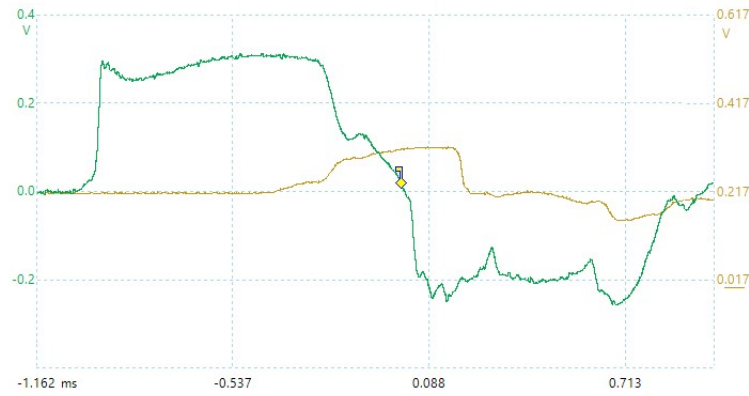


Figure 2.7: Example of a graph of the SHPB strain gauges responses over the time of a test

When studying SHPB tests, some background calculations and demonstrations need to be addressed, to know how the system works.

At any impact, it is possible to analyze an infinitesimally small portion of the impacted body, as represented in figure 2.8. The behavior of this portion can be represented and simplified in equation 2.1. [2, 13, 14]

$$\frac{\partial \sigma}{\partial x} = \rho \frac{\partial^2 u}{\partial t^2} \quad (2.1)$$

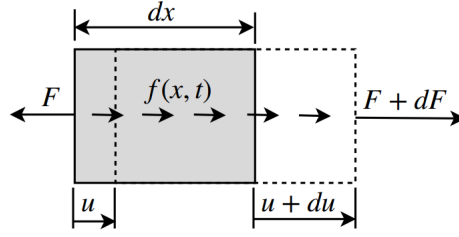


Figure 2.8: Representation of an infinitesimally small body at impact [17]

One can rewrite this expression taking into account Hooke's law (equation 2.2) and definition of deformation's definition (equation 2.3), resulting in equation 2.4. [2, 13, 14]

$$\sigma = E\varepsilon \quad (2.2)$$

$$\varepsilon = \frac{\partial u}{\partial x} \quad (2.3)$$

$$\frac{\partial}{\partial x} \left[E \frac{\partial u}{\partial x} \right] = \rho \frac{\partial^2 u}{\partial t^2} \quad (2.4)$$

By simplifying equation 2.4 into equation 2.5, it is possible to arrive at the mathematical representation of the velocity of a one-dimensional elastic wave, C_b , where b stands for bar (equation 2.6), which is dependent on the Young's modulus and the density of the material in question. [2, 13, 14]

$$\frac{E}{\rho} \frac{\partial^2 u}{\partial x^2} = \frac{\partial^2 u}{\partial t^2} \quad (2.5)$$

$$C_b = \sqrt{\frac{E}{\rho}} \quad (2.6)$$

With this wave velocity property, it is possible to relate the speed of each bar to its elastic deformation (equation 2.7).

$$v_b(t) = -C_b \varepsilon_b(t) \quad (2.7)$$

It is possible to calculate the initial strain as a function of the striker bar speed, to help with setup calibration.

Consider two long bars that will collide with each other, with speeds v_1 and v_2 , where the first bar is in motion, with $v_1 = v_{sb}$ that is the striker bar velocity and it isn't null, and the second bar is at rest, so v_2 is null. After impact, the bars will move in opposite directions but will have the same velocity w , as it is represented in figure 2.9. [2]

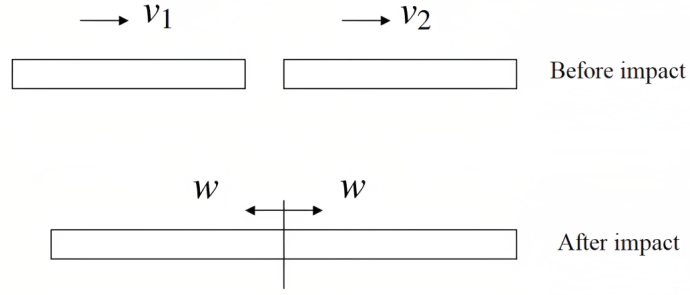


Figure 2.9: Representation of the impact between two bars (adapted from [2])

It is then possible to say that [2]

$$v_1 - w = v_2 + w \quad (2.8)$$

$$w = \frac{v_1 - v_2}{2} \quad (2.9)$$

which, when applying the mentioned conditions, makes

$$w = \frac{v_{sb}}{2} \quad (2.10)$$

Thus, it is possible to state that the velocity of the incident bar after impacted by the striker bar is half the initial velocity of this same bar. This way, the elastic deformation of the incident bar (ϵ_i) can be calculated, after impact with the striker and before it affects the specimen, as seen in equation 2.11 [2]. With this value, it is possible to establish a strain correction factor, dividing the calculated value by the measured strain value ($\epsilon_i^{calculated} / \epsilon_i^{measured}$), to help with the calibration of setup. [13]

$$\epsilon_i = \frac{-v_{sb}}{2C_b} \quad (2.11)$$

The average engineering strain rate of the test can be calculated by subtracting the velocities of each end of the test specimen and dividing by its length, l_s (equation 2.12). [2, 13, 14]

$$\dot{\epsilon}(t) = \frac{v_1(t) - v_2(t)}{l_s} \quad (2.12)$$

To know the values of these velocities, one can resort to equation 2.7, and by analyzing the evolution of the wave on the incident and transmission bars, represented in figure 2.10. [2, 13, 14]

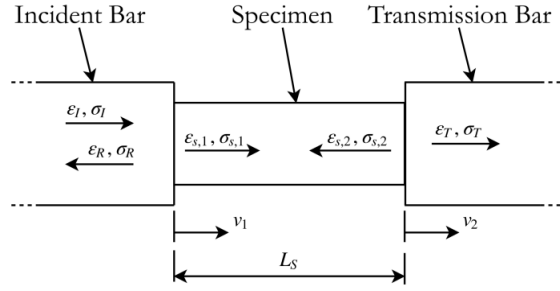


Figure 2.10: Strain and stress behavior representation in apparatus' bars and specimen (compression setup example) [17]

In this way, the velocity of the tip of the specimen connected to the incident bar will be dependent on the deformations corresponding to the initial pulse and the reflected pulse, as in equation 2.13, instead of the velocity of the tip connected to the transmission bar which will only be dependent on the value of the elastic deformation of this same bar, as in equation 2.14. [2, 13, 14]

$$v_1(t) = C_b[-\epsilon_i(t) + \epsilon_r(t)] \quad (2.13)$$

$$v_2(t) = -C_b\epsilon_t(t) \quad (2.14)$$

Applying these expressions to equation 2.12, one can directly relate the strain rate to the strains registered throughout the test.

$$\dot{\epsilon}(t) = \frac{C_b}{l_s}[-\epsilon_i(t) + \epsilon_r(t) + \epsilon_t(t)] \quad (2.15)$$

However, for the specimen to attain the state of stress equilibrium, there must be a certain degree of internal wave reflection. This period (equation 2.16) is directly related to the initial specimen length, to the elastic stress wave velocity in the specimen material C_s , and to the quantity of wave reflection inside the specimen, n (expected to be three for the majority of metallic materials). The specimen may only be deemed to be in the stress equilibrium condition and to be deforming evenly after this period. [2, 13]

$$T = n \frac{l_s}{C_s} \quad (2.16)$$

As a result, it is possible to consider a certain equilibrium of strains, as in equation 2.17, altering equation 2.15, and ending with equation 2.18. [2, 13, 14]

$$\epsilon_i(t) + \epsilon_r(t) = \epsilon_t(t) \quad (2.17)$$

$$\dot{\epsilon}(t) = \frac{-2C_b\epsilon_r(t)}{l_s} \quad (2.18)$$

Regarding strain measurement of a SHPB specimen, it is widely considered that using reflected pulse is a less reliable method to determine the test sample's history of strain, especially for hard brittle materials. By utilizing a high-speed camera to track the full-field deformation distribution of the test sample subjected to impact, Digital Image Correlation (DIC) stands out as a reliable measuring approach. The technique is demonstrated to provide more accuracy in getting the strain history than traditional wave analysis, which, in some circumstances, may make the stress wave measurement superfluous. [13]

2.3 Projectile speed sensing methods

When analyzing high strain rate tests, it is important to understand the strain of the tested sample, and the amount of energy involved in the test.

As discussed before, an SHPB test is initiated by a striker bar, that acts like a projectile, hitting the incident bar to which the specimen to be analyzed is attached. Throughout this test, there are significant energy losses, through sound, local temperature increases or friction. However, it would be beneficial to compare correctly performed tests with faulty ones, in order to be able to identify flaws, like poor motion transfer and, in turn, excessive energy loss, so they could be corrected as early as possible.

In order to analyze the movement of the bars of the apparatus in question, to check its proper functioning, it would be interesting to analyze the speed of at least one of the bars, in particular, the speed of the striker bar, since it initiates the movement. By analyzing the speed of this bar, it would be possible to compare the energy that would be passed on to the others, and see if the energy transfer was effective.

To analyze the striker bar's speed, a DIC system could be used, but this system's video analysis requires extensive post-processing. This implies that prior to this analysis, which is conducted after the realization of the tests, it is not possible to determine whether the tests were conducted correctly. If these tests conclude that the apparatus had some deficiency, and that it needed maintenance, the same tests would have to be repeated. This causes a lot of time to be wasted. A speed sensor could be installed in the SHPB to increase test execution efficiency and data collection. It would work in conjunction with the DIC system already in place to analyze the specimen rather than the apparatus' bars. Thus, it would be possible to analyze the speeds of the bars when the tests are executed, realizing if any parameter must be adjusted, or if the bar needs some maintenance, resulting in stopping the tests earlier, and, thus, saving time.

Also, if one would want to analyze the speed of several bars at the same time, it would be necessary to install several analysis points, which would not be feasible with DIC systems due to the cost, and rigorous setup, particularly of the high-speed cameras. In the other hand, a speed sensor system should be cheaper and easier to use.

Therefore, after understanding the usefulness of a linear velocity sensor system, some methods for projectile sensing systems were researched to serve as base for the developed system in this work.

2.3.1 Electromagnetic field effect

The ability to move a body with its own magnetic field by altering the electric current in a nearby circuit makes it possible to create mechanisms that regulate body speed. This happens because changing electric current in a circuit creates its own magnetic field, which then interacts with surrounding magnetic fields. This interaction can be represented by inducing motion in a body, or by inducing a new electric current in a nearby circuit. [18, 19]

Based on this effect, it was possible to create everything from motors to solenoids. Using a solenoid-like principle, Liu et al., 2014, could replace the pressurized air chambers of the SHPB with an electromagnetic gun. In this way, it was enough to use electric current to accelerate the striker bar to start the test. [18, 19]

The system consisted of several components, including launch coils, diode rectification circuits, capacitor groups, a charge and discharge indicator, and a trigger switch with an electromagnetic pulse launcher. When the power supply was activated, the rectifier circuit transformed the alternating current into direct current, fully charging the capacitor group in a small time interval. When using the trigger, the capacitor group quickly discharges, generating a current pulse in the launch coils. This pulsed magnetic force caused the striker bar to be released quickly. [18, 19]

Using this principle, it is also possible to describe motion rather than cause it. To measure the SHPB striker bar speed with this method, it would be necessary to install coils around the bar to be analyzed, which would need to have magnetic properties or create them, in order to alter the electromagnetic flux between each coil. This method would depend on the induced electromotive force, which would be generated when the bar cuts the magnetic field of the coils, which would be positioned at equal distances. This induced electromotive force would result from a change in the magnetic flux in each coil target. It would then generate an electrical pulse, which would act as an open and close trigger on the coil target. The maximum value of this electrical pulse would be used to measure a time interval between the maximum values of the electrical pulse of two consecutive coils. With this interval, the velocity would be calculated since the distance between the coils is known. [18]

This method setup is represented in figure 2.11.

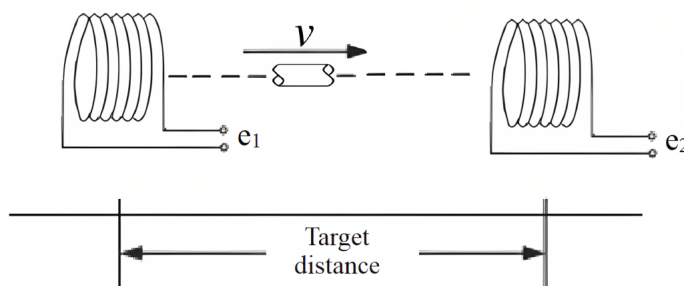


Figure 2.11: Electromagnetic coils setup representation (adapted from [18])

While the method's working principle seems simple to use, it has not been very developed and would also end up taking up a lot of space. This way, this method is not ideal to be applied to this work.

2.3.2 Doppler effect

The Doppler effect, named for physicist Christian Doppler, is a phenomenon where a wave's frequency changes as a result of the source or observer moving in relation to each other. Waves of sound, light, and radio can all be seen to exhibit this effect. [20–22]

The apparent frequency difference between a wave's source and receiver is caused by the source and receiver's relative motion. It is necessary to assume that the source's wave frequency is constant in order to comprehend the Doppler effect, and that the wave's wavelength won't change. The receiver will register the same frequency produced by the source if both the source and the receiver, are stationary. This is due to the fact that the source and receiver both produce and register an equal number of waves per second. [20–22]

The receiver should register a higher wave frequency if either the source or the receiver, or both, move toward the other. This happens because the receiver will take in more waves per second. In contrast, the receiver should register fewer waves per second and, thus, a lower frequency, if the source and the receiver are moving apart. The wave frequency produced by the source must have remained the same in both scenarios, so it is possible to make this direct comparison. [20–22]

A representation of the effect can be analyzed in figure 2.12, where it is possible to see three balls: a white one, a blue one and a red one. The white ball represents the wave source, while the blue and red ones represent receiver positions. Considering that the source has a movement to the left side of the figure, the waves emitted will be closer to each other on the left side of the source and further away from each other on the right side, as it is seen in the figure. This way, the blue ball will detect a higher frequency than the red one, as it was previously explained.

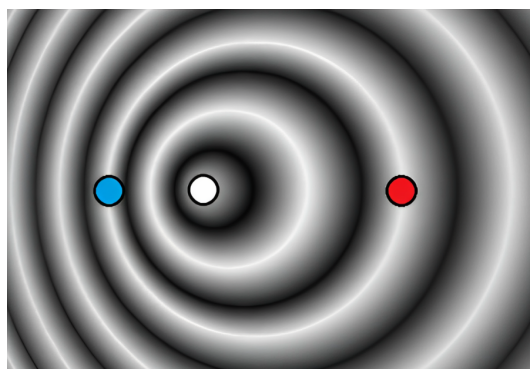


Figure 2.12: Doppler effect representation (adapted from [20])

This effect can be used in various ways, depending on the application and wave to be used. One common use of this effect is to calculate the speed of a moving object. For that, it is possible to use the Doppler radar method and the Photon Doppler Velocimetry (PDV) method.

Both methods use the principle of wave reflection when it encounters an object. Then, the reflected wave is registered by the method's devices. This new wave will have a different frequency than the emitted one, and, through comparison processes, the object speed can be calculated. [23,24]

The main difference between these methods is the used wave of each one. While Doppler radar uses microwaves for the reflection process, PDV uses a laser beam. [23,24]

Doppler radar systems emit microwaves from an emitter antenna, that reflects on objects, returning to a receiver antenna. The system analyzes the properties of reflected energy to determine the distance between the radar unit and reflecting objects. One of the main measurement methods employed in doppler radar sensors is continuous wave radar. Pulsing radar emits short pulses of electromagnetic energy that return after being reflected, which phase has changed compared to the emitted signal. With this phase shift, time of flight is calculated, and, thus, the distance between the radar and the object [25, 26]. On the other hand, continuous wave radar utilizes a continuous wave or signal with a known and stable frequency. It measures the Doppler shift, that is the frequency increase or decrease, of the reflected signal, enabling the determination of the speed at which the objects are moving, making it effective for measuring relative velocities. Doppler radar systems typically operate in the microwave spectrum, ranging from 300 MHz to 40 GHz. This choice of frequency range allows for achieving desired resolution while avoiding interference from environmental factors. [23,27,28]

A representation of a continuous wave radar can be seen in figure 2.13.

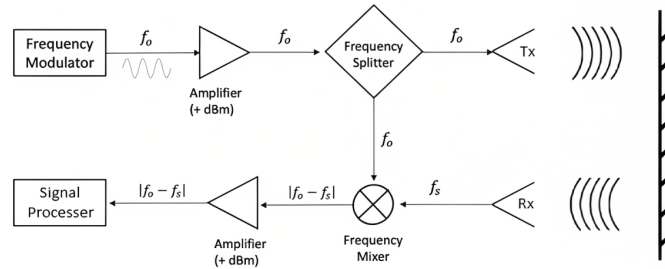


Figure 2.13: Doppler continuous wave radar representation (adapted from [29])

In recent years, techniques using Photon Doppler Velocimetry (PDV) have made measurements that are comparable to those made using strain gauges on a SHPB. This is accomplished by reflecting a laser beam off the surface to make a non-contact measurement of the bar's surface velocity. A combination of the emitted and reflected light, that results in an interference pattern that changes over time, can be used to determine a bar's velocity history. These techniques are well suited to collect data from a striker bar during a direct impact experiment where there are velocity components due to bulk motion and elastic deformation because they measure local velocity rather than strain. Also, this method can be used with an angle reflection of the beam on the side surface of a bar, or it can be applied to the end of a bar, where the reflection is linear. [24,30,31]

A PDV setup needs a high-power laser source, a detector, so it can detect the reflected light, and data acquisition system, apart from the structural components to mount the laser in the desired

angle. A PDV system can be very accurate, but it can take up a lot of space and be very expensive too. [24, 30–32]

A PDV setup representation can be seen in figure 2.14.

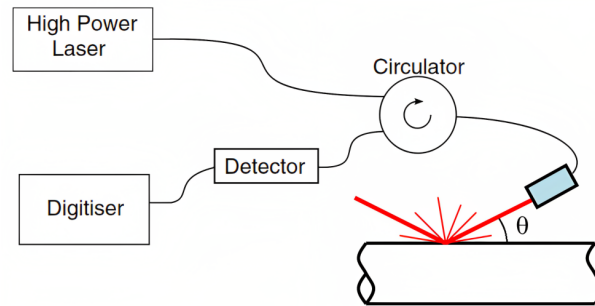


Figure 2.14: PDV setup representation (adapted from [33] and [31])

While Doppler radar works better for longer distances, as it works with an electromagnetic wave with a longer wavelength, PDV works better for shorter and more accurate measurements. PDV systems have already been developed for SHPB, however, these solutions are customized for each case [30], and there is no commercial solution.

2.3.3 Optical sensor

When speaking of optical sensors, one can speak of various types of sensors and uses, such as sensors for temperature, chemical reactions, pressure, or vibrations. However, in this work, the sensors analyzed are infrared optical sensors, namely phototransistors. [34–36]

Phototransistors can have various configurations depending on the intended application. Thus, they can detect objects by reflecting or blocking a beam of light emitted by an emitting diode, which normally accompanies the phototransistor. These configurations can be characterized in three ways: through-beam, retro-reflective, and diffuse-reflection. [34–36]

The through-beam and retro-reflective configurations consist of a beam of light, emitted by the emitting diode, being interrupted by an object. Thus, the phototransistor stops receiving this beam of light, changing the value of the output signal. These methods cannot calculate the proximity of the detected object, only its presence. This method is very easy to apply but implies having an area available where the object can pass, in order to interrupt the beam of light. [34–36]

The only difference between the through-beam and retro-reflective configuration is the position of the emitter relative to the phototransistor. In through-beam, the emitter is in front of the phototransistor, emitting the beam directly towards it. In retro-reflective, the emitter and phototransistor are next to each other, and the emitter emits a beam onto a reflecting surface, thus being detected by the phototransistor. [34–36]

These configurations' setups are represented in figures 2.15 and 2.16.

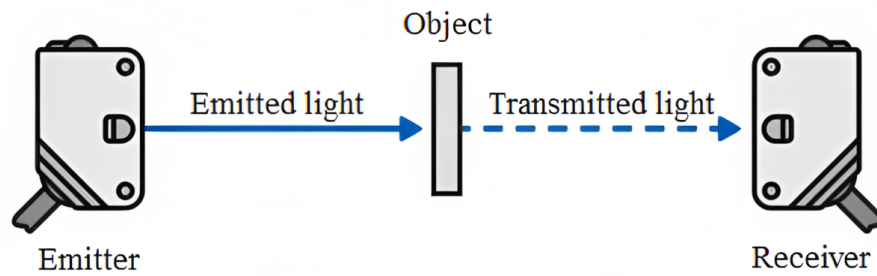


Figure 2.15: Through-beam setup representation (adapted from [37])

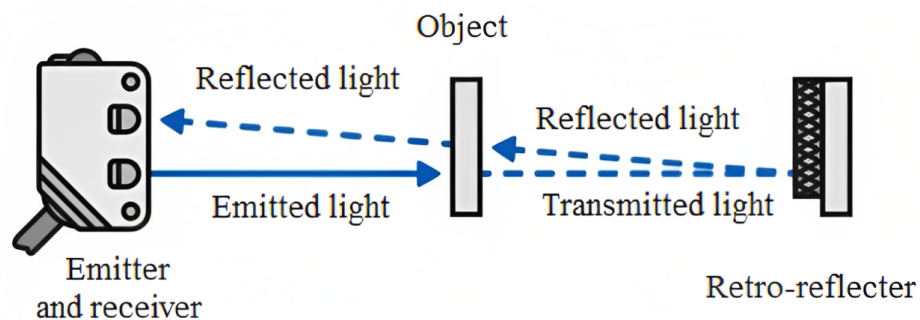


Figure 2.16: Retro-reflective setup representation (adapted from [37])

The diffuse-reflection setting consists of the emitted light beam being reflected by the object itself. Thus, the phototransistor detects a change in the intensity of the beam, because of the change in the distance at which the beam was reflected. With this, it is possible to calculate the proximity of the object. Also, this method does not need any specific area where it can be applied, making systems more compact. [34–36]

This configuration's setup is represented in figure 2.17.

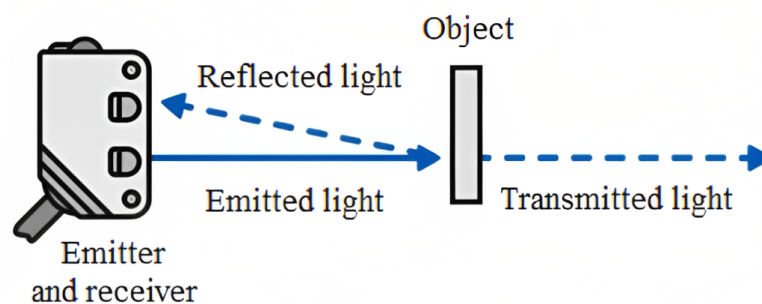


Figure 2.17: Diffuse-reflection setup representation (adapted from [37])

Whichever setup is used, the method for calculating linear velocity will be similar. When the system detects an object, the value of the digital signal changes, with a transition, positive or negative, depending on the sensor used. These transitions can be used to mark the start of a passing

object and start timing it. Knowing the length of the object, it is possible to calculate its speed, simply by dividing its length by the time it took to pass. Another variant of this method would be, for example, to simply count the number of objects detected, and, combining that number with an arbitrary time interval, calculate the frequency of recorded passages.

This way, it is possible to understand that these sensors are very versatile and easy to use, however, they may have technological limitations that compromise the functioning of the system if used for very high speeds and very short transitions.

2.3.3.1 Commercial solution

Even though the energy dissipation purpose might be useful, and interesting to study, not a lot dedicated systems for measuring the bar's speed from the apparatus in question, the SHPB, have been developed as of yet. In the research for this work, only one dedicated speed sensing system for the analyzed apparatus was found. This system was developed by REL Inc, and is represented in figure 2.18. [38]

Not much information is available about how this device works, but some are shared, such as its dimensions, its ability to be powered by a battery, and its speed accuracy of 100 nanoseconds up to 3,000 ft/s, or 914.4 m/s. [38]

Through analysis of the shared images, the system should work with infrared sensing technology, detecting transitions with a reference object placed on the bar itself. The setup of the optical infrared sensor suggests that it may work with a through-beam configuration, where a ring, positioned on a bar of the apparatus, should interrupt the beam and serve as length pattern.

With the shared information, it was possible to understand that it would serve as good reference to the work to be developed, helping to decide which methods should be used.

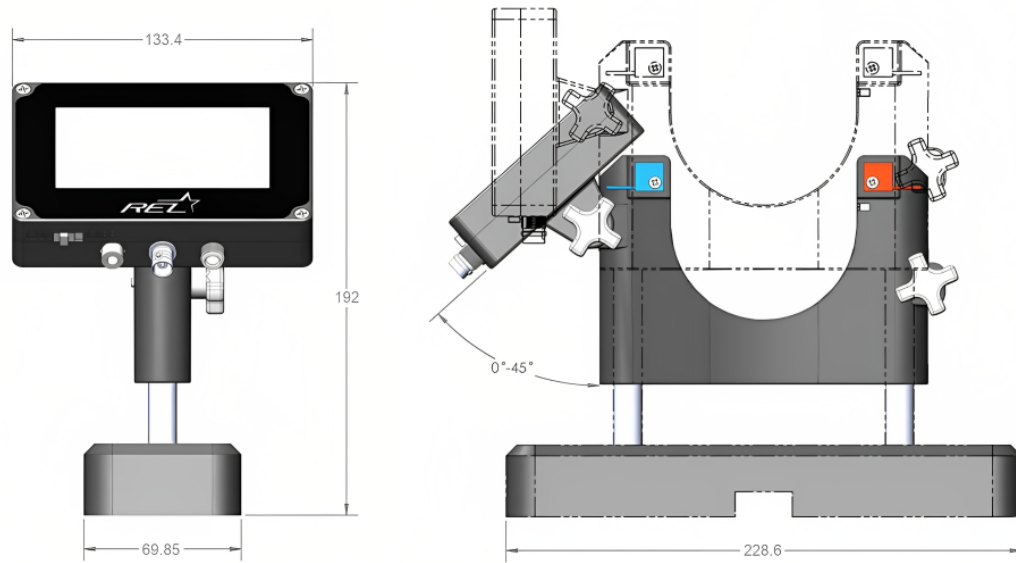


Figure 2.18: REL Inc mounting on apparatus speed sensor (adapted from [38])

2.3.4 Proposed solution

After analyzing some methods of speed sensors, it was decided to use the method of optical infrared sensors with diffuse reflection configuration, as it is an easily replicable, inexpensive method that would take up little space.

When using this type of sensor, it would be necessary to slightly modify one of the bars of the apparatus in question in order to get an effective reading from the sensor. Thus, the device's ability to relate reflected radiation with object proximity would be used, positioning it in front of a pattern of white and black bands, fixed on the bar to be analyzed. Since the sensor works with reflected radiation, monochromatic color analysis is hypothesized, where the white color will reflect more radiation than black color bands. This way, it should be possible to time transitions between the bands, and, by defining a fixed band size, to calculate an average speed of movement of one of the bars of the apparatus, such as the striker bar.

This system method aims to be a more compact and a potentially cheaper solution than the found dedicated one. However, it is not expected to be able to provide accuracy or a range of speeds as high as that of the commercialized system, given the limitation of the used sensor.

A representation of this proposed solution can be seen in figure 2.19.

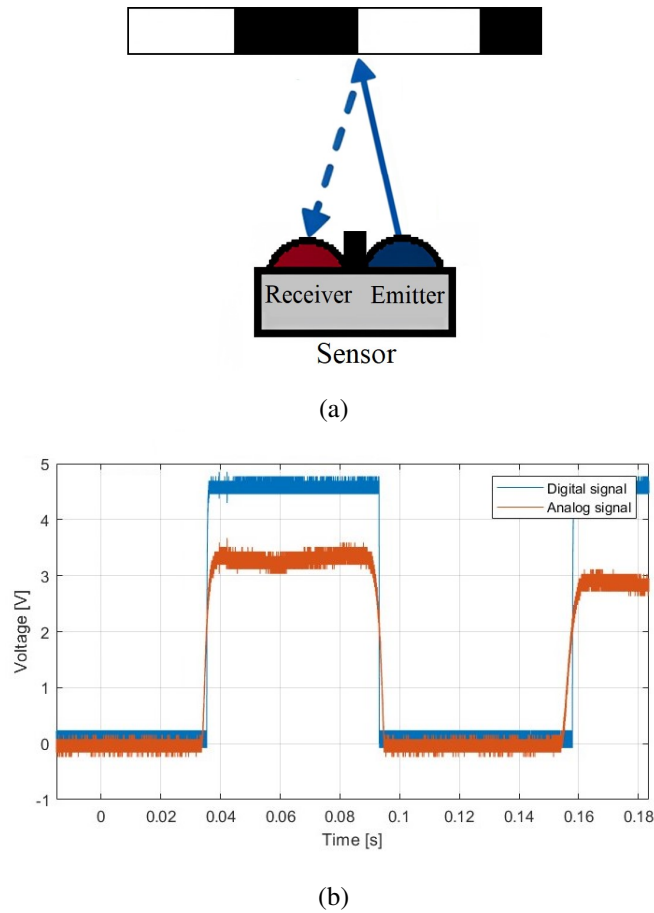


Figure 2.19: Proposed solution: a) method representation; b) expected signals examples.

2.3.5 Applied Digital Image Correlation

Even though an alternative solution to the DIC, to analyze the speed of the bars, was sought, a variant of this image motion analysis method was used in this work to validate the developed system.

As mentioned in chapter 2.2.2, DIC is used, for example, to analyze the strain of the specimen that was used in a SHPB test, as it allows information to be calculated or extracted quickly, easily and with high accuracy.

This method is a non-interferometric and non-contact method that consists of matching sets of pixels between consecutive images, like frames of a video, in order to extract intended information [3, 13, 39, 40]. For this, various algorithms were developed, such as iterative spatial cross-correlation algorithm, using the Newton-Raphson method, and the coarse-fine search and peak-finding algorithms, to name a few [39]. One challenge that all algorithms must overcome is the need to match each pixel between images. This is not viable, since the grayscale value of a pixel can be associated with multiple other pixels in a second image. Thus, it is normal practice to analyze a small area around the pixel of interest, called a subset, providing the system with more data. This is represented in figure 2.20. However, the first subset can still be matched to another

one in a second image. This ambiguity of matching pixel areas, between images, is called aperture problem. To resolve this, the aperture of an analyzed area should be increased, revealing more end points of shapes. This way, it is possible to match pixel areas more efficiently and determine unique motion vectors between images. [39,40]

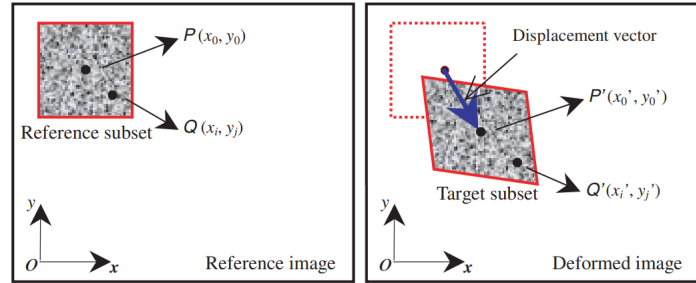


Figure 2.20: Schematic representation of a reference subset and a target subset following deformation (adapted from [39])

However, the correspondence problem between pixels becomes more complex if the test is not restricted to rigid motion and has components like specimen deformation. This is worse for structure without significant texture, where it is impossible to obtain data between image boundaries. To resolve this problem, the structure surface should not have a preferred orientation, since repeating textures may cause issues with registration. These conditions imply that the use of random textures obtains better results. For this, speckle patterns are applied to surfaces before the tests since they adhere to the specimen and accompany the deformation of the specimen. Therefore, no information correlation is lost. [39,40]

In figure 2.21 is shown an example of a speckle pattern applied on a SHPB specimen, and the DIC results of the same test.

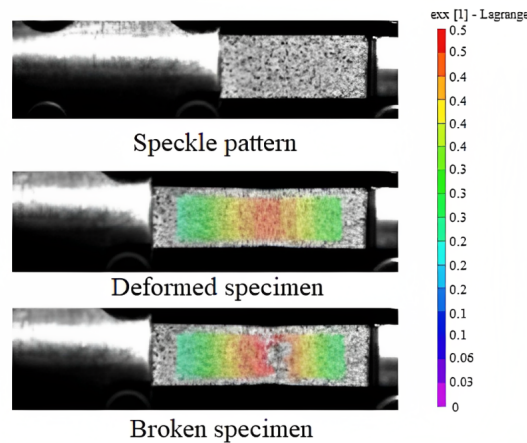


Figure 2.21: Speckle applied on specimen and DIC results example (adapted from [13])

A DIC setup consists of using a camera and a light source. The latter is needed to illuminate the surface to be recorded to increase the contrast detected by the camera sensor. For flat surfaces, the use of one camera is sufficient, but for curved or irregular surfaces, the use of two, or more,

cameras is necessary to be able to make the three-dimensional relationship of the images. A computer is needed so it possible to analyze the recorded video, with dedicated DIC software. A DIC setup is shown in figure 2.22. [3, 13, 40]

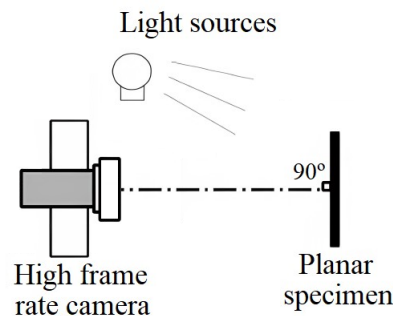


Figure 2.22: DIC planar specimen setup example (adapted from [13])

As mentioned, in order to validate the developed system, a variation of DIC was used. Since the system will be based on transitions between black and white colors, it was crucial for the pattern's bands to have as few imperfections as possible. Therefore, it wasn't possible to use a speckle pattern, which is commonly used in DIC analysis. Instead, it was opted for a simpler approach. This involved tracking points defined within the used DIC program, GOM Correlate, which were associated with the regions where color transitions occurred. This allowed to determine whether the digitally defined points accurately matched the reality depicted in the recorded videos, particularly in terms of tracking these transitions. By utilizing the tools provided by the selected program, it was possible to gather information about the speed of the defined points and, consequently, of the bar where the band pattern was applied. GOM Correlate was chosen because of its user-friendliness interface and its powerful tools that suited the needs for this task. An example of the use of this program and obtained results is shown in figure 2.23.

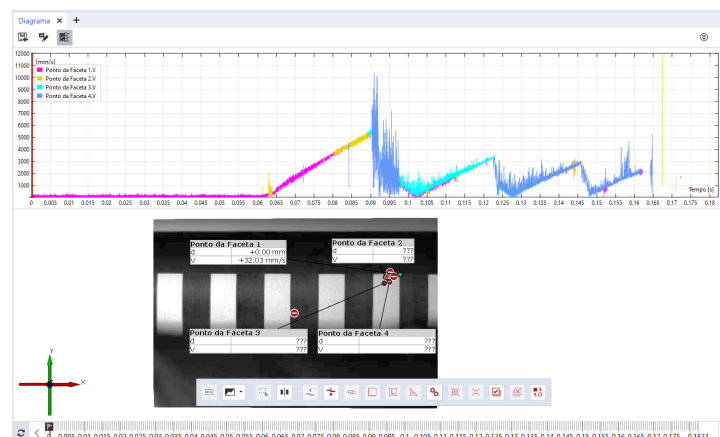


Figure 2.23: DIC example: point tracking and respective speed calculation (GOM Correlate software)

Chapter 3

System Development

3.1 System Requirements

After knowing the general working method of the apparatus in which the developed system will be applied to, and after choosing the base method for this latter system, it was time to engineer the idealized system.

As previously mentioned, it was decided to elaborate a system that used a diffuse-reflection optical sensor and a black and white band pattern. The sensor would detect transitions between bands and count the time between these. Since the distance between bands would be fixed, and equal between each other, speed could be easily calculated.

For this, it is very important to define a major parameter that is the time resolution that the system must deliver. However, this parameter depends on the working speeds at which the apparatus operates. This range should correspond to speeds between 0 and 10 m/s.

It should be noted that there is a usable area of 430 mm available for positioning the pattern on the striker bar. This distance corresponds to the stroke the striker bar travels when shot. However, the pattern will have to be placed at a distance equal to this distance from the nearest support, to not interfere with the movement of the bar, nor to be damaged. With this, the sensor module should be mounted at least 430mm away from the nearest support, ideally coinciding with the beginning of the pattern, in order to make the most of this usable length.

Also, it was defined that the system needed to be able to distinguish speed differences in the order of 0.1 m/s. This way, it would be possible to distinguish if the striker speed is 9.9 m/s, or 10 m/s. However, as any object speed, this will be dependent on the travelled distance, in this case, it is the transitions' distance, and on the time measured too, as is possible to understand from equation 3.1.

$$v = \frac{\Delta x}{\Delta t} \quad (3.1)$$

This way, a time resolution for the system needed to be calculated, by determining the time difference between the time intervals of two considered speed, with a difference of 0.1 m/s between each other, and with an arbitrated transition length. This calculus results in equation 3.2,

and various time resolutions are shown in table 3.1, at a maximum speed of 10 m/s, with some arbitrated transitions lengths.

$$\Delta t_{resol} = \frac{\Delta x}{v_{max}} \cdot \left[\frac{1}{1 - \frac{0.1}{v_{max}}} - 1 \right] \quad (3.2)$$

Table 3.1: Corresponding resolution times for different transition lengths at 10 m/s

$\Delta t_{resol} [s]$		$\Delta x [mm]$				
		5	10	20	40	80
$v_{max} [m/s]$	10	5.05	10.10	20.21	40.40	80.81

With this information, it is possible to understand the minimum amount of time that should be counted, thus the time resolution, is 5.05 μs . This will be the resolution that the system needs to deliver for a pattern of 5mm, at 10 m/s, so that a 0.1 m/s difference is noticeable. However, the pattern size can be altered if it is necessary. Below is a table summarizing the established requirements and limitations for the development of the sensor system (3.2).

Table 3.2: Table of requirements and limitations

Requirements	
Speed range [m/s]	0 - 15
Speed resolution [m/s]	0.1
Maximum stroke [mm]	430
Time resolution for 5 mm transitions [μs]	5.05
Time resolution for 40 mm transitions [μs]	40.4

3.2 Optic sensor

For the optic sensor part, it was chosen between two types of infrared tracker modules that were available. These two modules had incorporated the same sensor, the TCRT5000 proximity sensor, as shown in figure 3.1.a). Its schematic top view is shown in figure 3.1.b).

This sensor works with a reflective principle: it emits infrared radiation and then processes the amount of the same radiation that is received, as represented in figure 3.1.c), resulting on a certain amount of voltage. [41]

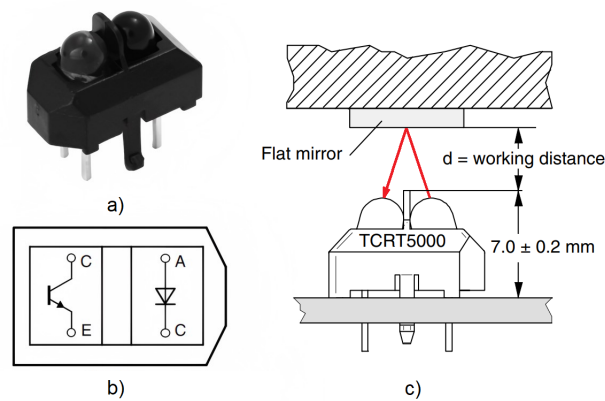


Figure 3.1: a) TCRT5000 sensor; b) Schematic top view of TCRT5000 sensor; c) Test conditions (adapted from [41]).

The main differences between these modules were the comparator incorporated in each of them and the number of output pins available. One of these modules had a comparator with a fixed threshold, with just one output pin for the digital signal originated from the comparator. The second module had a potentiometer associated with the comparator, and two output pins were available, one for the digital signal after the comparator process and one for an analog signal before the comparator process. This analog output pin offered the possibility to understand precisely how the sensor behaved, separately from the rest of the module. This feature and the capability to alter the threshold of the comparator were decisive for the choice between the modules, favoring the second module. This sensor module, as shown in figure 3.2, is very small, very cheap and it brings good capabilities for the system replicability.



Figure 3.2: TCRT5000 module with analog pin [42]

In figure 3.3, it is possible to see how the module works. The circuit includes some resistors, a potentiometer that is connected to the LM393 comparator, the TCRT5000 sensor, as well as the connection pins. The LM393 comparator receives the voltage values from the potentiometer and the sensor, and, using the potentiometer value as threshold for the sensor voltage values, it processes these values, resulting in a digital signal. The sensor values without being processed by

the comparator also result in an analog signal, which can be analyzed by another pin.

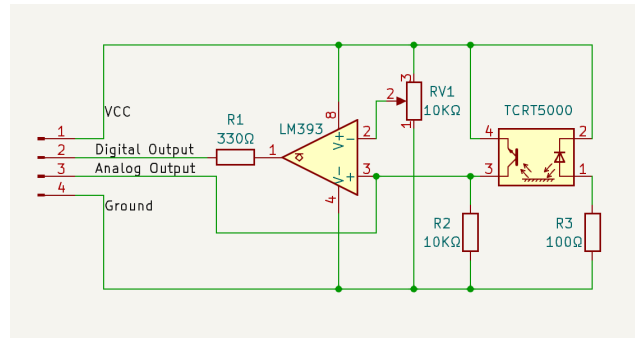


Figure 3.3: Sensor module's simplified circuit representation

3.3 Timing hardware

When choosing processing hardware, some aspects need to be accounted for such as needed features, computing power and system cost.

3.3.1 Method and components selection

The system will need to timestamp the transitions of the pattern, as previously established. For this, solutions based on dedicated Integrated Circuits and microcontroller hardware modules were considered, such as a time-to-digital converter (TDC) and a microcontroller input capture unit (ICU), respectively. These solutions would need to be conjugated with some acquisition and processing hardware, like a Data Acquisition computer board (DAQ) or a microcontroller.

Starting with the timing components, the TDC is a stopwatch integrated circuit used to measure time between a single event and subsequent events, also known as time-of-flight (TOF), and it returns this value in the form of a digital signal, that can be used by other components depending on the application. These devices have an internal time base that is used to measure time with accuracy in the order of picoseconds, which makes the TDC a good option for precise timing applications such as flow meter, where zero and low flow measurements require such high accuracy. [43]

Input Capture Units are hardware module, commonly implemented in microcontrollers, that mark the timestamp when an input trigger event is detected. These timestamps can, then, be used in diverse applications. These units tend to work with a time base that is related to the internal microcontroller clock, which is then associated to time resolution of the unit. This way, time resolution in the order of nanoseconds is obtained with ease. [44]

About acquisition hardware, DAQ boards are devices that connect to a computer and interface with sensors to acquire analog and digital data. They convert the sensor data into a digital format that the computer can process, but they can also provide analog outputs. These boards are used in data acquisition systems for monitoring, analyzing, and controlling processes. [45] They are

powerful from a computational point of view and can have features such as high-speed counter channels that provide high frequencies and scan rates. A DAQ board can run at high clock speeds and can provide different time resolutions, such as 20.83 ns. [45,46]

A microcontroller is an integrated circuit that is capable to perform a many different tasks, since it is equipped with various features. It is designed to run a single program repeatedly and is used for specific tasks in embedded applications. Microcontrollers detect and react to external stimuli through GPIO pins and can be programmed for various functions. The components of a microcontroller include a CPU, RAM, ROM, internal oscillator, I/O ports, and optional peripheral control integrated circuits. These components work together to execute programs and control connected systems. However, while microcontrollers can be versatile and useful for small tasks, or situations where space is scarce, these components have limited computing power. [47,48]

Firstly, a DAQ board was considered since it was precise enough for the application. However, since it would be expensive, and would be too powerful for the application in case, it was considered to use a microcontroller instead, with a TDC. This combination would be a good option since the timing between transitions would be done by the TDC and the data acquisition and processing would be executed by the microcontroller. An Arduino Mega2560 board, with an ATmega2560 microcontroller was used since it was available for this project, and it was easier to use than other microcontrollers since it had already connections made to microcontroller itself. While studying this board, it was seen that it had connections made to some ICU pins of the microcontroller. The time resolution of these units was 62.5 ns, since it was the time that takes between ticks of the microcontroller clock. This value corresponds to the inverse of the clock speed of 16 MHz. This way, it was decided to change from the TDC to the ICU since the latter offered sufficient time resolution and was already embedded in the microcontroller. By doing it like this, no additional wiring or space would be required, and since no components were purchased, there would be no need to wait for their delivery.

3.3.2 Microcontroller Input Capture Unit

As already mentioned, the used microcontrolled was the ATmega2560, assembled on an Arduino Mega2560 board. Arduino is an open-source electronics platform that consists of hardware and software components. The hardware part includes various microcontroller boards that can read inputs from sensors and control outputs. The software part is developed in a C++ based language, which can be written with a text editor (IDE). A typical Arduino board consists of a microcontroller, digital and analog input/output pins, and a serial or USB interface for communication with a host computer. It utilizes Atmel AVR or ARM microcontrollers with built-in input/output support. [49,50]

The Arduino Mega 2560, as shown in figure 3.4 is a microcontroller board that offers a wide range of features including 54 digital input/output pins, 14 of which can be used as PWM outputs, and 16 analog inputs. It has a 16 MHz crystal oscillator which will set the pace at which the system will work. The board can be powered with a USB connection, which is used to connect the board to a computer. [44,51,52]

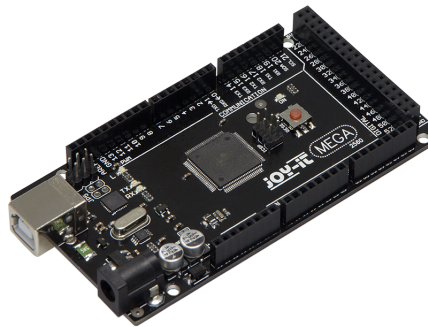


Figure 3.4: Arduino Mega2560 from JOY-iT [52]

This board's microcontroller has embedded timers that are associated with the input capture units, which register external events and provide a timestamp indicating the time of occurrence. These units trigger a time-stamping routine when a change in logic level occurs and meets the edge detector settings, writing the 16-bit value of the counter in the Input Capture Register (ICRn). This value represents the number of ticks that the microcontroller clock has made in the clock cycle. This way, it is possible to measure time with it, since a tick corresponds to 62,5 ns, which is the inverse value of the clock frequency, 16 MHz in the case of the ATmega2560, as previously mentioned [44]. This way, this time value will be the time resolution of the system.

With this functionality, it is possible to measure real time intervals, with nearly zero delay. Using this unit, the need to execute an interrupt routine of a normal timer is avoided since that time value is captured by the hardware in a separate register. This way it is possible to eliminate the delay of entering and exiting the interrupt routine, that is around 2 μ s [53]. Even though this setup should work in the order of microseconds, in this work, most of the data will be large enough to be on the order of milliseconds, but not large enough to reach the order of seconds. However, the near zero delay will give more certainty about the collected data and keep the system more precise and reliable. [44, 53]

The ATmega2560 has 4 pins with this functionality, but only two of them are directly connected to Arduino Mega2560 board pins, pins 48 and 49, corresponding to timers 5 and 4, respectively, of the microcontroller. The table 3.3 shows a summary of the pins with input capture capability. [44, 51, 52]

Table 3.3: Input Capture Pins in Arduino Mega2560 [44, 51, 52]

Pin Number	Pin Name	Board Pin Name
9	PE7 (CLKO/ICP3/INT7)	
35	PL0 (ICP4)	Digital pin 49
36	PL1 (ICP5)	Digital pin 48
47	PD4 (ICP1)	

After studying the microcontroller board and the ICU, the sensor module was connected to the Arduino. The power input pin, VCC, was connected to the board's 5V voltage output pin, and the ground pin was connected to a pin of the same type, in order to facilitate wiring. The module's digital signal pin was connected to pin 49 on the board, for the respective ICU to analyze the signal. The module's analog signal pin was not connected to the board as this information was not used by the developed firmware later, but was used to check the digital signal's transitions and adjust its threshold, with an oscilloscope. This circuit resulted in the representation in figure 3.5.

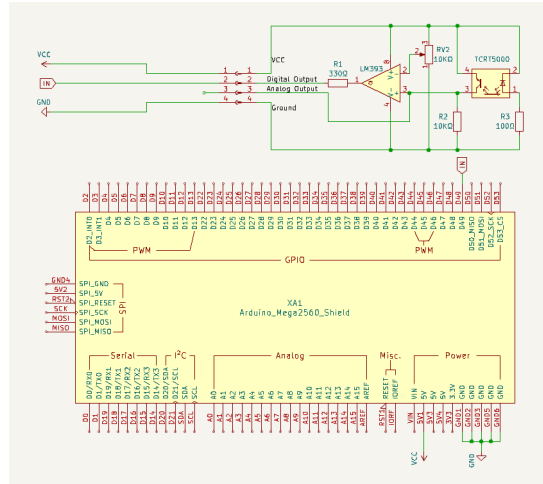


Figure 3.5: Connection between Arduino board and sensor module circuit representation

3.4 Firmware Development

To be capable to use this functionality, it is necessary to develop firmware for the Arduino board. For this, a publicly available firmware code was used as a basis for the code written for an input capture unit used on the chosen microcontroller, the ATmega2560.

The example firmware was developed to be used in Arduino Uno, which is a board that uses the ATmega328 microcontroller. It uses an ICU associated to microcontroller timer 1 to register the timestamp of each transition that triggers the unit [54]. The example firmware triggered the used ICU with rising edge transitions in pairs, as it interprets only one first transition and one second transition in order to calculate an average velocity. This way, a third transition is treated as a new first transition and continues in the same manner for subsequent transitions. This working approach is not optimal for the intended purpose of this project since it discards many transitions that could be used to describe the test in much greater detail. [53]

Since the microcontroller board to be used did not have established connections for its own Timer 1, it was necessary to choose a different timer with pre-established connections. Therefore, pin 49, which is one of the two ICU related available pins on the board, was chosen. This pin is associated with Timer 4, which required all references related to Timer 1 to be modified to refer to Timer 4.

Next, it was necessary to alter the firmware's working method so it would consider consecutive positive transitions, instead of counting them in pairs. This change aimed to increase the number of transitions analyzed, resulting in a smoother and more detailed description of the test. To correct this, the flag that indicates which is the first transition has only been set to true for a first transition, instead of being set every two transitions. With this, it was necessary to define whether the flag that indicates whether the UCI should be started is only set to true when a transition is detected. Otherwise, after the first transition was detected, the firmware would continue to process phantom transitions because it could not determine if they had stopped or not. These changes ensured that all positive transitions were consecutively considered, and the time between transitions was consistently calculated.

In the figure 3.6, it is possible to see how the example firmware worked with the transitions on the planned work pattern, represented by the red line, and how the developed firmware works, represented by the blue line.



Figure 3.6: Work pattern and firmware's behavior to transitions

In figure 3.7, it is shown that there are three ways to interpret existing transitions:

- In red, the way the example firmware interpreted transitions, which it was already realized that brings no advantage;
- In blue, the way the transitions are currently interpreted, which is from positive transition to positive transition;
- In green, a new way of interpreting transitions, and which would bring benefits for collecting more information, which would be to account for each transition, whether positive or negative.

However, this new way of counting every transition would not be correct if one was to analyse an analog signal, because nothing would assure that it would be measuring equivalent points of the signal at each period of the signal. Thus, the blue way is the most correct way because it could be used in an analog signal that would give assurance that one would be counting from period to period, that is, at equivalent points. In addition, this new way, the green one, goes against the operation of the input capture unit that only lets one define one type of transitions, either positive or negative, to trigger the unit.

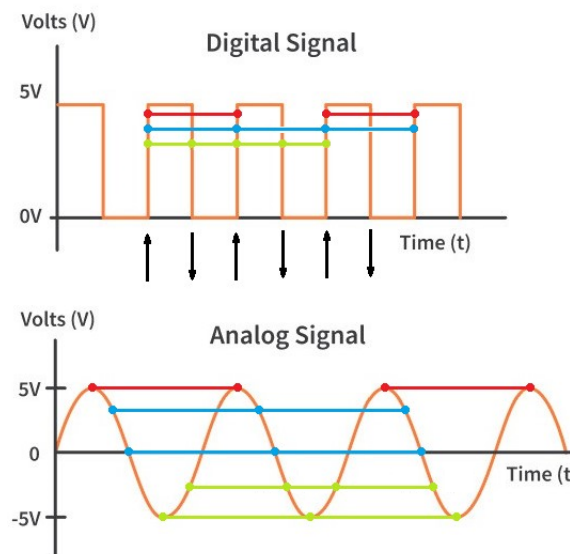


Figure 3.7: Working methods of transitions detection

Figures 3.8 and 3.9 represent how the Arduino firmware works, through flowcharts.

From the moment the Arduino board is connected and the program loaded into its system, it will immediately run, continuing cyclically until something happens to stop it. The program starts by setting the serial communication speed, later defining the input pins and declaring the next detected transition as the first. Then an interrupt preparation pass is executed, which causes the microcontroller's internal variables to be cleaned and prepared for a new pass. Next, the program waits for a transition to happen. When it is detected, it saves the cycle number of the microcontroller's clock. This value will equal the start time value, if it was the first transition. If it was not the first, this value will equal the end time of the current process, and the start time of the next one. Afterwards, the process start value is subtracted from the process end value, thus calculating the value, in system ticks, corresponding to the time between transitions that will be sent by serial port to the computer.

In parallel, the overflow confirmation routine occurs to keep track of the time passed, since after 65536 ticks, the overflow system resets. [53]

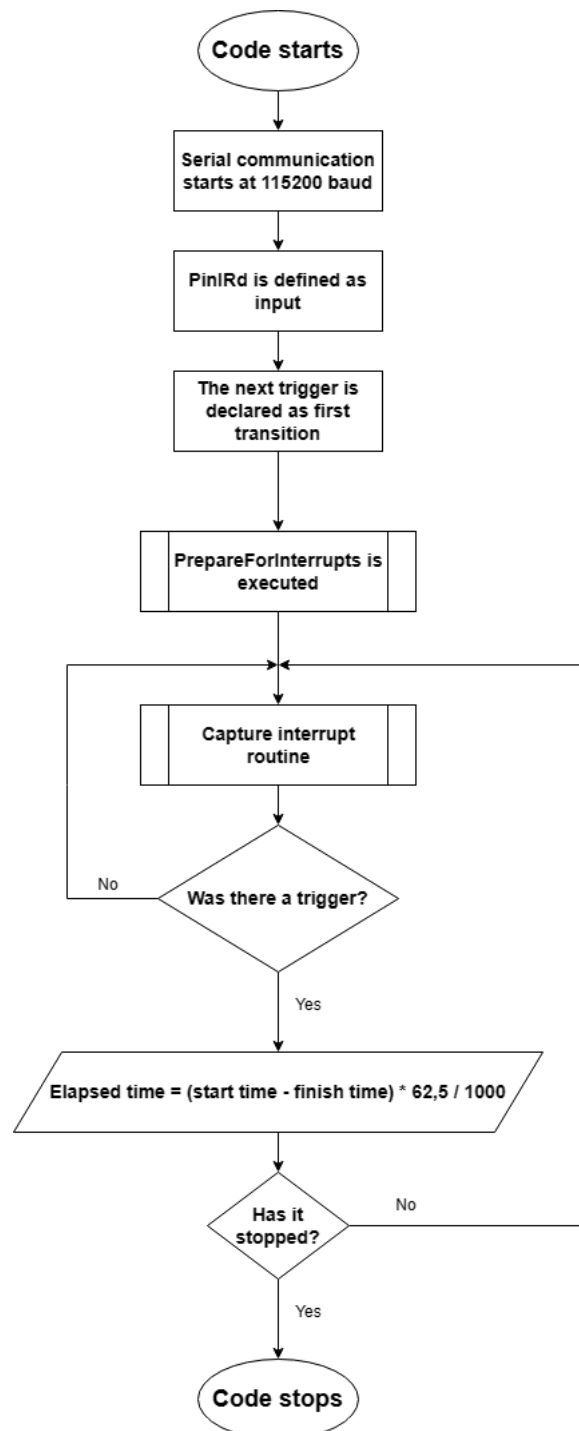


Figure 3.8: Arduino firmware main flowchart

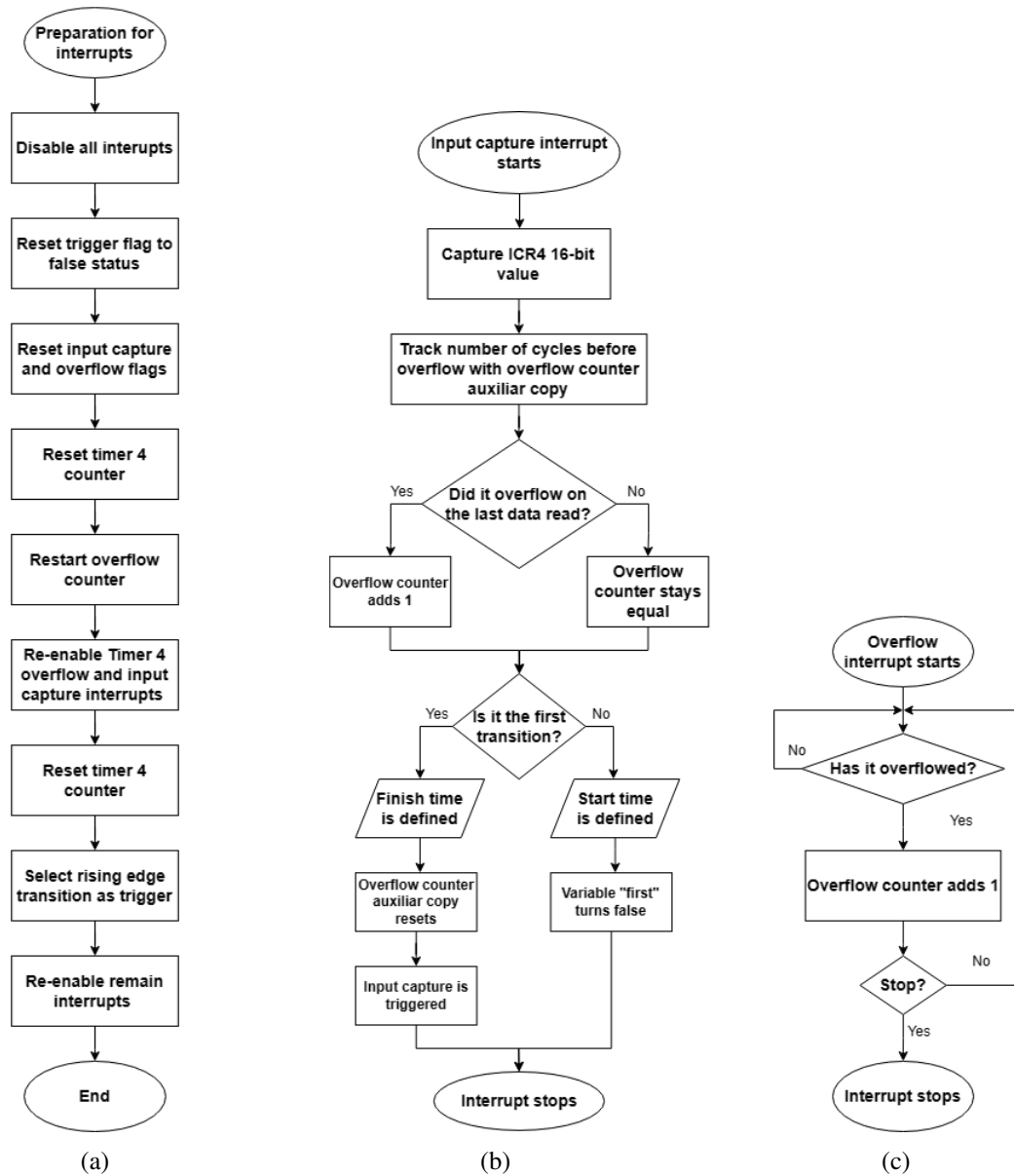


Figure 3.9: Arduino firmware secondary flowcharts: a) Preparation for interrupts routine; b) Input capture interrupt routine; c) Overflow counter routine.

3.5 Board Validation

Once the processing hardware and software components were defined, it was necessary to validate the timing system performance. Since it is intended to measure small values of time, it made sense to know what was the smallest amount of time that could be measured and understand if it would fulfill the requirements for the system's resolution. It also made sense to know if there was a maximum value of time that could be measured, or if the counter would keep going on.

For this purpose, it was used the capability of some pins of the Arduino Mega2560 to create a Pulse Width Modulation (PWM) signal, with a frequency as high as possible to understand what is the smallest time that the system can count.

3.5.1 Pulse Width Modulation

Pulse Width Modulation (PWM) is a method for using a digital signal to control analog devices. Using this method, a digital device, such as a microcontroller, can output a signal that resembles an analog signal, as shown in figure 3.10). The produced PWM signal can be used to operate actuators, motors, lights, and other devices. [55–58]

A digital signal is based on high (1) or low (0) voltage levels. In the context of controlling the speed of an electrical motor with inertia, Pulse Width Modulation (PWM) is a technique that uses a switch to vary the energy supplied to the motor. Closing the switch accelerates the motor, while opening it slows it down. By adjusting the duration of the switch's open and closed states, one can maintain a desired medium speed for the motor. The digital signal keeps permutating between 0 and 1, low and high, in a way that would simulate an intended analog signal. [55–58]

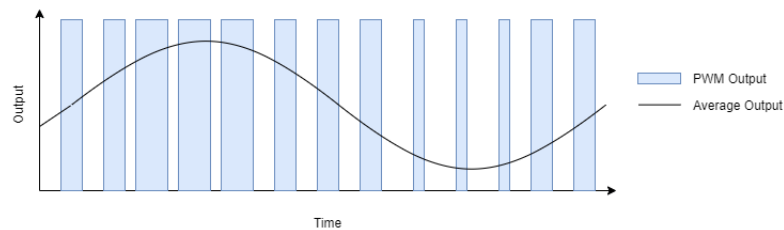


Figure 3.10: PWM digital signal and average output representation [59]

When talking about this method, it is needed to talk about the duty cycle of digital signal, so it will assimilate to the analog one. [55–58]

The duty cycle (D) is the amount of time that the digital signal is at high level (T_{on}), over a bigger period. The rest of the time, the digital signal goes to low level. Overall, the duty cycle is the percentage of a period when the signal is at 1. [55–58]

$$D = \frac{T_{on}}{Period} \cdot 100 \quad (3.3)$$

This impacts the voltage that is supplied to the application. If a PWM signal at 10V of maximum voltage ($V_{\max}$), with a duty cycle of 50%, the equivalent supplied voltage (V_{avg}) would be 5V, as in figure 3.11. [55–58]

$$V_{\text{avg}} = \frac{D}{100} \cdot V_{\max} \quad (3.4)$$

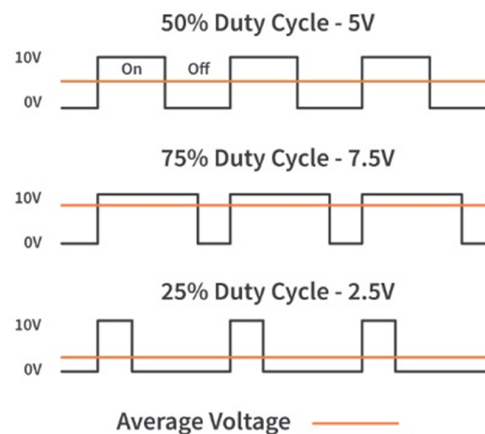


Figure 3.11: Duty cycle examples [55]

When talking about PWM, it is mandatory to talk about the frequency of the signal. It determines a PWM signal's behaviour, much like duty cycle is. It refers to how many times a signal repeats per second. Depending on the application, a certain frequency may be necessary. For instance, a PWM signal's frequency needs to be high enough to produce the desired dimming effect when controlling LEDs. For example, the human eye will see that an LED is blinking with a duty cycle of 20% at 1 Hz, but it will be tricked to believe that the light of the LED is dimming if the frequency is increased to 100Hz or more. [55–58]

A PWM signal can be produced in several ways. To create one, one can use, for example, a comparator circuit or a 555-timer integrated circuit. However, utilizing a microcontroller to create a PWM signal, and using an oscilloscope to observe the signal in operation, is one of the simplest methods. By just writing a few lines of code, it is possible to use microcontroller devices like an Arduino board to create a PWM signal. Timers are used in the PWM circuitry on microcontrollers, however they are internally coupled with the pins. [55–57]

The `analogWrite()` function, from Arduino programming language as an example, accepts values in the range of 0 to 255, depending on the bits that are being used, with 0 being 0% duty cycle and 255 denoting 100% duty cycle. This way is possible to control analogical devices through the microcontroller, or sampling testing if some functionality of the project is working as it should. [55,58]

The Arduino board that will be used has the pins that have PWM compatibility. These pins are presented in the table 3.4.

Table 3.4: PWM pins [44,51,52]

Pin Number	Pin Name	Board Pin Name
1	PG5 (OC0B)	Digital pin 4 (PWM)
5	PE3 (OC3A/AIN1)	Digital pin 5 (PWM)
6	PE4 (OC3B/INT4)	Digital pin 2 (PWM)
7	PE5 (OC3C/INT5)	Digital pin 3 (PWM)
15	PH3 (OC4A)	Digital pin 6 (PWM)
16	PH4 (OC4B)	Digital pin 7 (PWM)
17	PH5 (OC4C)	Digital pin 8 (PWM)
18	PH6 (OC2B)	Digital pin 9 (PWM)
23	PB4 (OC2A/PCINT4)	Digital pin 10 (PWM)
24	PB5 (OC1A/PCINT5)	Digital pin 11 (PWM)
25	PB6 (OC1B/PCINT6)	Digital pin 12 (PWM)
26	PB7 (OC0A/OC1C/PCINT7)	Digital pin 13 (PWM)
38	PL3 (OC5A)	Digital pin 46 (PWM)
39	PL4 (OC5B)	Digital pin 45 (PWM)
40	PL5 (OC5C)	Digital pin 44 (PWM)

3.5.2 PWM testing

To test the limits of the system engineered so far, a PWM pin from the Arduino board was used to create a steady, constant, and known signal.

For this, the function *analogWrite()* with a duty cycle of 50% was added to the firmware code in work. Then, the output pin, for the PWM signal, was directly connected to the input capture pin in use. This way, it was possible to verify how the input capture unit worked.

To have a better understanding of the PWM signal, it was also used an oscilloscope, to which was connected a jumper wire that had a connection to the PWM signal, via a breadboard. This way, it was possible to confirm the results that the input capture pin was delivering.

With this setup set in figure 3.12, it was still necessary to understand how to alter the PWM signal in order to understand the real limitations of input capture unit. For that, the Arduino firmware code needed to be altered in order to alter the clock speed of the microcontroller of some of the pins with the PWM capability.

Before tweaking with the clock speed of the microcontroller pins, it was necessary to understand how those pins were controlled. The PWM pins are associated to respective timers, that are controlled by the board clock. The table 3.5 shows how the associations were established on the board.

Each of these timers are controlled by internal 8-bit variables. Each bit controls a different module on the microcontroller. This variable is called Timer/Counter Control Register (TCCR),

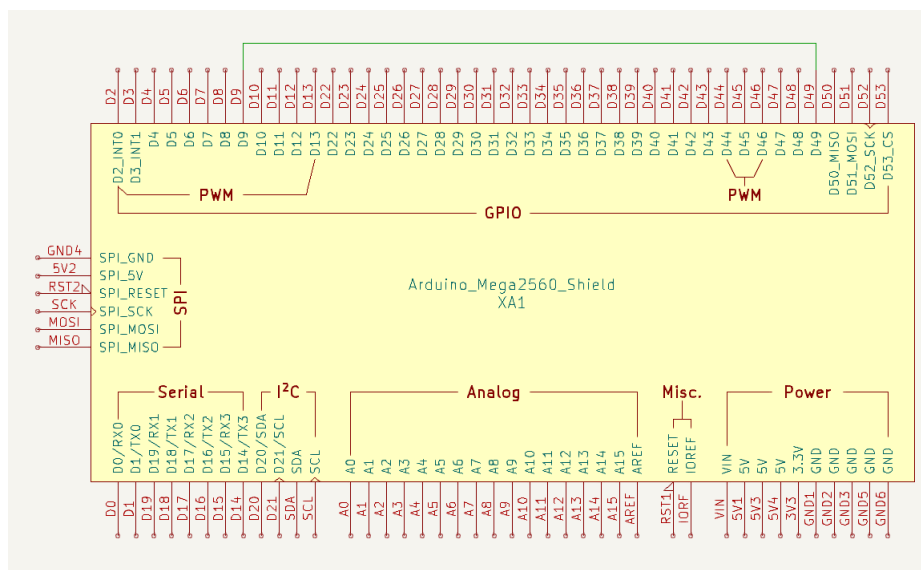


Figure 3.12: PWM testing circuit representation

Table 3.5: Microcontroller’s timers and respective PWM pins [44, 51, 52]

Timer	Board pins associated
0	4, 13
1	11, 12
2	9, 10
3	2, 3, 5
4	6, 7, 8
5	44, 45, 46

and to alter the speed at each timer works, it is necessary to change the bits named Clock Selections (CS). When altering the values of these bits, each combination results in a different behaviour, changing the speed at which the timer works. This can be set to slow down or speed up the timer frequency. [44]

It is important to know that the timer 0 is the responsible to keep the microcontroller working at a normal rate, since it is the principal timer of the chip and controls the basic time behaviour. Time functions like *millis()* or *micros()* rely on this timer, and if its clock speed changes, this functions will be kept functional but at a different timescale [44]. All other timers can be tweaked without interfering with basic functionalities only altering the behavior of the associated pins, in particular, the PWM signal that each pin delivers.

To change the three mentioned bits of the timer register, it is possible to simply assign an integer type variable with a number from 1 to 7. Then, this number will be converted to its binary form and will replace the CS bits [44]. Each number is associated to a certain clock speed, presented in the table 3.6. These combinations and results were verified through testing the firmware code, presented in listing 3.1.

Table 3.6: Prescalers and respective clock speeds

Decimal number	Binary number	Clock Speed for timers 1-5 [Hz]	Clock Speed for timer 0 [Hz]
1	001	31372	62500
2	010	3921	7812
3	011	980	976
4	100	490	244
5	101	245	61
6	110	122	-
7	111	30	-
Default	-	490	976

Listing 3.1: C++ code example to alter timers clockspeed [60]

```

1  int myEraser = 7;    // this is 111 in binary and is used as an eraser
2  TCCR2B &= ~myEraser; // this operation (AND plus NOT), set the three bits in
   TCCR2B to 0
3  int myPrescaler = 3; // this could be a number in [1 , 6]. In this case, 3
   corresponds in binary to 011.
4  TCCR2B |= myPrescaler; // this operation (OR), replaces the last three bits in
   TCCR2B with our new value 011
5
6  const int thisPin = 9;
7  pinMode(thisPin, OUTPUT);
8  analogWrite(thisPin, 255 * .5);

```

Throughout the tests, it was possible to verify the behavior of the system answering to slow and fast signals. With slow and medium speeds, that is, with clock speeds between 30 and 980 Hz, the system worked consistently and delivered expected results, according to some reference values [60], even if they were slightly mismatched. With higher clock speeds, that is, higher than 980 Hz, the results presented in a serial monitor would not be as consistent as in slower speeds, presenting at times values with twice the normal frequency, followed by an almost null frequency. It was understood that both results were overlapping each other. For this, the baud rate of the serial communication was increased, from 9600 baud to 115200 baud and, even, 250000 baud, and the problem was mitigated. This solution made sense, since the number of occurrences per second that the ICU was detecting was constantly increasing. It was, therefore, necessary to increase the speed of serial communication, allowing for a greater rate of occurrences to be correctly recorded.

At the end of these tests, it was clear that the system could handle very fast transitions, that the input capture worked as it should, and it was able to read consistently the values. This way, it was understood that the system could detect a minimum of 16 μ s, since it was not possible to generate a PWM signal with a higher frequency than 62500 Hz, with the used hardware. This didn't assure that the minimum measurable time was the same as the arbitrated resolution, of 5.05 μ s, but it was

useful to understand how small the time values could be counted consistently.

This method was useful to give an idea of how small the input capture unit could measure time, but it doesn't determine the smallest measurable time possible of the system, since the PWM generated signal had limits as maximum speed could not get higher than 62500 Hz. However, the result obtained demonstrated enough potential to keep the development of this work.

Knowing how small the system can time transitions, it left the maximum time to be seen. For this, the PWM method with changing clock speeds wasn't enough, since the frequencies still corresponded to values higher than 1 Hz, which correspond to time values smaller than 1 second.

Thus, the sensor module was connected to the board, and the tests were made manually, which consisted in passing an object in front of the sensor very slowly. The results for these tests granted that the system could detect transitions in the order of seconds as well, without having any clear upper limit.

Also, to simulate the striker bar being shot, an instantaneous button was added to the previous circuit, as it is shown in figure 3.13. This way, the simulation of the bar was made by pressing the button. This would close the circuit and let the input capture pin acquire the PWM signal. As it is shown in figures 3.14 and 3.15, the system was able to keep up with the button presses, and acquire data in a very short time interval.

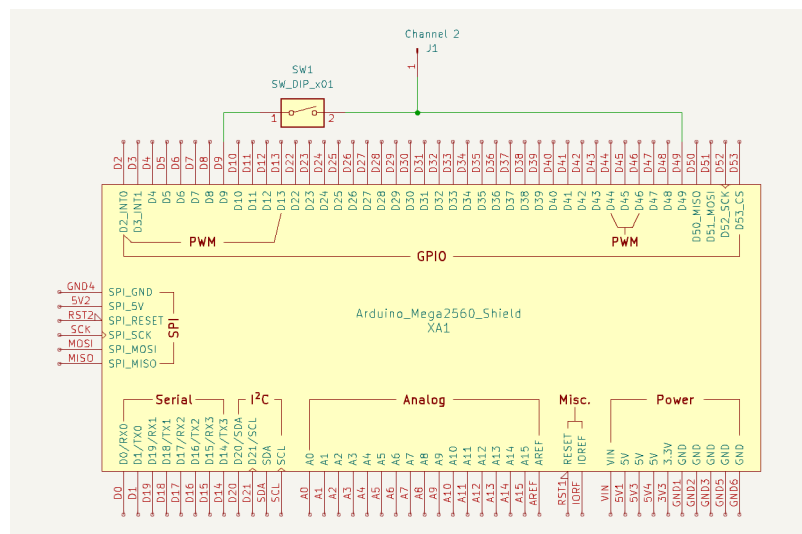


Figure 3.13: PWM testing circuit with button representation

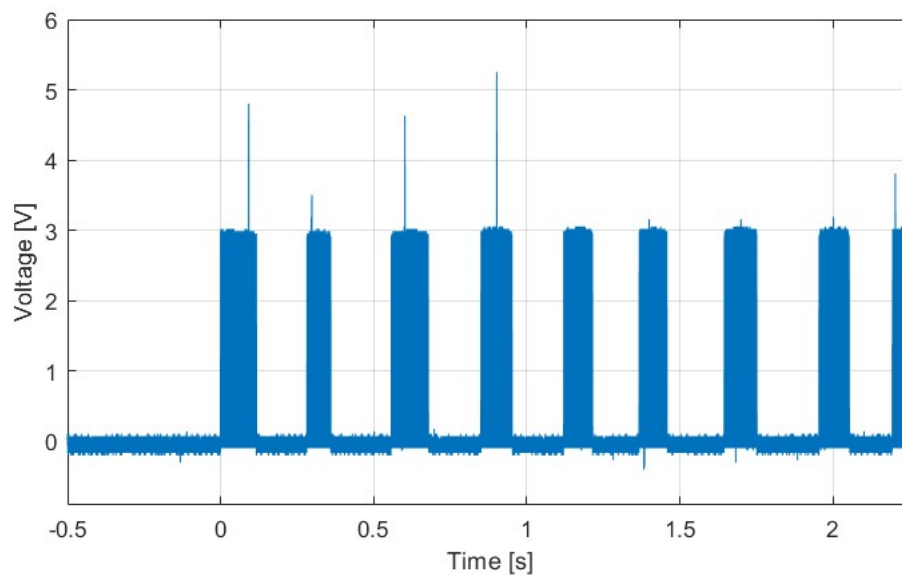


Figure 3.14: PWM test with button oscilloscope result

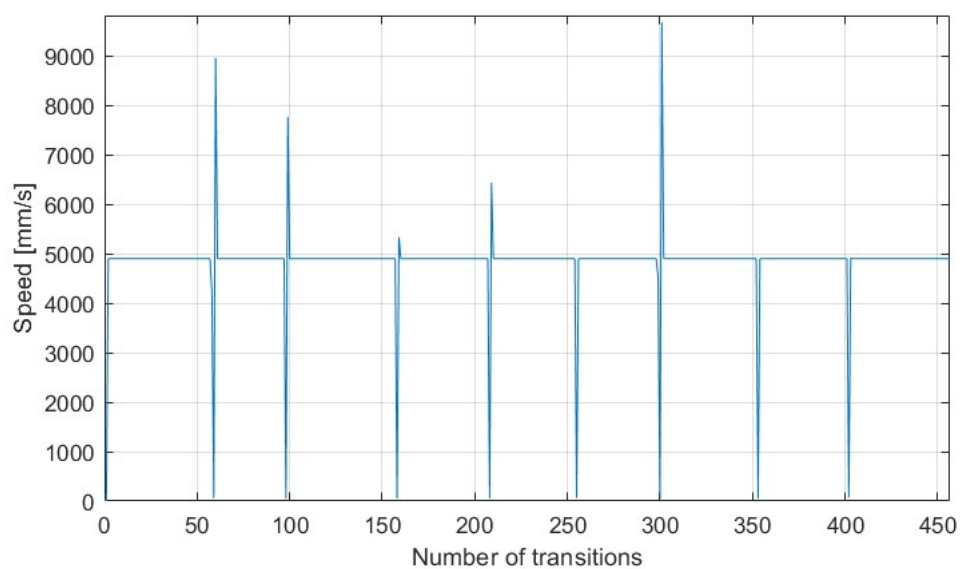


Figure 3.15: PWM test with button ICU result

Also, tests where the PWM signal was acquired by at least 1 second were performed, as shown in figure 3.16, 3.17 and 3.18. These tests indicate that the system is reliable to long periods of data acquisition. These last tests reinforce the reliability of the developed system.

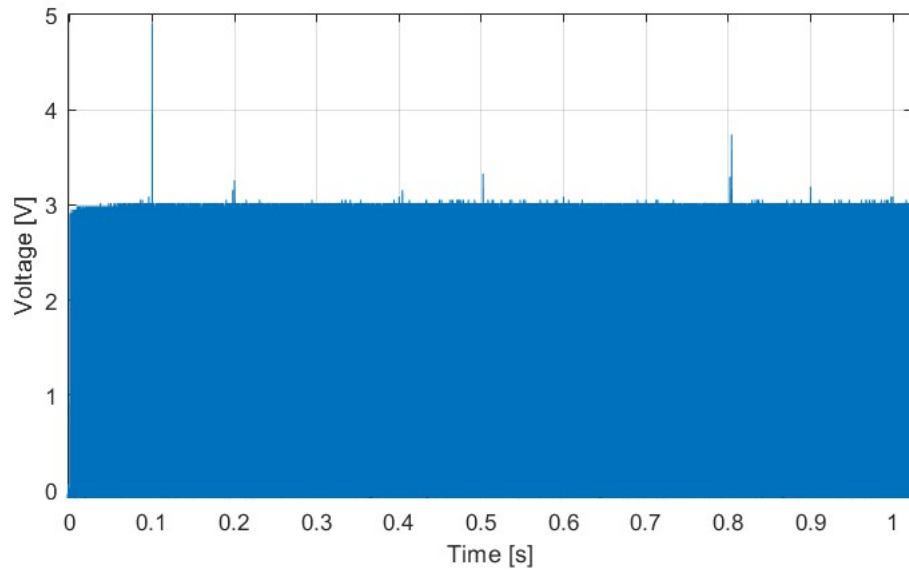


Figure 3.16: Continuous PWM test oscilloscope result

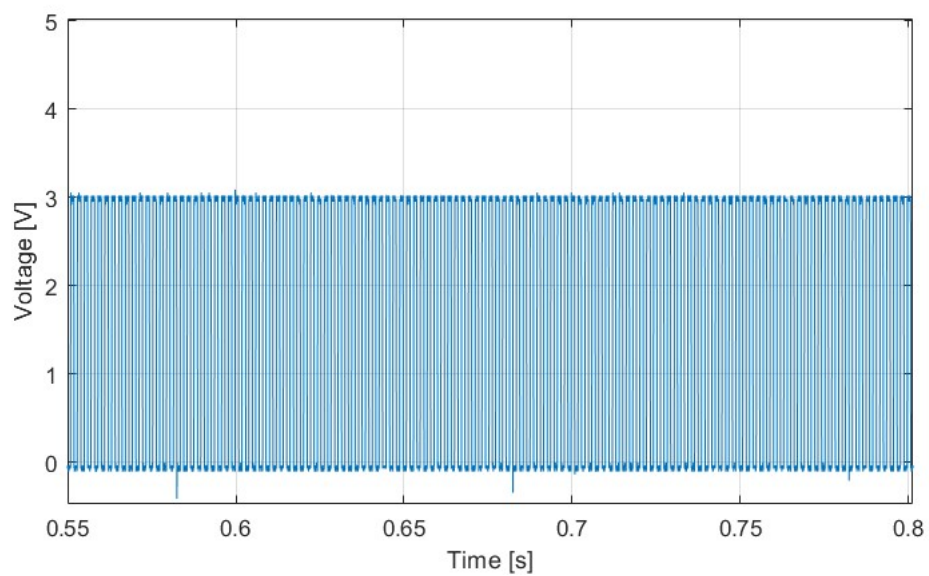


Figure 3.17: Continuous PWM test oscilloscope result detail

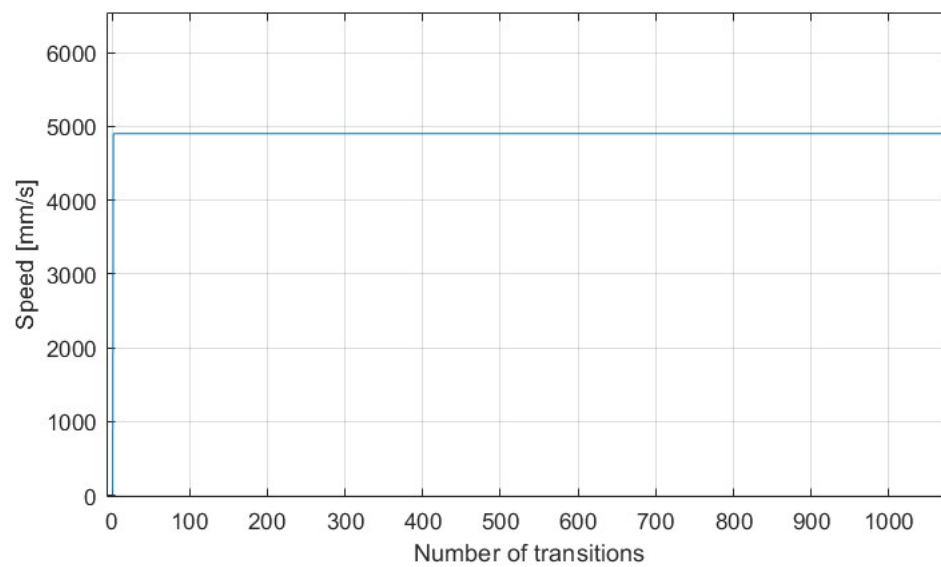


Figure 3.18: Continuous PWM test ICU result

3.5.3 Signal Smoothing

Having already understood the system limitations, testing the system was continued. This time, the purpose was to find if the sensor would interfere in the lower limit of the general system, or not. For that, it was continued testing as in the upper limit test way, by passing the obstacle as fast as possible and always looking at the data collected in the serial monitor. The sensor module was connected to the used oscilloscope, the same way and with same purpose that it was used when it was connected to the PWM pins. With that it would be possible to understand the sensor behaviour, and it would help to understand the origin of any possible problem.

While testing the system manually, it was seen that the communication between the sensor and the board worked as expected, with data at real time scale being delivered. However, the collected data would, sometimes, demonstrate a bit of an erratic behavior by the sensor, since it was detecting transitions by simply moving the sensor, or even if it was quiet.

Allying the collected data in the serial monitor and in the oscilloscope, it was seen that the sensor signal wasn't smooth enough, presenting transitions with many intermediate small signal changes, which were then detected by the input capture unit. For this, some capacitors were added to a breadboard circuit. Two ceramic capacitors of 10 nF each were combined in parallel, resulting in 20 nF, and they were added to the circuit, smoothing the sensor system enough so that intermediate signal transitions were mitigated, and only true transitions were detected by the input capture pin. The final circuit is represented in figure 3.19.

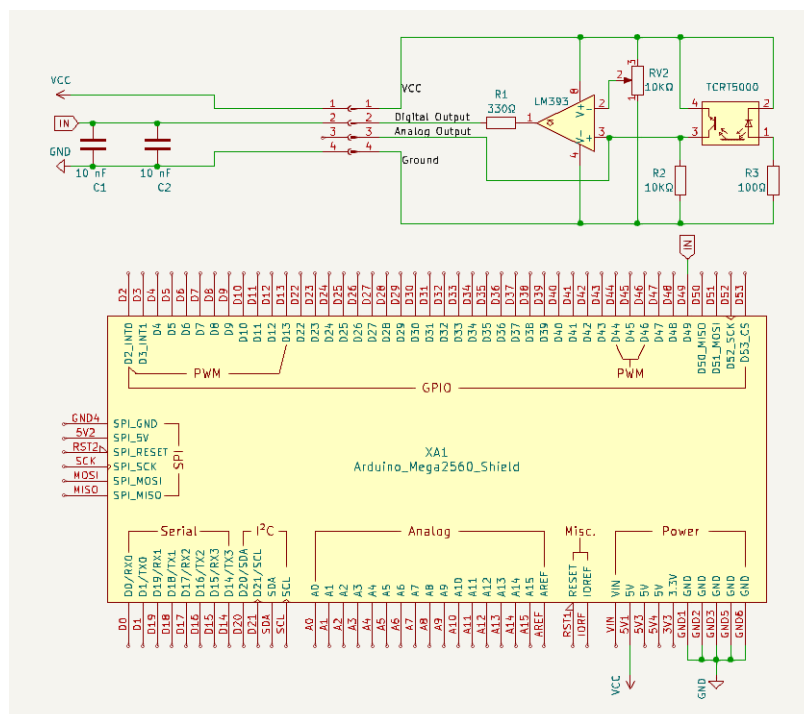


Figure 3.19: Final circuit after signal smoothing

3.6 User interface and support components

3.6.1 User Interface

It was proposed to elaborate an user interface that could be accessed from the computer to which the Arduino board would be connected. This way, the collected information would be easily interpreted through a chart and a received data list. Also, this data should be stored in a dedicated file. To do this, Visual Studio was used to create an application based on .NET Windows Forms, in C#.

It was established that the user interface should be simple to use and to understand the collected data. With this in mind, to establish the communication with sensing system, two selection boxes were added so the user would be able to select the port that the system was connected to, and the baud rate of the communication between the Arduino board and the computer.

To detect used serial ports, a button that fetches those ports was implemented. Additionally, other two buttons were implemented, one to start the communication between the computer and the sensing system, and the other to stop this communication. Also, a label was added to indicate the status of the connection.

The data chart and the data list display the same information, using different graphical representations. These components work at the same time and they register each value at the time that the data is received. This way, the transitions speeds are registered at real time. Also, average, median, maximum and minimum speed values are registered in some textboxes, at real time.

At the end of the test, after stopping the communication with the sensing system, the data can be stored in a .csv file, prompted by the click of a button. This way, it will write the complete data all at once, in a dedicated file.

The user's interface is shown in figure 3.20.

The way that this user's interface should be used starts with the selection of the serial port, through which the communication between the computer and the Arduino board will be made, and the respective baud rate. After, one should press the connect button to start the serial communication. Once the communication is established, the status label updates itself. From this point on, the computer waits for a first value to be received. After receiving the first value, the computer will start storing these values in a dedicated array, but they will be shown in the designated list as in the graph. As the information is received, the time values are used to calculate the corresponding speed, and a few statistics are made to make the data easier to read. When the user wants to stop receiving data and end the serial communication, they must press the button to disconnect the computer from the Arduino. Once communication is stopped, the status label is updated again, and the obtained values can be saved in a dedicated file by pressing the designated button. If there was already a file with the same name, the file name index is incremented. To resume communication with the Arduino board, one should press the connect button again. This will clear the previous data, as the graph and the list, and the acquisition of new values will start from scratch. These routines are represented in figures 3.21 and 3.22. The figure 3.20 shows the user interface with some data collected.

The development of this user interface was fundamental for validation tests analysis, since it was possible to collect data with ease, and, with the data chart representation, it was possible to understand if the test worked as intended, that is, if the data collection was effective or not, and if it made sense priorly to a dedicated analysis.

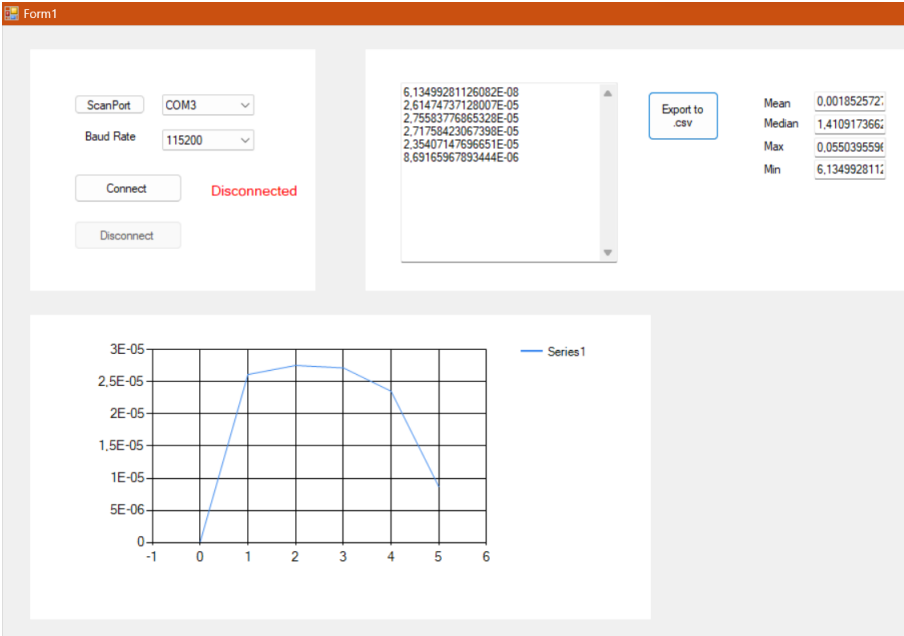


Figure 3.20: Developed interface

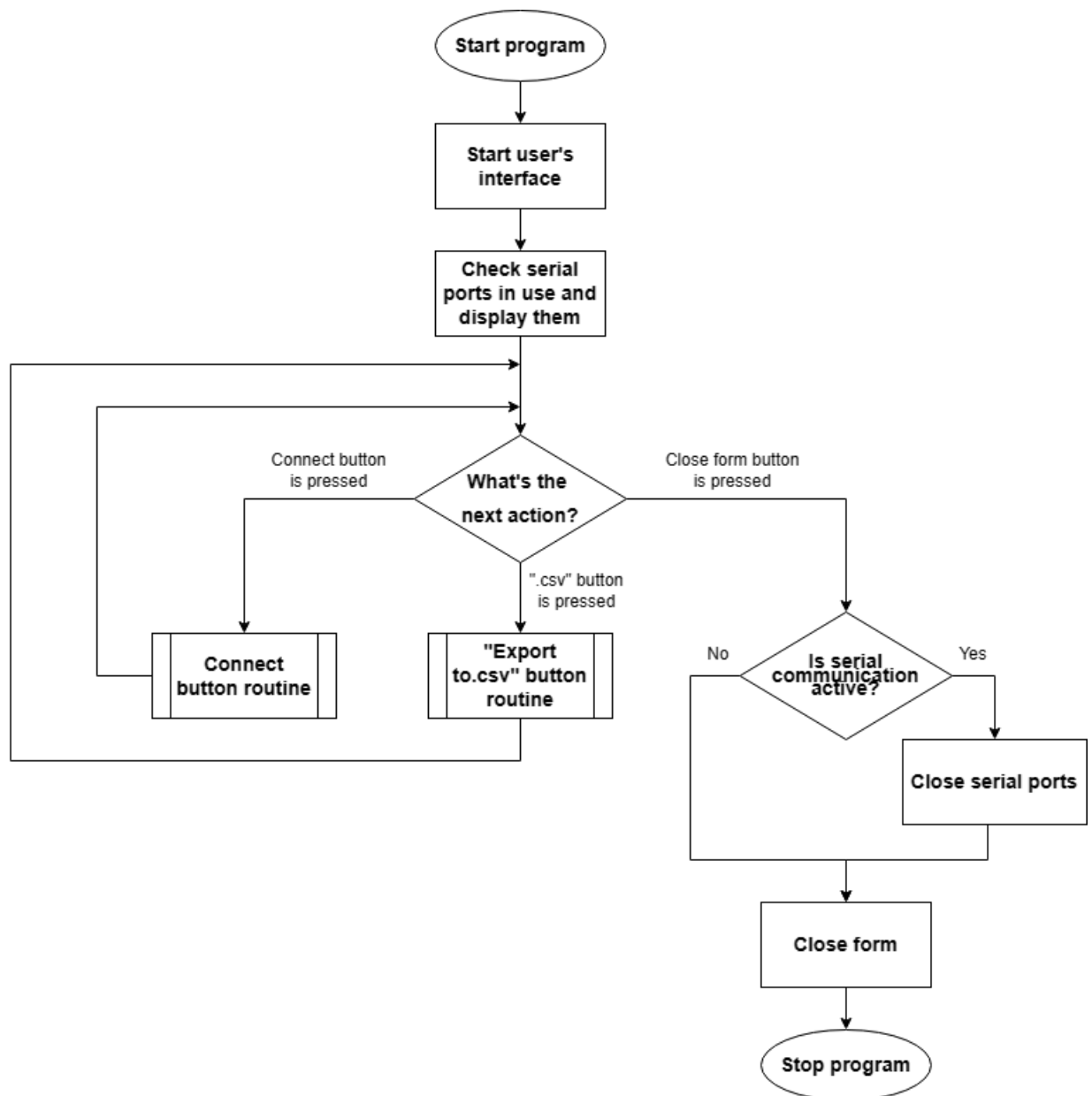


Figure 3.21: User's interface software code main flowchart



Figure 3.22: User's interface software code secondary flowcharts: a) Connection routine; b) Processing data routine; c) Exporting data to a .csv file routine.

3.6.2 Support components

To position the components of the sensing system, namely the sensor module and the Arduino board, it was necessary to develop some small supports to attach to the structure of the SHPB. This structure is made of aluminium Bosch profiles with slots for affixing components using screws and T-nuts, to which the supports were fixed on. The main function of the sensor module support would be to position the module at the top face of the structure profile, under the striker bar of the apparatus. It would be on this bar where the band pattern would be fixed. The support of the Arduino board would be positioned at the bottom or side of the frame profile, to minimize the space required.

A design phase of the supports in question was carried out. It was started by measuring the distance between slots to find out what the distance between holes would be. For the sensor module support, it was decided to make some slots of its own to support the module by the its own board. So, one would just slide the module into the holder. Some tabs were designed to serve as a base, through which the screws would pass and fix to the apparatus' structure. To support the Arduino board, a case for the board, which already existed in libraries of CAD models, was adapted. In this case, two holed tabs were added in order to create a part where the screws would pass through to fix to the apparatus' structure, once again. Once these supports were designed, they were 3D printed.

In figure 3.23, it is shown the CAD models of these supports and their 3D printed counterparts. In figure 3.24, it is shown the assembly of the supports on the aluminium profile of the SHPB structure.

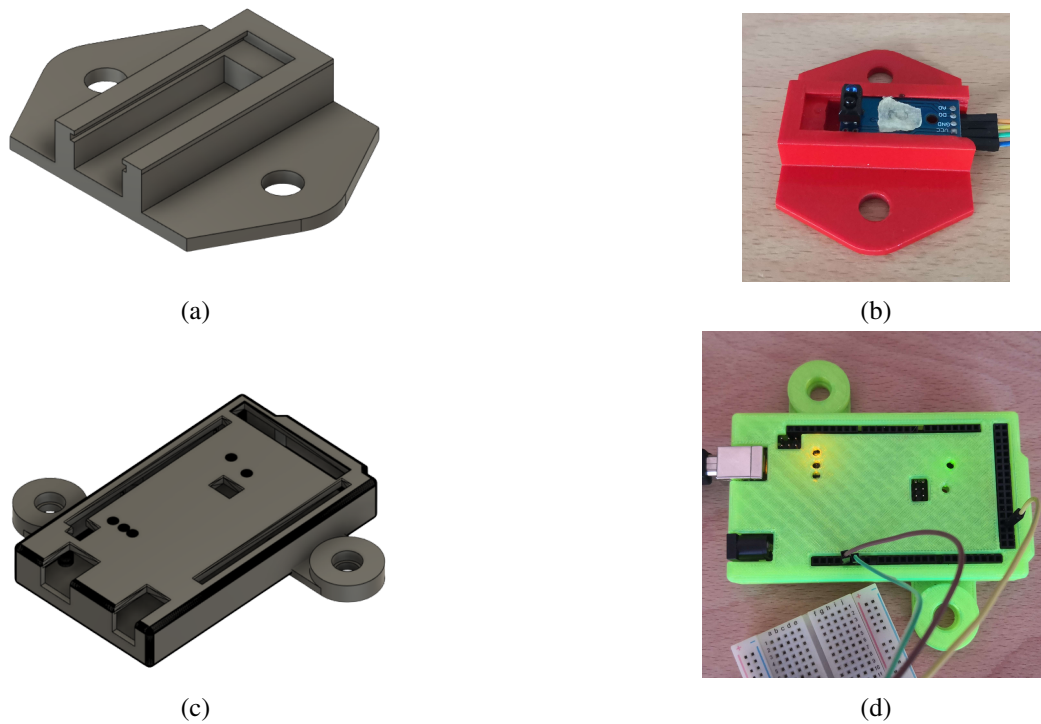


Figure 3.23: Developed support components: a) Sensor support CAD model; b) Printed sensor support; c) Arduino support CAD model; d) Printed Arduino support.

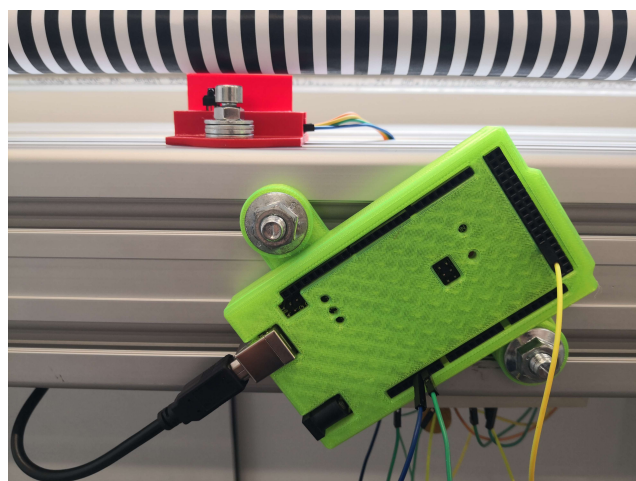


Figure 3.24: Supports for the sensor module and Arduino mounted on the profile of the SHPB structure

Chapter 4

System Validation

With the sensing system developed, it was necessary to validate it through several tests, comparing the results obtained by the system, with the results obtained through analysis of high-speed camera footage.

For this, the sensor module was mounted under the striker bar, with the developed support; a Photron Fastcam Nova S6 high frame rate camera was mounted on a tripod next to the same bar, as well as a GS Vitec MultiLED QT, with a MultiLED G8 controller, to increase the contrast of the image captured by the camera. A Rigol MSO5000 oscilloscope was, later, connected to the sensor module, to analyze the signals that it produced.

A pattern of black and white bands was printed on an A3 sheet of paper, which was cut into a strip since it was fixed to the striker bar, with strips of black electrical tape of the same length. It was ensured that the paper strip was taped in such a way that it did not slip, and that there was no gap between it and the bar, in order to reduce surface height changes that could influence the detection of transitions between black and white bands, and in turn compromise the timing between them. The paper strip was cut wide enough to cover most of the surface of the bar, so that the patterned bands were detected by the sensor, placed under the bar, and by the high-speed camera, which was on the side of the bar, as previously mentioned. Also, two sizes of patterns were used, 40mm and 5mm, starting with the largest and ending with the smallest.

The pattern strip taped to the striker bar and sensor support are shown in figure 4.1, and the high speed camera and high power LED setup is show in 4.2.

All tests were conducted without test specimens, since the maximum speed of the striker bar, that is, its speed immediately before impact with the incident bar, is independent of the existence of a specimen connected to the incident and transmissions bar.

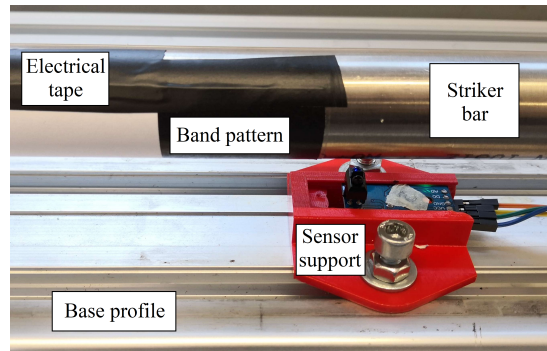


Figure 4.1: Pattern and sensor setup on SHPB striker bar

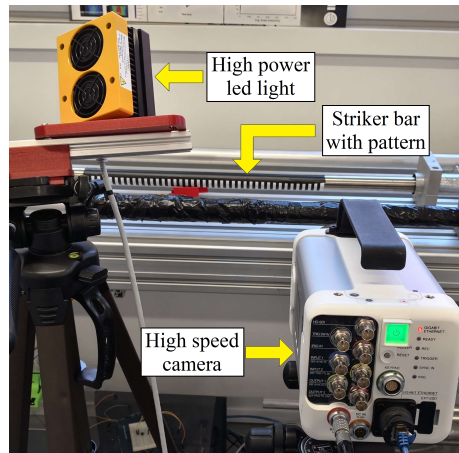


Figure 4.2: Test setup with high speed camera and high power LED light

4.1 Validation results with 40 mm bands

To start validating the developed system, a first set of tests was carried out, in which the setup used was devoid of a high-speed camera, high-power LED or oscilloscope. For these tests, a 40 mm band pattern was used, and the bar was shot with a pressure of 2.5 bar. This test was carried out in order to see if the developed firmware for the system worked as intended, and to detect possible errors, of which none was detected, since coherent data was obtained, as it is shown in figure 4.3.

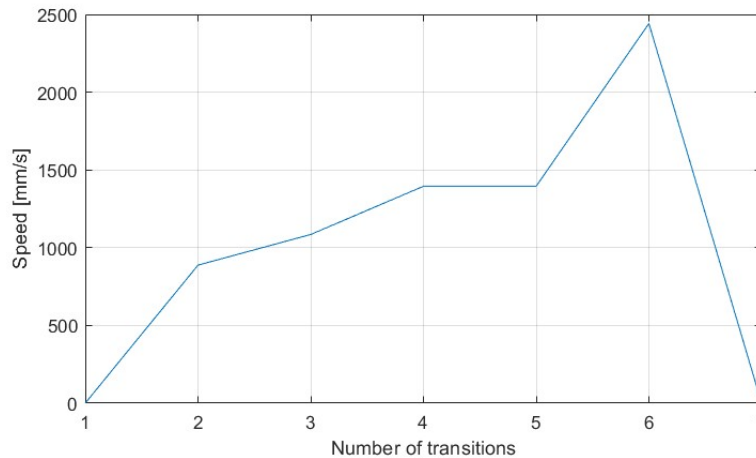


Figure 4.3: Sensor system results for the first executed test, without camera nor LED, at 2.5 bar with 40 mm bands

This way, a second set of tests with the high-speed camera and with the high-power LED setup was conducted. These tests aimed to have a first perception of the performance of the system.

This second set consisted of four shots of the bar, at four different pressures: 10, 7.5, 5 and 2.5 bar. It was noticed that at this point in the development of the system, it was not capable of detecting the transitions between bands, at any of the pressures, and no information was recorded by the system.

Since in the tests of first set, coherent data was obtained, one of the considered possibilities was that the high intensity of the LED illumination could affect the normal operation of the reflective sensor.

With this hypothesis, it was decided to add visors to the sensor support to reduce the brightness coming from the LED light, making sure that only the radiation emitted by the sensor itself was captured by the phototransistor. It was also decided to add an oscilloscope to the setup in order to analyze the signal coming from the sensor, to ensure that it actually had variations.

The oscilloscope was connected to the sensor, with two signals under analysis: digital and analog. In this way, it would be possible to know if the digital signal would correspond to the analog signal or if it would be necessary to change the value of the built-in comparator. A representation of the circuit with the connected oscilloscope is represented in the figure 4.4.

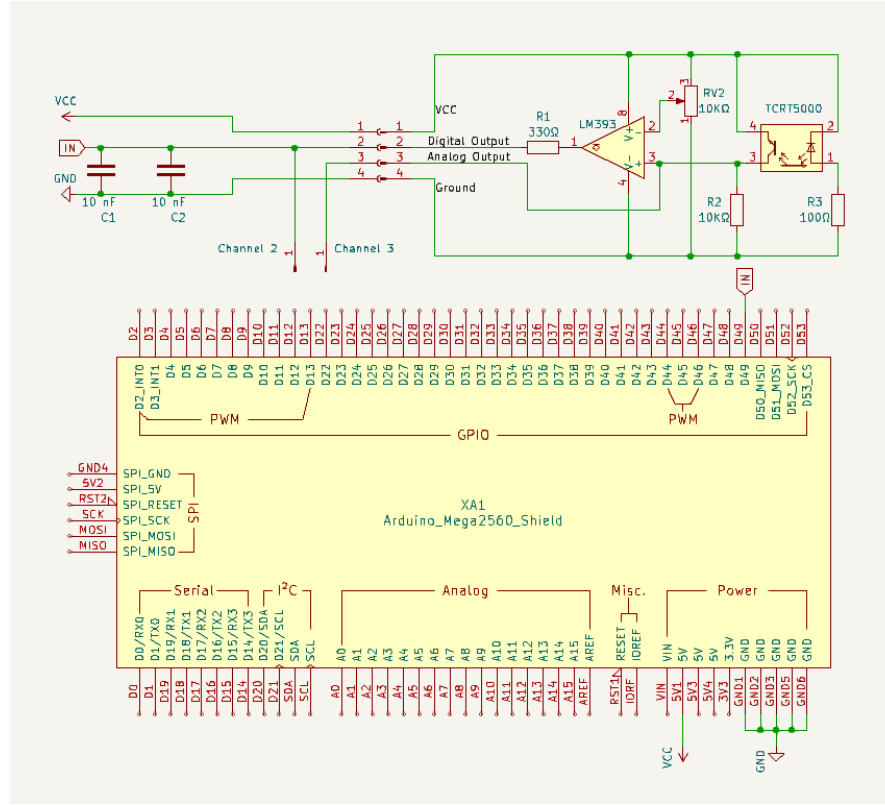


Figure 4.4: Connection between oscilloscope and sensor module circuit representation

The digital and analog signals from the sensor in each test were recorded on the oscilloscope, using the “single” mode. When specific trigger conditions, defined by the user, are detected while using this trigger mode, the oscilloscope performs a single acquisition of the signals in a certain time interval [61]. When the time interval ends, the acquisition stops and the obtained data is registered.

The visors consisted of small pieces of plastic glued to the support, with enough height to touch the striker bar and cover the sensor, but without influencing the movement of the bar. However, these visors were not enough, and a cover around the bar and the sensor was added, so that the light from the illumination system was prevented from reaching the sensor. This cover consisted of a sheet of paper, and later on a piece of cardboard, since an expedited solution was required. This way, it would be necessary to create a more robust cover.

With the cover added and the oscilloscope mounted, the tests of a third set proceeded. Four tests were carried out, but this time with only two pressures: 10 bar (maximum pressure) and 2.5 bar (minimum pressure). If the system correctly detected and calculated the transitions and velocities, respectively, it would be safe to assume that it would work correctly for any test pressure presented in this range.

After carrying out the tests, analyzable data was obtained by the DIC of the images obtained from the high-speed camera, by the oscilloscope and by the developed system. The data from the first two methods required processing routines, since it was necessary to make filtering and

calculations so that they could be compared with the values obtained by the sensor.

4.1.1 DIC results

The images from the Photron camera were captured at a rate of 50000 fps, with a shutter speed of 1/200000 seconds. with a 50 mm lens. These images were processed in GOM Correlate software. It is important to point out that, even having used a DIC program, the treatment that the images underwent should not be considered true DIC since it does not follow the usual practices of this method. Namely, a speckle pattern was not used, since a smooth pattern of bands was needed to not interfere with the functioning of the sensor. Thus, this program was used to point tracking, since it was an easy-to-use program with powerful tools at a computational level, obtaining the necessary speed results.

In order to carry out a correct analysis, it was necessary to calibrate the software for each test, defining the scale of both distance and time. The distance scale was calibrated by associating a pixel distance with a real distance, in millimeters. For this, it was used the length of a band, associating the beginning and end, in a certain frame, with a distance of 40 mm. For temporal calibration, manual calibration was not necessary, as the program imported the necessary information from the acquired images. If this did not happen, it would be necessary to indicate the time interval between two frames, that is the inverse of the frame rate that has been used, in this case 50000 fps.

Next, a certain number of points, which varied from test to test, but having been defined between 3 to 6 points per test, were associated with the set of images for each test, so that there was always at least one point that could be analyzed. For this, the points were positioned at the transitions of the bands. This way, it would be known whether the point was following the movement of the bands correctly. Otherwise, a displacement of the original position would be noticed. For a better result, the points were positioned in image locations where the quality of the pattern, which is an intrinsic dimensionless parameter of GOM Correlate, was of greater value. Normal values for this parameter would be between 0 and 4, as no value higher than 4 was found when searching for the best spots to define the points. Usually, these spots had values of pattern quality between 2 and 3.5.

With this parameterization, the displacement and velocity of each point were graphically represented. The displacement could only be calculated for the first point since it was the only point with a position reference, as it was the first point to be visible, and therefore analyzed, in the test video. When it left the analysis zone, its analysis would stop. However, this did not affect the calculation of the speed of the other points, which proved to be effective and consistent, making it possible to analyze the speed of the bar throughout the entire test, in the various tests performed.

After calculating the velocity of all points, throughout all tests, it was possible to create graphs of the points speed for each test. These graphs are shown in figures 4.5, 4.8, 4.10 and 4.12.

These values were further processed using Matlab. It was possible to filter the data, in order to make the analysis easier and less confusing. However, the data needed to be corrected to be possible to apply a low pass filter.

To correct the data from a single test, the first points' speed information, i.e., the speed information of the points that registered the movement until impact, was merged, in order to have unified information. In some cases, it was needed to replace zero values or very low values, when compared to adjacent ones, with linear regression ones, so the data would be more uniform to be filtered. These values may appear for some lack of image matching when the points are analyzed in GOM Correlate. Sometimes, it may only decrease the registered speed, but the points can disassociate and no longer correspond to the place where they were marked, causing an absence of movement analysis. All this data correction avoids problems that could affect the later filtering.

In figure 4.5, data from the first test at 10 bar is shown, where it is possible to see the speed curves of each point. In figure 4.6, merged and corrected data of the same test is shown.

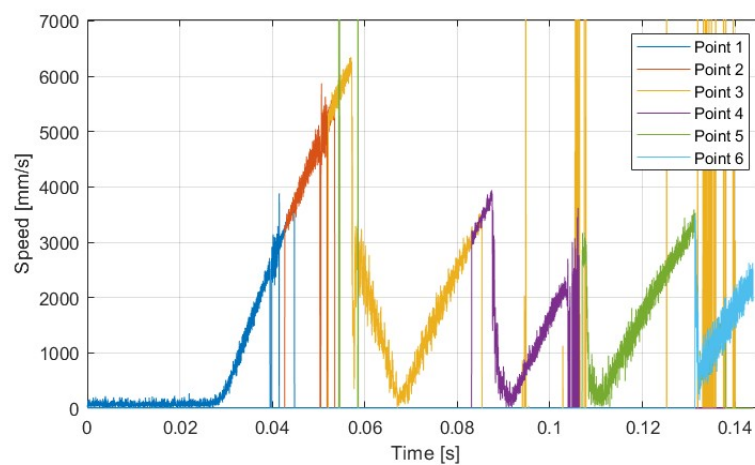


Figure 4.5: DIC results for the first test at 10 bar with 40 mm bands

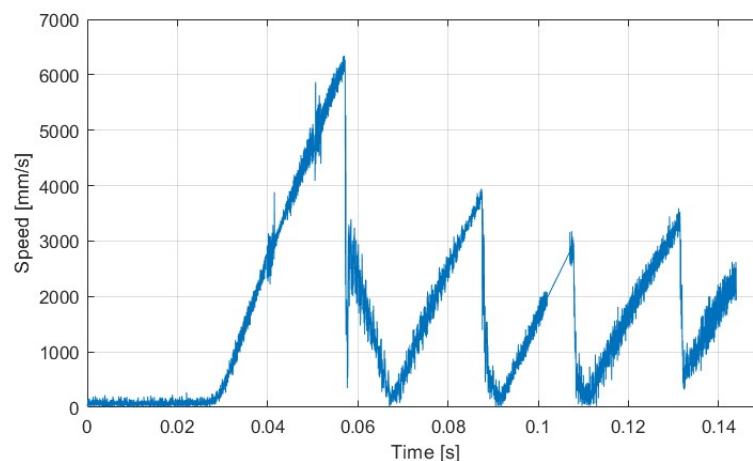


Figure 4.6: DIC results for the first test at 10 bar with 40 mm bands

After data correction, a low pass filter was applied. Filter parameters were defined so the results were clear enough to be analyzed. To transform the data, it was used the Signal Analyzer

application from Matlab, to create a script with the filter parameters. The filter was applied to all data of the DIC tests. The parameters used are indicated in table 4.1

Table 4.1: Lowpass filter conditions

Passband Frequency [Hz]	2500
Steepness	0.85
Stopband Attenuation [dB]	60

After filtering each test speed curve, more easily analyzable graphics were obtained, with which it was possible to estimate the maximum speed of the striker bar, before impact. In these new graphs, the maximum speeds displayed are much higher than the speeds immediately before. This does not correspond to reality, as nothing increases the speed of the bar just before it hits the incident bar. Thus, by analyzing the graph of the filtered data, the speed evolution trend was followed and possible results for real maximum speeds were considered. An example of this can be seen on figure 4.7. The maximum speed shown by the graph is indicated by the red point, while the green point represents the considered true maximum speed of the test, by following the speed curve. This analysis was made to all tests and it is summarized in table 4.2. The speed curves of the other tests can be seen in figures 4.9, 4.11 and 4.13.

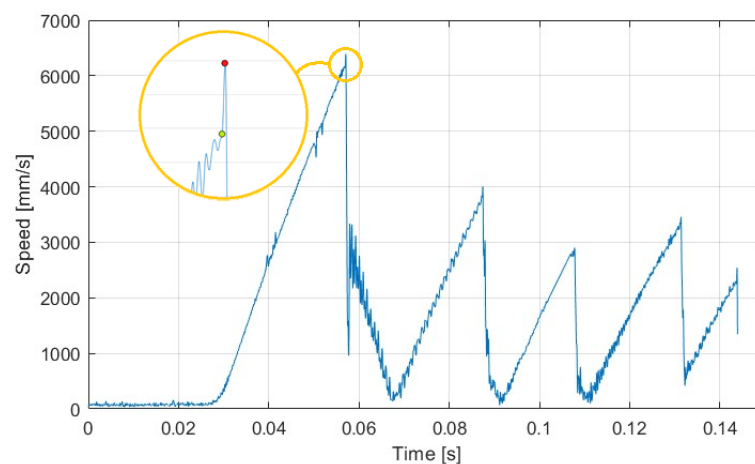


Figure 4.7: Filtered DIC results for the first test at 10 bar with 40 mm bands

Table 4.2: DIC analysis maximum speeds from each test with 40 mm bands

Test	Max speed [mm/s]	Considered max speed [mm/s]
1° test at 10 bar	6392.4	6185
2° test at 10 bar	6427.1	6150
1° test at 2.5 bar	3607.3	3450
2° test at 2.5 bar	2222.2	2100

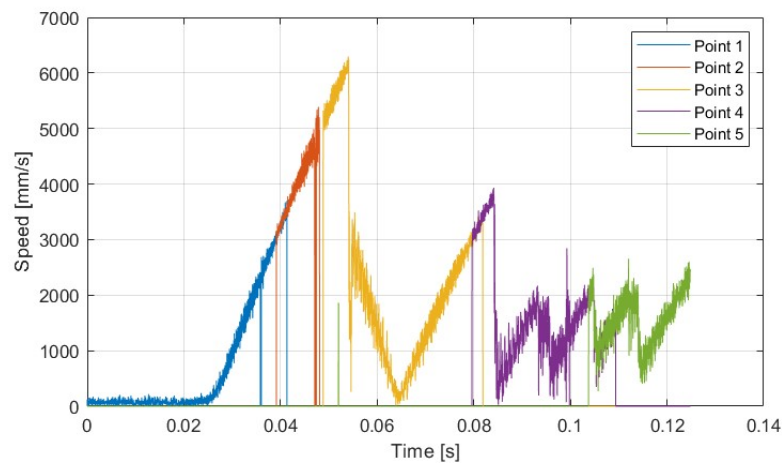


Figure 4.8: DIC results for the second test at 10 bar with 40 mm bands

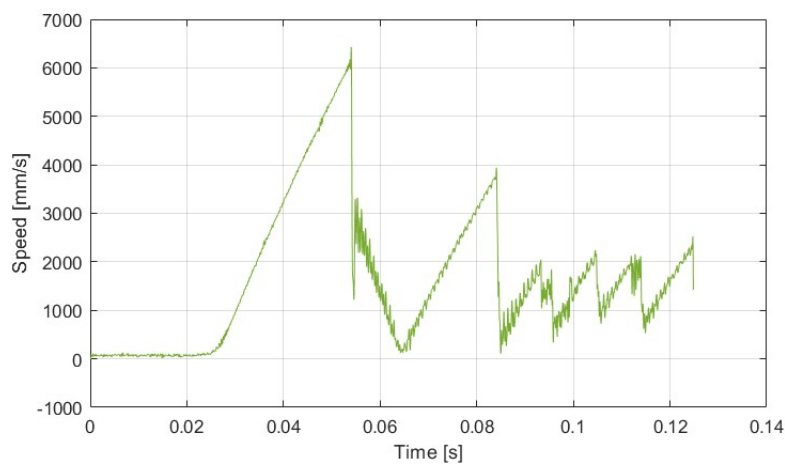


Figure 4.9: Filtered DIC results for the second test at 10 bar with 40 mm bands

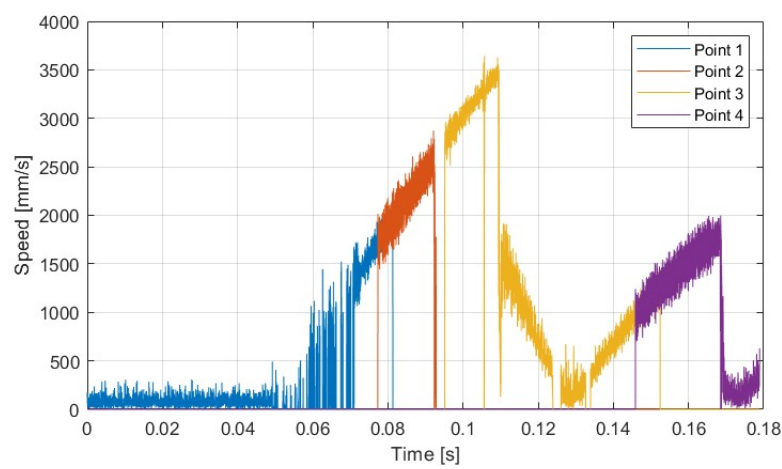


Figure 4.10: DIC results for the first test at 2.5 bar with 40 mm bands

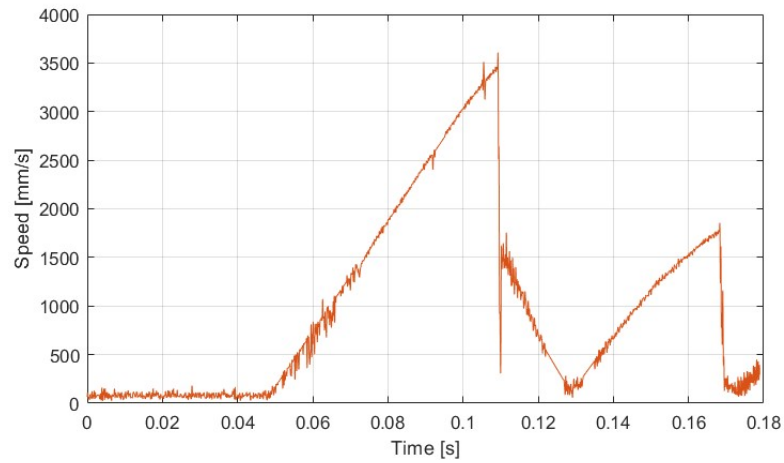


Figure 4.11: Filtered DIC results for the first test at 2.5 bar with 40 mm bands

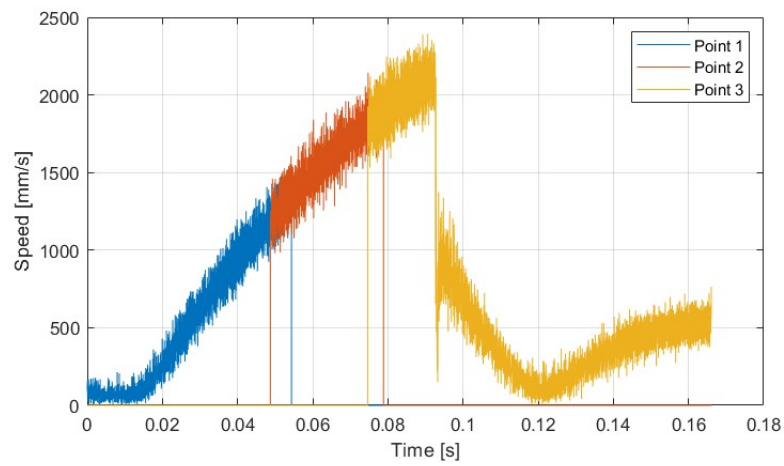


Figure 4.12: DIC results for the second test at 2.5 bar with 40 mm bands

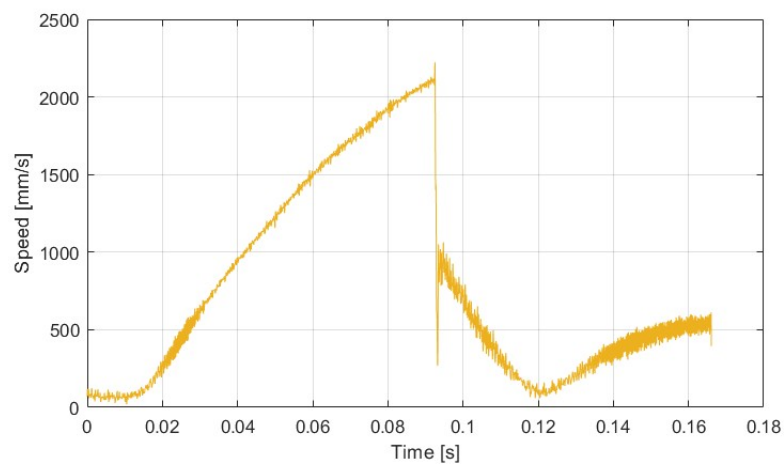


Figure 4.13: Filtered DIC results for the second test at 2.5 bar with 40 mm bands

This DIC data analysis routine is summarized in figure 4.14.

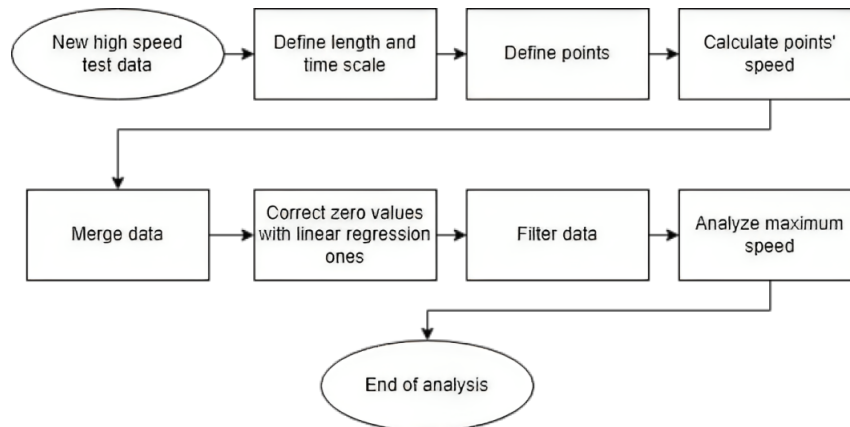


Figure 4.14: DIC data analysis routine

4.1.2 Oscilloscope results

The oscilloscope data was also processed in Matlab. This consisted in transforming the voltage values of the sensor's digital signal, registered by the oscilloscope, in striker bar speed values.

As the digital signal is dependent on the analog signal, their behaviors are similar, but the signals have different natures. The analog signal consists of values inversely proportional to the amount of reflected radiation, while the digital signal consists of only two possible voltage values, which will be associated with logic values. These logical values are the product of comparing the analog signal with a threshold value. If the value of the analog signal is greater, the digital signal will have a true logic value. Otherwise, it will have a false logic value. These logic values are, therefore, associated with the color of the bands. If the band is black, the sensor will associate it with a logical value of 1, and, thereby, a voltage of 5V, and if it is a white band, it will associate it with a logical value of 0, and, thereby, a voltage of 0V. This happens due to the architecture of the sensor itself, since it is a proximity sensor. The analog signal will be displayed with values proportional to the voltage range.

Both signals were recorded so that it was possible to confirm that all transitions were detected, comparing the analog and digital signals. If the analog signal presents any transition not registered in the digital signal, the comparator must be adjusted. These signals are shown in figures 4.15, 4.17, 4.19 and 4.21. As it is possible to see in 4.15 and 4.17, the analog signal detected a transition that was not registered in digital signal.

With the digital signal, it is possible to perceive at what instant of time there is a positive transition, that is, when the voltage value changes from 0 to 5V, which corresponds to the beginning of a new timing between black bands.

To make it easier, this signal was normalized, making all values greater than 2V take on a logical value of 1. This way, only logical values are worked with, facilitating forward processing.

To detect transitions between bands, subtractions of consecutive values were performed. This returned values between -1 and 1, having only used values equal to 1. With this, it was possible to show at which instants positive transitions occurred, so it would be possible to understand at which moments new transitions between bands began. After this, data was merged, so that only information remained regarding the instants of time in which positive transitions occurred. This facilitated speed calculation and subsequent analysis, as a velocity evolution graph could be created. The striker bar speed was, then, calculated by dividing the value of the distance of two bands, one black and one white, which in this case was 80 mm, by the time difference between consecutive positive transitions.

After processing, it was possible to obtain velocity graphs as a function of number of detected transitions. This should prove that the values obtained by the sensor were real. However, the veracity of these values can only be proven by the data obtained by the DIC. The obtained speed graphs are represented in figures 4.16, 4.18, 4.20 and 4.22, and the maximum registered speeds are summarized in table 4.3.

Table 4.3: Oscilloscope analysis maximum speeds from each test with 40 mm bands

Test	Max speed [mm/s]
1° test at 10 bar	4164.1
2° test at 10 bar	4099
1° test at 2.5 bar	2251.8
2° test at 2.5 bar	1695.8

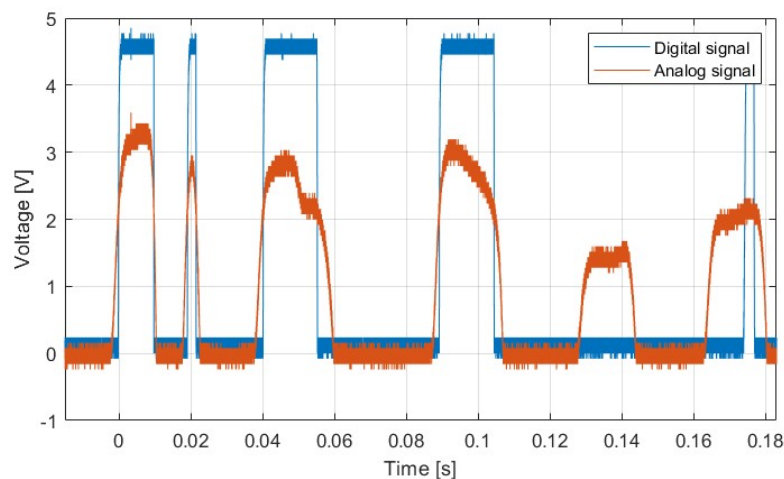


Figure 4.15: Oscilloscope results for the first test at 10 bar with 40 mm bands

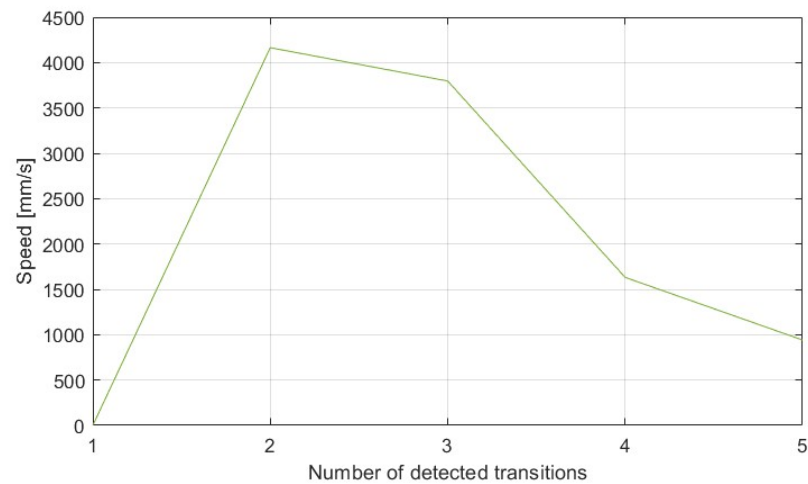


Figure 4.16: Processed oscilloscope results for the first test at 10 bar with 40 mm bands

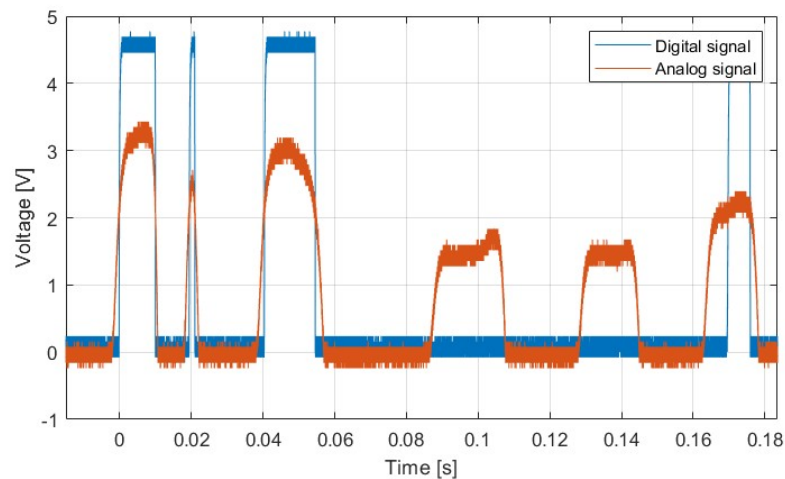


Figure 4.17: Oscilloscope results for the second test at 10 bar with 40 mm bands

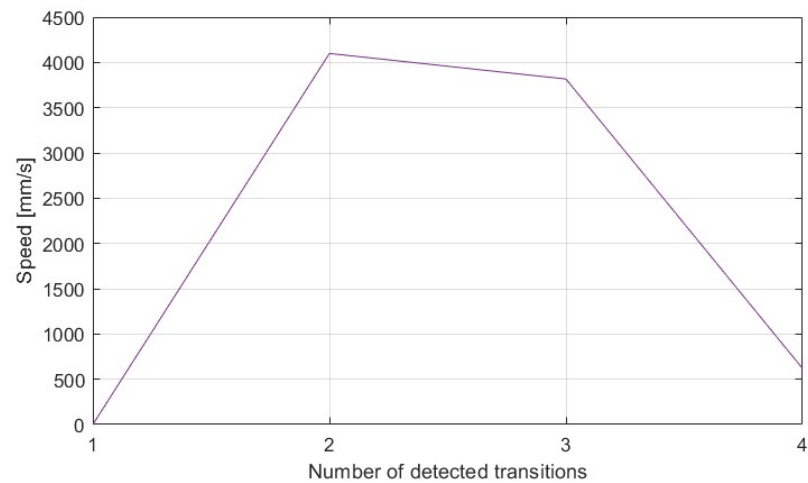


Figure 4.18: Processed oscilloscope results for the second test at 10 bar with 40 mm bands

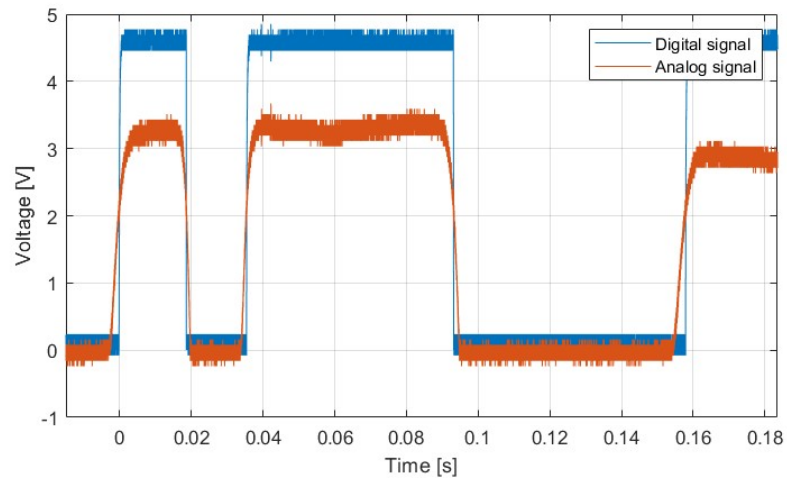


Figure 4.19: Oscilloscope results for the first test at 2.5 bar with 40 mm bands

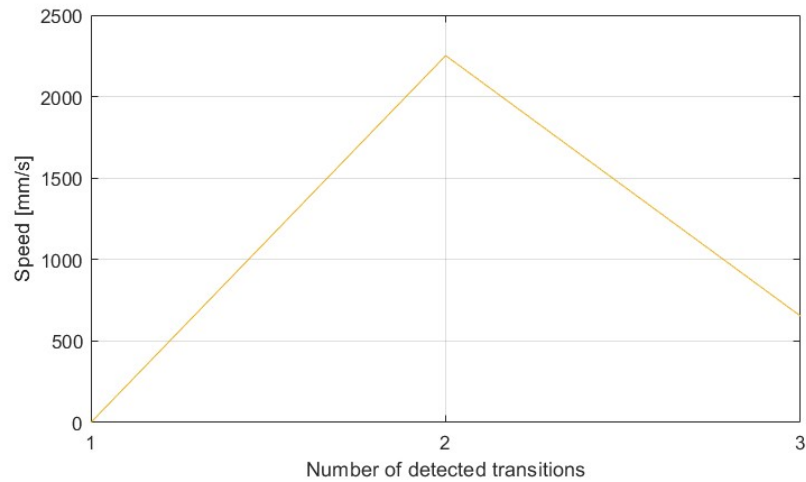


Figure 4.20: Processed oscilloscope results for the first test at 2.5 bar with 40 mm bands

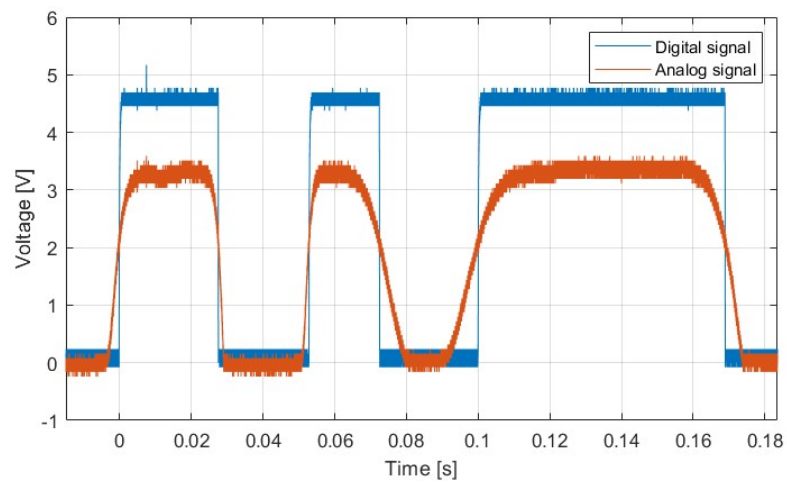


Figure 4.21: Oscilloscope results for the second at 2.5 bar with 40 mm bands

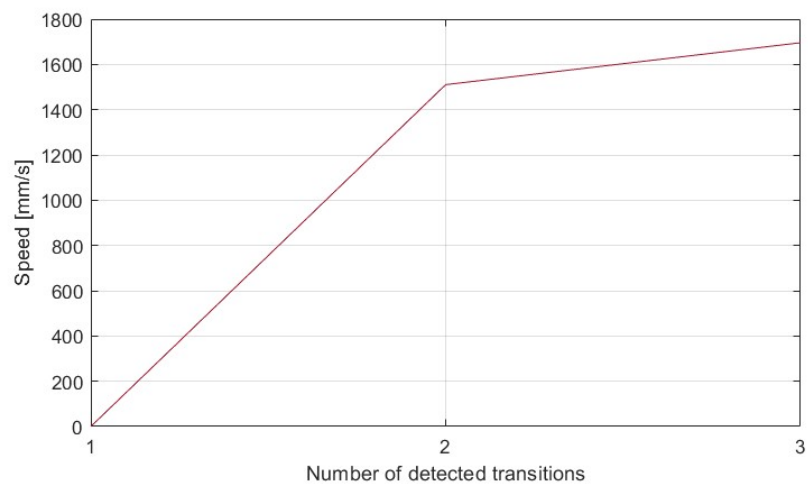


Figure 4.22: Processed oscilloscope results for the second test at 2.5 bar with 40 mm bands

This oscilloscope data analysis routine is summarized in figure 4.14.

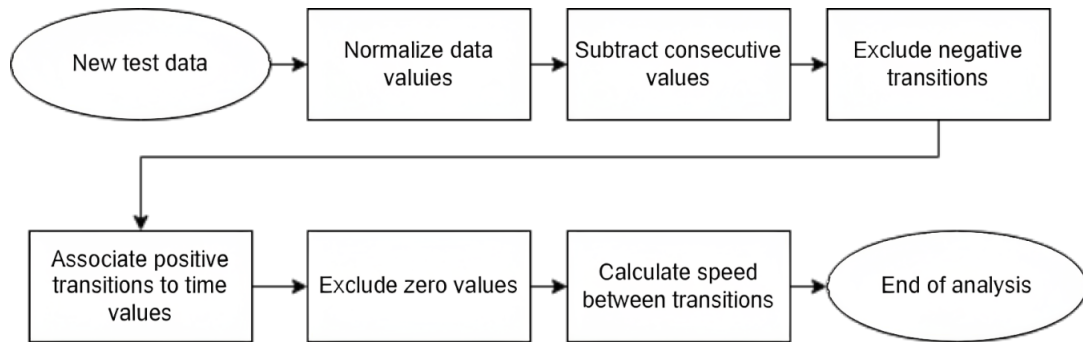


Figure 4.23: Oscilloscope data analysis routine

4.1.3 Sensing system results

The values obtained by the sensing system did not require any type of processing. These results can be analyzed in figures 4.24, 4.25, 4.26 and 4.27, and the maximum registered speeds are summarized in table 4.4.

Table 4.4: Sensor system analysis maximum speeds from each test with 40 mm bands

Test	Max speed [mm/s]
1° test at 10 bar	4164.7
2° test at 10 bar	4098.4
1° test at 2.5 bar	2251.8
2° test at 2.5 bar	3816.4

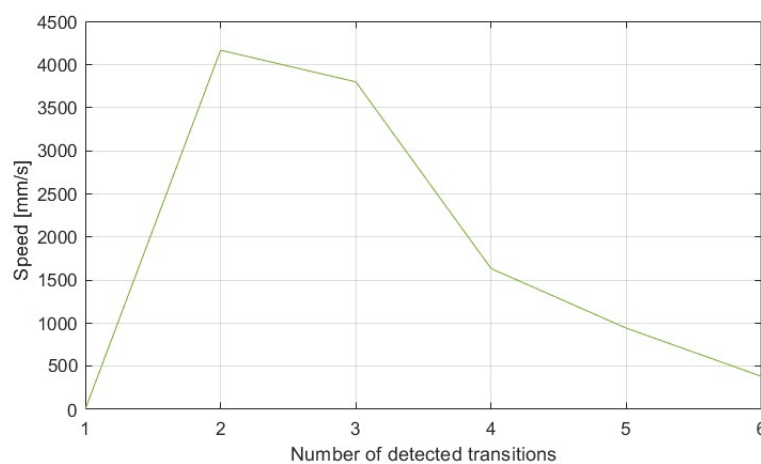


Figure 4.24: Sensor system results for the first test at 10 bar with 40 mm bands

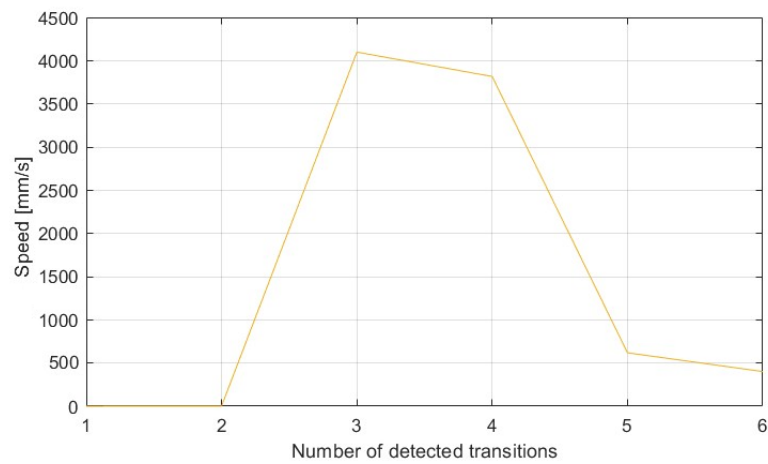


Figure 4.25: Sensor system results for the second test at 10 bar with 40 mm bands

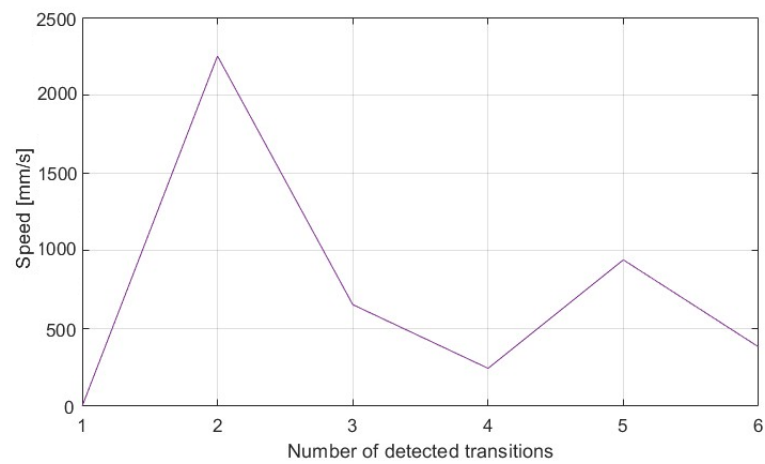


Figure 4.26: Sensor system results for the first test at 2.5 bar with 40 mm bands

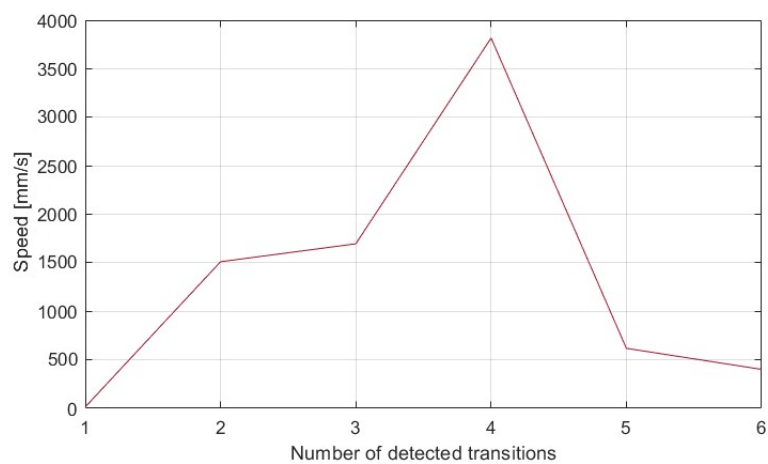


Figure 4.27: Sensor system results for the second test at 2.5 bar with 40 mm bands

4.1.4 Discussion

When analyzing the oscilloscope data, it became evident that the comparator needed to be tuned because there were transitions registered by the analog signals in tests at 10 bar, figures 4.15 and 4.17, which were not registered in the corresponding digital signal. These unregistered transitions occurred after the impact of the striker bar with the incident bar, not directly affecting the analysis regarding the maximum speed of the striker bar. Regardless, it showed a flaw that can be easily fixed, to avoid future errors.

Comparing the data obtained from the oscilloscope and the sensor system, the values of both systems are mostly in agreement, except for some discrepancies. It is possible to see that the sensor system data shows more transitions than the oscilloscope detected.

For extra detected transitions before impact, as seen in the second test at 10 bar, in figures 4.18 and 4.25, considering the speed values associated with this data, it is possible to say that it resulted from an excessively long timing. This can happen because the Arduino code never stops counting, once it starts. This way, the system will keep counting time between tests, only restarting when it detects a new transition. This will generate time intervals that are too long, which will lead to an almost zero speed.

For extra detected transitions after impact, as seen in all tests, the reason there are more transitions on the sensor is the simple fact that the oscilloscope had a fixed time to detect the signals, unlike the sensor system, which requires the user to start and stop the system from registering data. This way, the system can analyze various transitions without any time restriction. This feature is sufficient to collect more data because the test may last longer than expected, resulting in the oscilloscope does not detect all the transitions, but only a few more after the impact. In contrast, the sensor system records all transitions during the test, due to the fact that its operation has to be stopped by the user, rather than timed. This is not a problem, since the oscilloscope was able to detect transitions at the moment of impact and a short time afterwards.

Between the sensor system data and the DIC data, it can be mentioned the lack of similarities in the speed evolution profiles, where in the first, there are no gradual evolutions, with more peaks of information being displayed. This may be explained by the size of the pattern being too large for the application in question. This can be shown by the lag between maximum speed values, meaning that, in the instant before the impact, there was no positive transition, and the speed at that time was not recorded.

It should also be noted that the maximum speed value recorded by the sensor in the second test at 2.5 bar, as shown in figure 4.27, does not make sense since there is a spontaneous increase in speed, and the natural evolution of this test from the third transition should be one of slowing down, as happened with the first test at the same pressure, as shown in figure 4.26. Comparing with the DIC values of the test in question, as shown in figure 4.13, it is seen that the speed increases again after impact. However, this new acceleration leads to speeds around 500 mm/s, ending the graphic with a plateau with this magnitude. This way, one might suspect that a new transition was detected at a moment near the recoil impact. This would explain the higher speed, since the timing

between detected new transitions would be smaller than intended.

4.2 Validation results with 5 mm bands

After analyzing the results of the 40 mm band tests, some areas of improvement were noticed.

Thus, it was decided to reduce the size of the pattern to a 5 mm band size. With this pattern, the system would be analyzing transitions with a distance of 10 mm, instead of the 80 mm of the previous pattern. This way, much more data will be collected, obtaining a much more gradual velocity evolution, which will be much closer to the real evolution of the bar. This new pattern was again printed on an A3 sheet, following the same method to tape it to the bar. During this process, care was taken to ensure that it was not damaged before being applied in order to reduce possible imperfections.

Since the new size of the pattern was so reduced, the sensor was brought closer to the bar, so that the reflection of the radiation emitted by the sensor itself was captured as best as possible, preventing it from returning inconclusive results regarding the color of the analyzed band.

As detected in previous tests, the sensor module comparator was also adjusted, causing lower values of the analog signal to result in true values of the digital signal. For this, it was tried to understand which voltage values were associated to each band color. To do this, a multimeter was used to measure the voltage on the terminals of the comparator integrated in the module. When a black band was positioned in front of the sensor, the voltmeter registered a voltage of 3.016V, but when it was a white band in front of the sensor, the voltmeter registered a voltage of 0V. Based on these values, and with previous test attempts, the comparator value was defined as 0.3V, in order to try to ensure that all transitions would be recorded in the digital signal.

With this new setup and new parameterization, the tests were carried out, which consisted of three tests at a pressure of 10 bar and three tests at a pressure of 2.5 bar. The subsequent analysis of the data was done in the same ways as the analysis of the previous tests.

4.2.1 DIC results

Although six tests were analyzed, only five of them were analyzed with a high-speed camera. This test, which was not recorded by mistake, showed a different behavior from the rest of the tests, so it was decided to analyze it with the other two methods.

The recordings of these tests were made with the same camera under the same recording conditions as the previous ones. The data from these recordings were also analyzed in the same software, having undergone a subsequent filtering with the previously used parameters.

Thus, the obtained data is summarized in table 4.5 and in figures from 4.28 to 4.37.

Table 4.5: DIC analysis maximum speeds from each test with 5 mm bands

Test	Max speed [mm/s]	Considered max speed [mm/s]
1° test at 10 bar	5900.2	5650
2° test at 10 bar	5269.8	5100
1° test at 2.5 bar	1941.2	1850
2° test at 2.5 bar	1351.9	1300
3° test at 2.5 bar	2367.5	2275

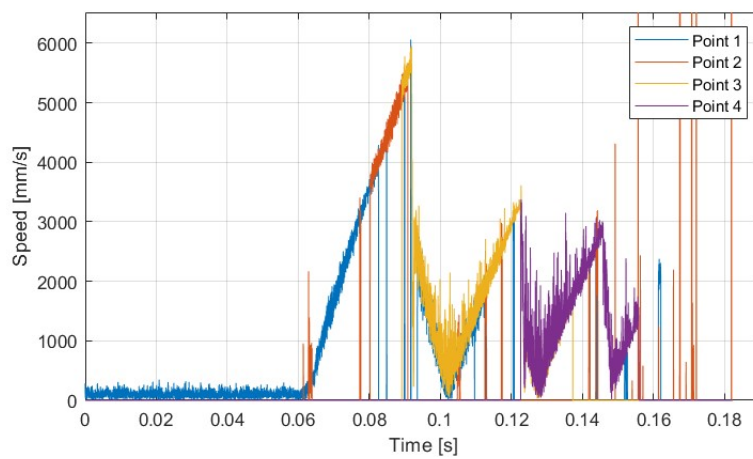


Figure 4.28: DIC results for the first test at 10 bar with 5 mm bands

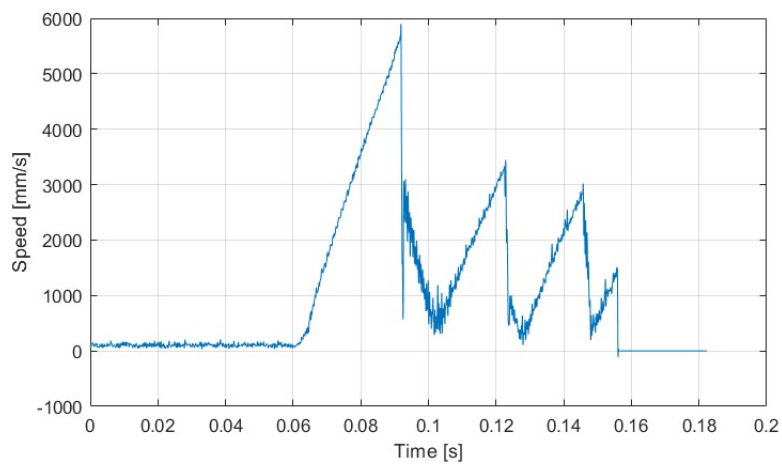


Figure 4.29: Filtered DIC results for the first test at 10 bar with 5 mm bands

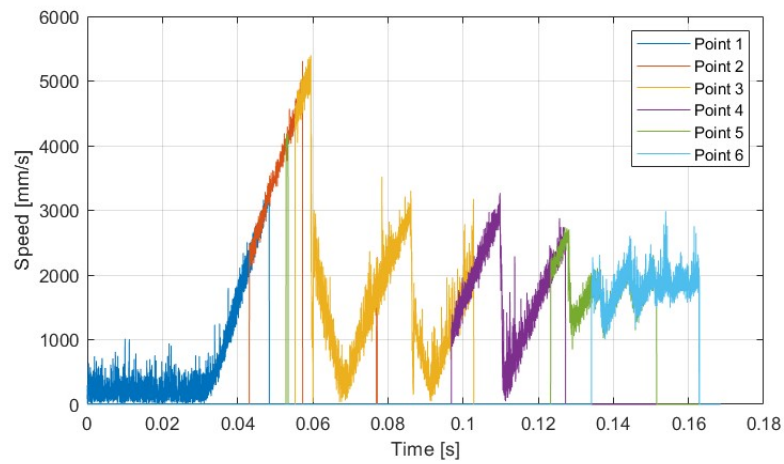


Figure 4.30: DIC results for the second test at 10 bar with 5 mm bands

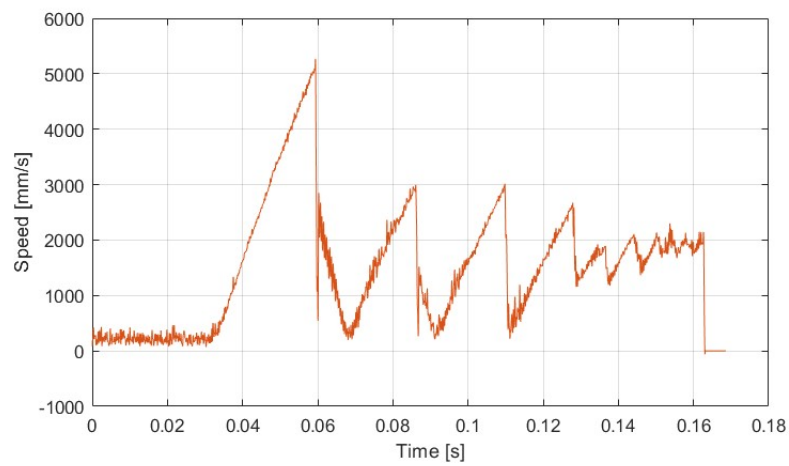


Figure 4.31: Filtered DIC results for the second test at 10 bar with 5 mm bands

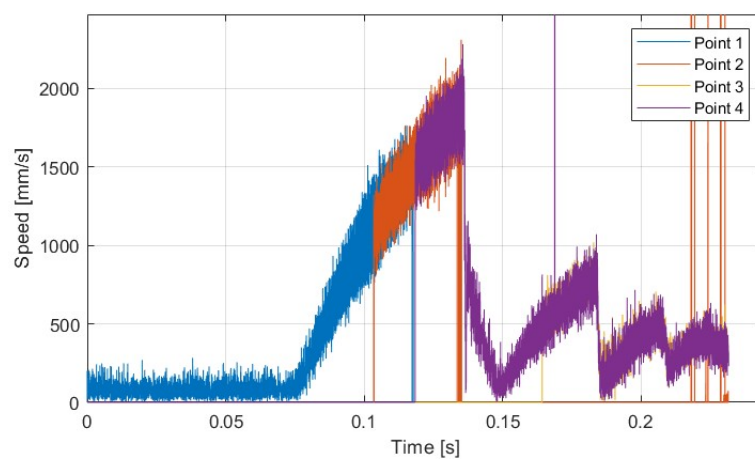


Figure 4.32: DIC results for the first test at 2.5 bar with 5 mm bands

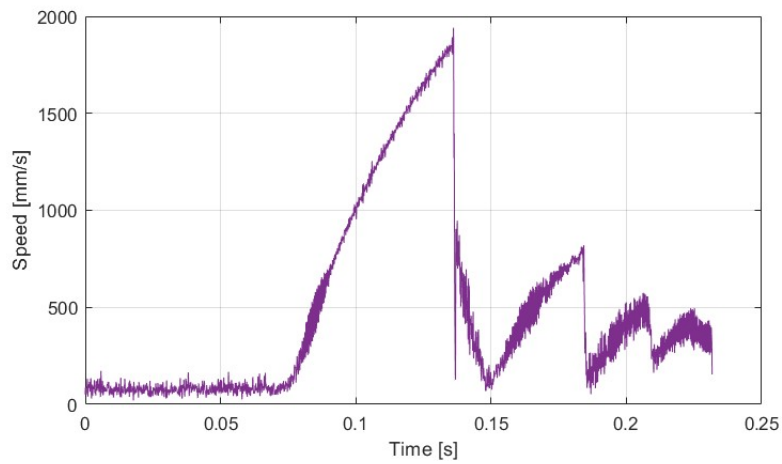


Figure 4.33: Filtered DIC results for the first test at 2.5 bar with 5 mm bands

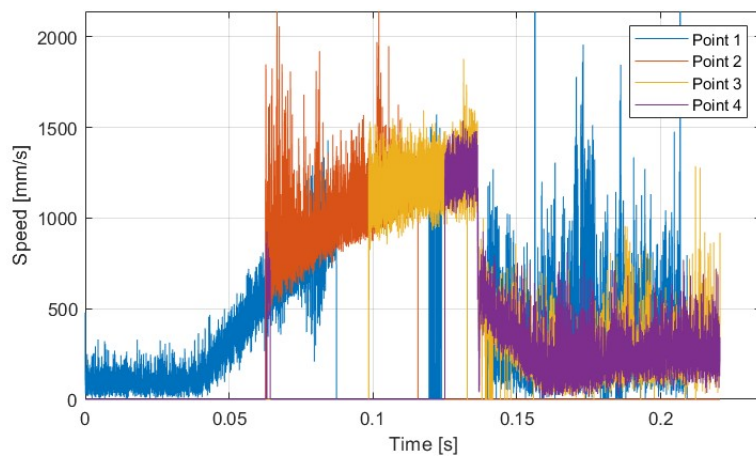


Figure 4.34: DIC results for the second test at 2.5 bar with 5 mm bands

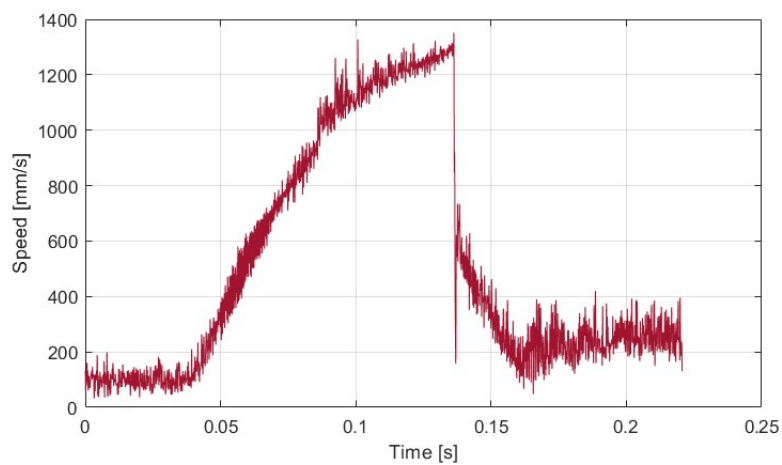


Figure 4.35: Filtered DIC results for the second test at 2.5 bar with 5 mm bands

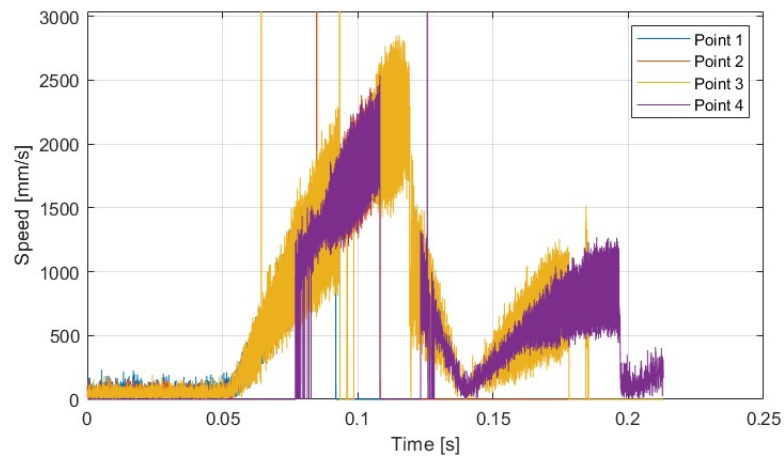


Figure 4.36: DIC results for the third test at 2.5 bar with 5 mm bands

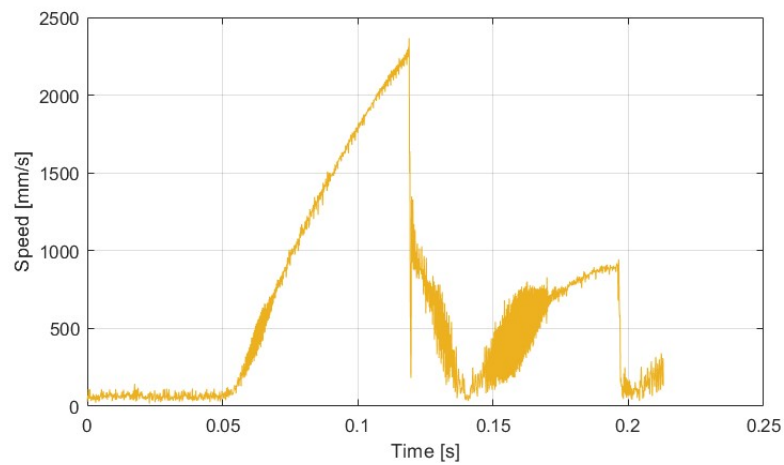


Figure 4.37: Filtered DIC results for the third test at 2.5 bar with 5 mm bands

4.2.2 Oscilloscope results

The oscilloscope data obtained from these tests was processed using the already used routine.

Thus, the registered and processed oscilloscope data is summarized in table 4.6 and in figures from 4.38 to 4.49.

Table 4.6: Oscilloscope analysis maximum speeds from each test with 5 mm bands

Test	Max speed [mm/s]
1° test at 10 bar	5549.4
2° test at 10 bar	4699.2
3° test at 10 bar	9940.4
1° test at 2.5 bar	1929.0
2° test at 2.5 bar	1210.9
3° test at 2.5 bar	2235.1

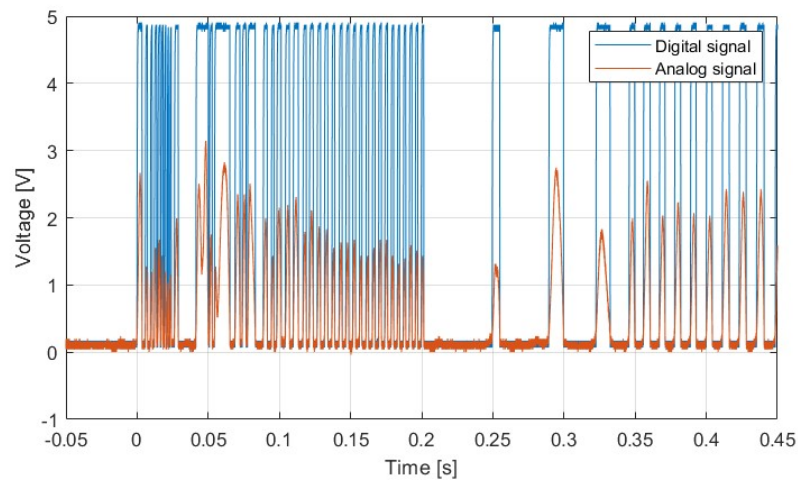


Figure 4.38: Oscilloscope results for the first test at 10 bar with 5 mm bands

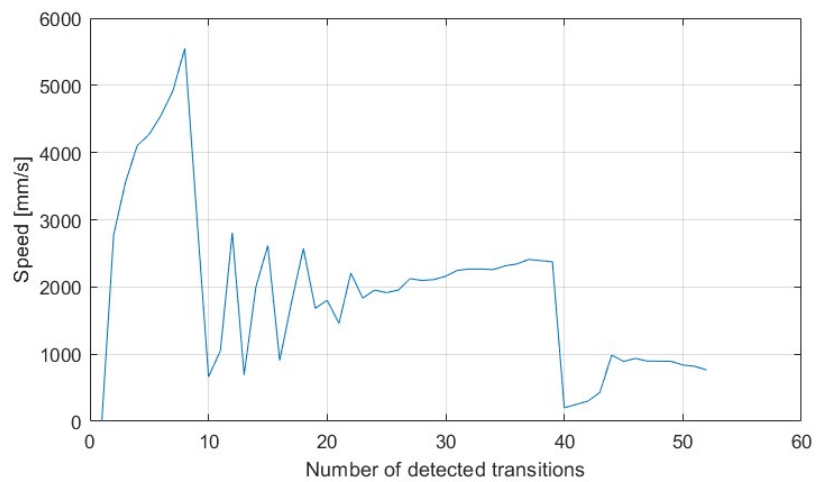


Figure 4.39: Processed oscilloscope results for the first test at 10 bar with 5 mm bands

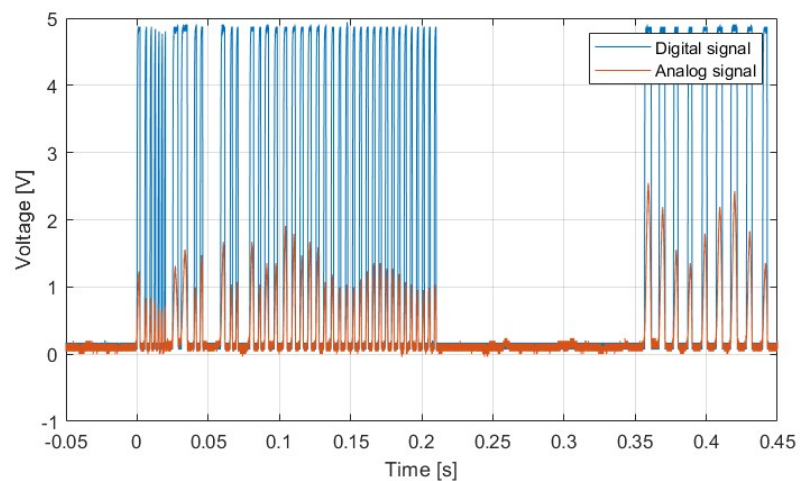


Figure 4.40: Oscilloscope results for the second test at 10 bar with 5 mm bands

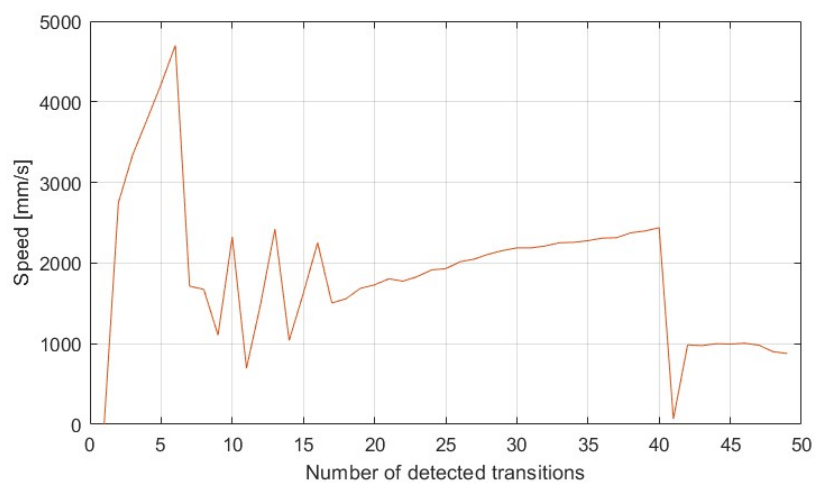


Figure 4.41: Processed oscilloscope results for the second test at 10 bar with 5 mm bands

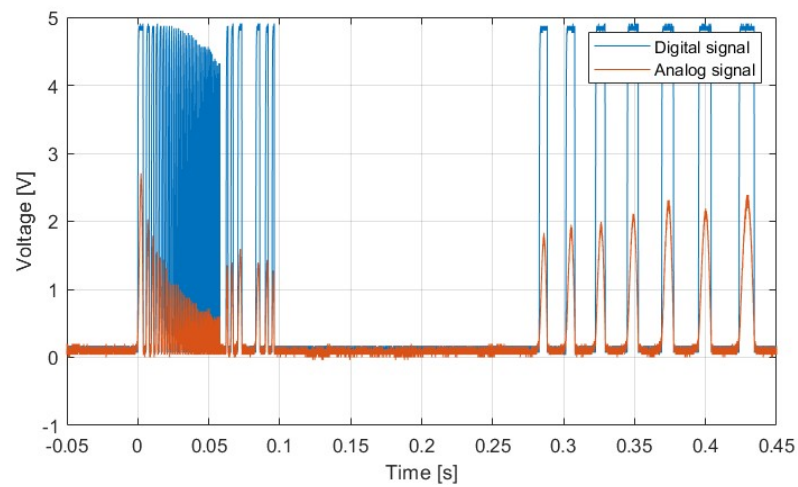


Figure 4.42: Oscilloscope results for the third test at 10 bar with 5 mm bands

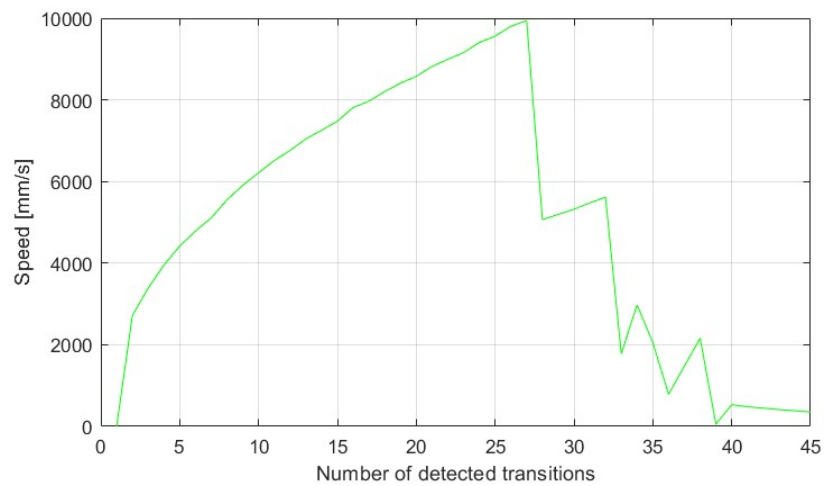


Figure 4.43: Processed oscilloscope results for the third test at 10 bar with 5 mm bands

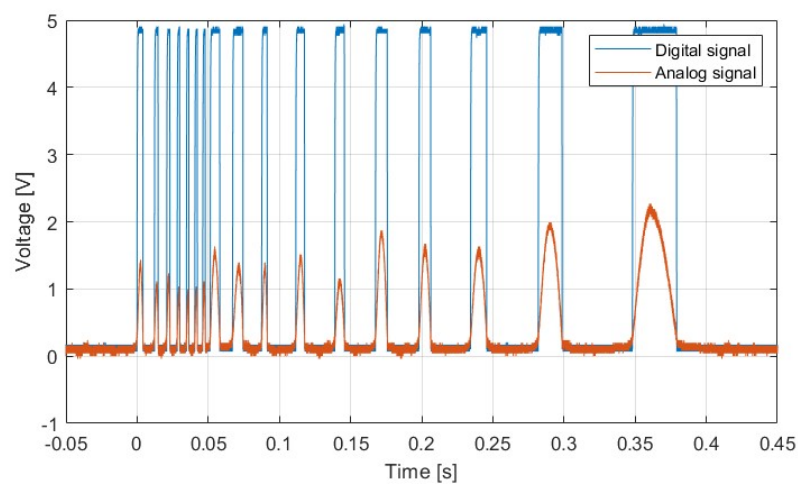


Figure 4.44: Oscilloscope results for the first test at 2.5 bar with 5 mm bands

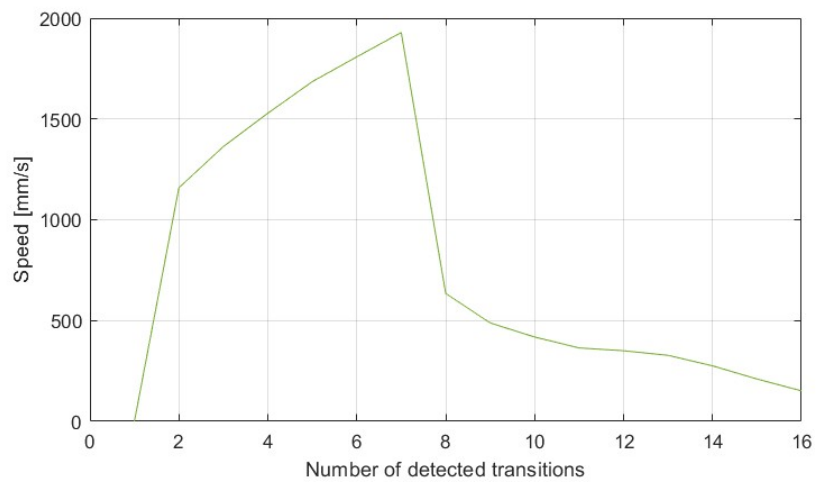


Figure 4.45: Processed oscilloscope results for the first test at 2.5 bar with 5 mm bands

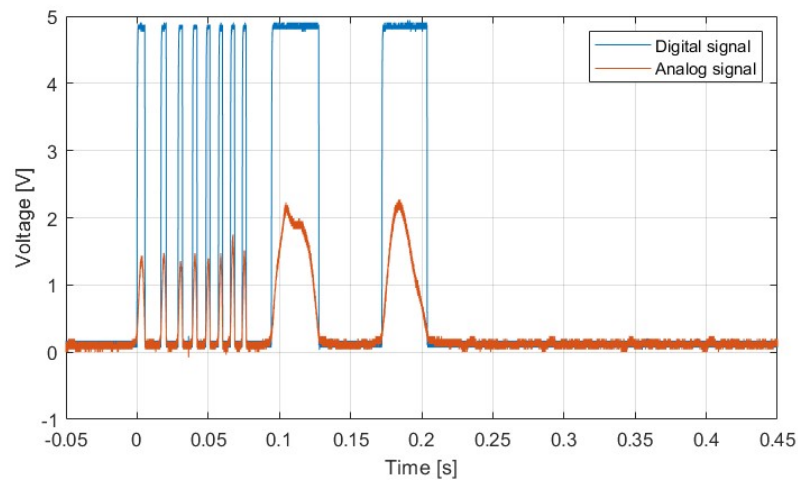


Figure 4.46: Oscilloscope results for the second test at 2.5 bar with 5 mm bands

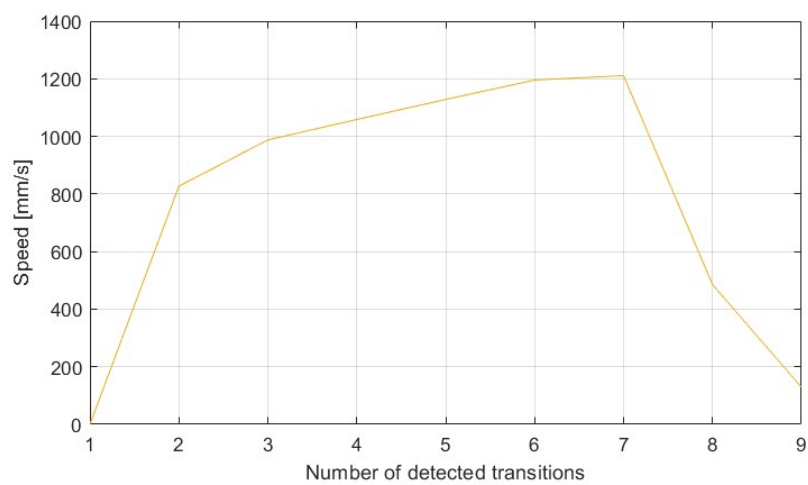


Figure 4.47: Processed oscilloscope results for the second test at 2.5 bar with 5 mm bands

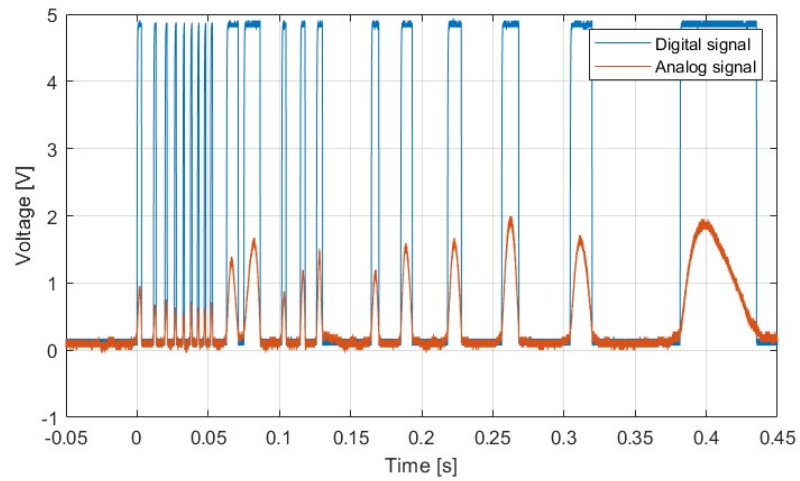


Figure 4.48: Oscilloscope results for the third test at 2.5 bar with 5 mm bands

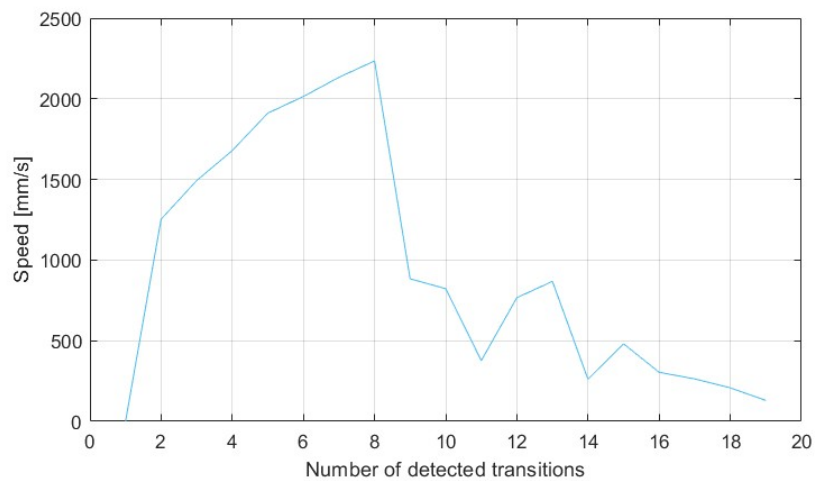


Figure 4.49: Processed oscilloscope results for the third test at 2.5 bar with 5 mm bands

4.2.3 Sensing system results

Once again, the values registered by the sensor system did not require further processing. This data is summarized in table 4.7 and in figures from 4.50 to 4.55.

Table 4.7: Sensor system analysis maximum speeds from each test with 5 mm bands

Test	Max speed [mm/s]
1° test at 10 bar	5547.5
2° test at 10 bar	4700.6
3° test at 10 bar	9412.9
1° test at 2.5 bar	2439.4
2° test at 2.5 bar	2439.4
3° test at 2.5 bar	2234.4

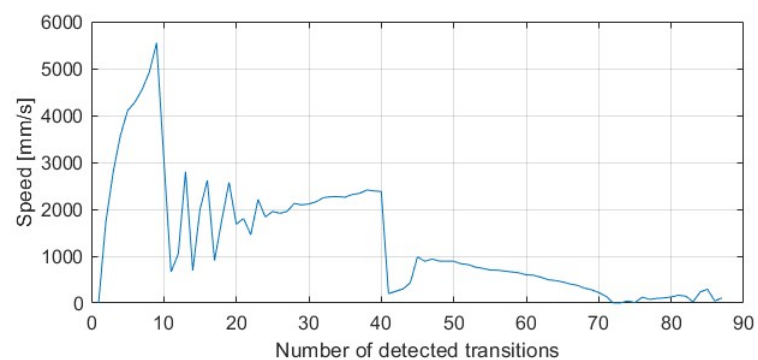


Figure 4.50: Sensor system results for the first test at 10 bar with 5 mm bands

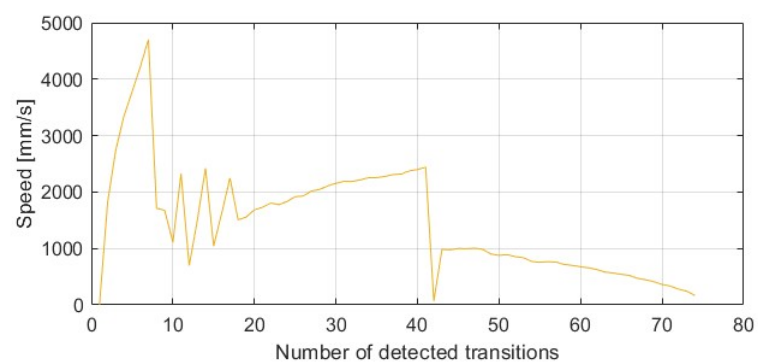


Figure 4.51: Sensor system results for the second test at 10 bar with 5 mm bands

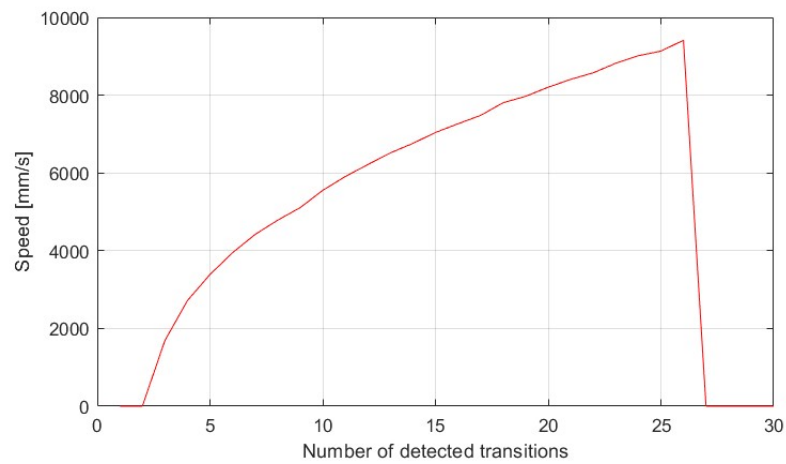


Figure 4.52: Sensor system results for the third test at 10 bar with 5 mm bands

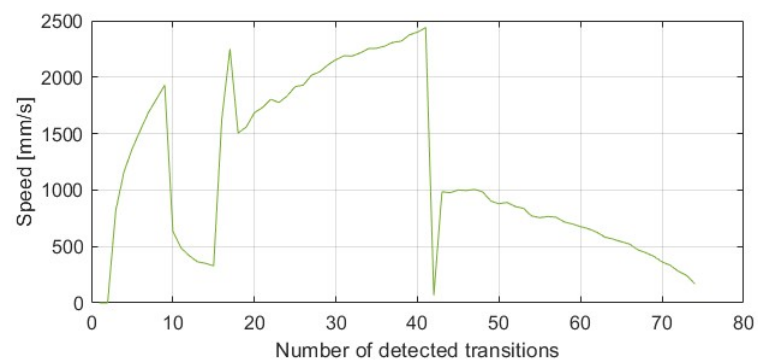


Figure 4.53: Sensor system results for the first test at 2.5 bar with 5 mm bands

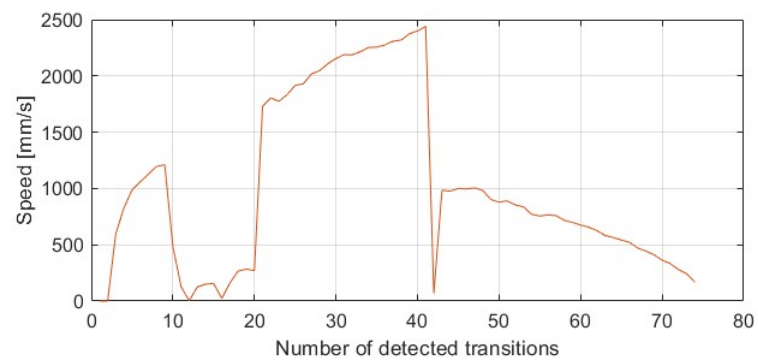


Figure 4.54: Sensor system results for the second test at 2.5 bar with 5 mm bands

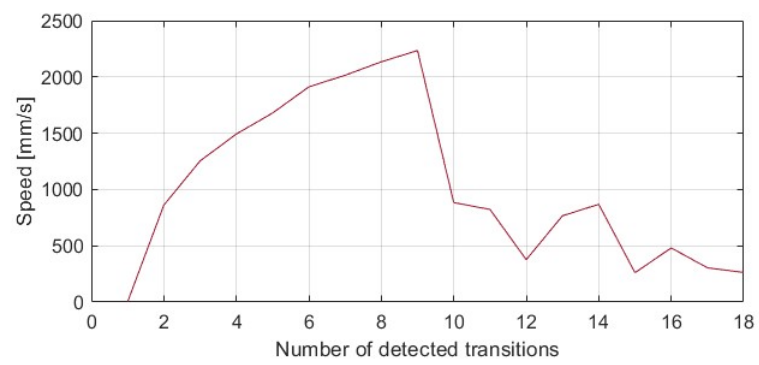


Figure 4.55: Sensor system results for the third test at 2.5 bar with 5 mm bands

4.2.4 Discussion

By analyzing the results of these last tests, it was possible to draw some conclusions.

Regarding the tests in a general context, it was possible to perceive that with a smaller pattern size, results were obtained that characterized the test in a more detailed and gradual way.

Comparing the maximum speed values of the various tests, across the three analysis methods, it was noticed that the oscilloscope and sensor system values, which were very similar, have already become closer to the DIC values. Even though these results were not identical, the speed curves from the sensor and the oscilloscope showed a velocity evolution much more similar to the curves obtained by the DIC, and to what would be expected from the normal movement of the projected bar. This evolution is described by the closer positive transitions of the digital signal, as speed increased. It is also described by the pre-impact velocity curve, which shows a more accelerated start, passing through an inflection point, and, if the movement had a slower speed, a more constant velocity plateau, which signals a slowdown of the group.

These aspects demonstrate a positive progression in the development of the system. However, some differences were still detected. While analyzing the 2.5 bar tests, namely the first, figure 4.53, and second ones, figure 4.54, it was noticed that the sensor system detected more transitions than the oscilloscope. This behavior by itself is natural, since it occurs after impact, which is marked by a velocity drop and practically zero speed interval. However, after this event, the bar accelerated again.

After comparing the graphics of the relevant tests and analyzing the developed user's interface software, the array which stores the velocity values wasn't cleared properly at the beginning of the two tests in question, since this array was only declared when the interface program is started, but is never properly reset, when the serial communication is reconnected. This way, the fact that the oscilloscope graphics, figures 4.45 and 4.47, for these tests do not show any new acceleration and end with values below 500 mm/s makes more sense. This speed makes it seem unlikely that a new impact due to the recoil of the striker bar would cause it to instantly move at a higher speed than it had at the initial impact with the incident bar. This is reinforced by the fact that the new acceleration portion of these graphs is nearly identical to the corresponding portion of the graphic for the second test at 10 bars, figure 2.14. This might have occurred because this test, during the testing phase, was conducted right before the 2.5 bar tests, not following order of presentation of this work, to facilitate the understanding of the results, due to the third test at 10 bar. The absence of this new acceleration in the results of the third test at 2.5 bars, figure 4.55 can be explained by the hypothesis that the user's interface program was restarted due to some occasional event. This would result in the resetting of all variables, including the mentioned array. Either way, this error was corrected by associating the declaration of the array in question with the initiation of the connection with the computer and the microcontroller rather than just associating it with the initiation of the user's interface program.

When analyzing the case of the third test at a pressure of 10 bar, figure 4.43 and 4.52, many inconsistencies were detected. The most obvious are the much higher top speed than those ob-

tained in the other tests, and the fact that it detects more transitions than in the other tests as well. This second inconsistency should explain the first. It makes no sense that more transitions were detected in this test than in the others, as the course of the SHPB is always the same, and the pattern size was not changed just for this test. This can be explained by the fact that a new pattern sheet was taped on top of a previous one, which had been damaged by the test that took place before. Combining some rotation of the striker bar and the mismatch between patterns, the sensor may have analyzed twice as many transitions, and hence calculated a much higher speed, since the defined distance between transitions was not the analyzed one.

Even though this third test at 10 bar was a test full of errors, and that it does not resemble reality by comparing it with other DIC results, the absence of a direct reference does not allow us to undoubtedly discard this test. What can be learned from this test is that the sensor system can analyze even smaller transitions than the last pattern used. It is also noticed that for the system to be more capable, it would need a faster comparator. This is noticed by the fact that the digital signal decreases in the last part of the movement before impact. This demonstrates that the comparator integrated in the module couldn't keep up with such high speeds, and hence causes the digital signal to be impaired.

Chapter 5

Conclusion

5.1 Conclusions

In this work, it was possible to develop a speed sensing system, that was incorporated in a SHPB to measure its striker bar speed. For this, a TCRT5000 reflective sensor module was combined with an Arduino Mega2560 board, which was connected to a computer. The sensor detected transitions between white and black bands of a printed pattern, which were timed by the microcontroller board, using one of its Input Capture Units. Then, registered times were sent to a computer where the speed was calculated, since band length was known, and it was shown in a graphical interface. The adapted and developed firmware and software codes worked as intended, providing fast communication between the components, and executing efficient data acquisition and calculation.

Validation tests were done to prove that the system worked as intended. These tests were done with two types of patterns, the first one with 40 mm bands and the second one with 5 mm bands. A Photron high frame rate camera and a Rigol oscilloscope were used to validate and troubleshoot the results from the system. It was possible to conclude that, while the first pattern did not provide the necessary spacial resolution, the second pattern was able to more closely approximate the expected results. Nonetheless, since the pattern has discrete transitions and much lower resolution than the high frame rate camera, it was already expected that results would not be equal, but it was possible to get similar values. It is worth noting that the defined requirements were accomplished, since it complies with the limitations of the available stroke, and since it manages to have a lower speed resolution than it was looked for, since it was defined as speed resolution of 0.1 m/s, that is, 100 mm/s, and it was obtained a speed resolution of 0.1 mm/s.

In conclusion, the developed system provided satisfactory results, it was very compact, and it can be easily replicable. This way, it was possible to engineer a system for the intended purpose, achieving the objectives defined in the beginning of this work.

5.2 Future work

Nonetheless, this system can be improved. A critical aspect that would need to improve would be the light barrier, replacing the cardboard version. This barrier would benefit from a box structure, but it would have to be carefully thought, so it wouldn't touch the bar, and, this way, it wouldn't interfere with the tests carried out. The sensor module support could be incorporated into it too, creating a unique piece for this small structure. Also, the Arduino support could be upgraded to include a dedicated space for a printed circuit board.

For this system to work at higher speeds, and with smaller bands, some aspects would need to be improved.

Firstly, the Arduino firmware could be revisited to make it more efficient timing wise, and to create a feature that would gather all transitions data, before it is sent to the computer to be processed. This way, smaller bands and higher speeds could be used more reliably.

Pattern wise, it could be interesting to improve it using a more reflective surface for the white bands, so that the radiation emitted by the sensor could be reflected more into itself, and using a less reflective surface for the black bands, to further reduce the amount of reflected radiation. This way, one would increase the difference between minimum and maximum analog values, to ensure correct transition detection, and avoid inconclusive analog values.

To study if the system can be more accurate, when compared to the high speed camera data, it would be interesting to analyze how the pattern transitions relate to the speed curves from DIC data, that is, where it would be the optimum position for the pattern to be applied on the bar, and to which DIC values the sensor speed values correspond.

To analyze smaller areas, it would be suggested to acquire a more powerful sensor and a faster comparator, to ensure fast transitions of the digital signal.

References

- [1] Trunal Bhujangrao, Catherine Froustey, Edurne Iriondo, Fernando Veiga, Philippe Darnis, and Franck Girot Mata. Review of Intermediate Strain Rate Testing Devices. *Metals*, 10(7):894, July 2020. Number: 7 Publisher: Multidisciplinary Digital Publishing Institute.
- [2] S. Sharma, V.M. Chavan, R.G. Agrawal, R.J. Patel, R. Kapoor, and J.K. Chakravartty. Split-Hopkinson Pressure Bar: an experimental technique for high strain rate tests. Technical report, India, 2011. BARC–2011/E/013 INIS Reference Number: 42105051.
- [3] Weinong Chen and Bo Song. *Split Hopkinson (Kolsky) Bar: Design, Testing and Applications*. January 2011.
- [4] Subhajit Sen, Biswanath Banerjee, and Amit Shaw. Taylor impact test revisited: Determination of plasticity parameters for metals at high strain rate. *International Journal of Solids and Structures*, 193-194:357–374, June 2020.
- [5] Egor S. Rodionov, Victor G. Lupanov, Natalya A. Gracheva, Polina N. Mayer, and Alexander E. Mayer. Taylor Impact Tests with Copper Cylinders: Experiments, Microstructural Analysis and 3D SPH Modeling with Dislocation Plasticity and MD-Informed Artificial Neural Network as Equation of State. *Metals*, 12(2):264, February 2022. Number: 2 Publisher: Multidisciplinary Digital Publishing Institute.
- [6] Hengfu Li, Zhenyu Yu, Peng Rong, Yi Wu, Xulong Hui, Fengguo Zhang, Zhe Chen, and Haowei Wang. High Strain Rate Response of In-Situ TiB2/7055 Composite by Taylor Impact. *Materials*, 14(2):258, January 2021. Number: 2 Publisher: Multidisciplinary Digital Publishing Institute.
- [7] Chong Gao and Takeshi Iwamoto. Instrumented Taylor impact test for measuring stress-strain curve through single trial. *International Journal of Impact Engineering*, 157:103980, November 2021.
- [8] High Strain Rate Infrastructure, December 2017. [<https://uwaterloo.ca/waterloo-forming-crash-lab/research-facilities/high-strain-rate-infrastructure>, Accessed: 3.3.2023].
- [9] A. A. Cenna, N. W. Page, E. Kisi, and M. G. Jones. Single particle impact tests using gas gun and analysis of high strain-rate impact events in ductile materials. *Wear*, 271(9):1497–1503, July 2011.
- [10] T. Jiao, R. J. Clifton, S. E. Grunschel, Mark Elert, Michael D. Furnish, Ricky Chau, Neil Holmes, and Jeffrey Nguyen. PRESSURE-SENSITIVITY AND TENSILE STRENGTH OF AN ELASTOMER AT HIGH STRAIN RATES. pages 707–710, Waikoloa (Hawaii), 2008.

- [11] Shmuel Samuha, Eyal Kahana, Oren Sadot, and Roni Z. Shneck. Improved Formability of Mg-AZ80 Alloy under a High Strain Rate in Expanding-Ring Experiments. *Materials*, 11(2):329, February 2018. Number: 2 Publisher: Multidisciplinary Digital Publishing Institute.
- [12] W. H. Gourdin, S. L. Weinland, and R. M. Boling. Development of the electromagnetically launched expanding ring as a high-strain-rate test technique. *Review of Scientific Instruments*, 60(3):427–432, March 1989.
- [13] Sahand Pourhassan Shamchi. High Strain Rate Behavior of Aerospace Materials. May 2021. Accepted: 2022-09-09T03:48:39Z.
- [14] Bazle A Gama, Sergey L Lopatnikov, and John W Gillespie, Jr. Hopkinson bar experimental technique: A critical review. *Applied Mechanics Reviews*, 57(4):223–250, October 2004.
- [15] Stephen M. Walley. The Origins of the Hopkinson Bar Technique. In Ramzi Othman, editor, *The Kolsky-Hopkinson Bar Machine: Selected Topics*, pages 1–25. Springer International Publishing, Cham, 2018.
- [16] Akihiro UENISHI, Hiroshi YOSHIDA, Yukihsa Kuriyama, and Manabu Takahashi. Material Characterization at High Strain Rates for Optimizing Car Body Structures for Crash Events. *Nippon Steel Technical Report*, 88, July 2003.
- [17] António Francisco de Galhano Tenreiro, António Manuel Ferreira Mendes Lopes, Carlos Manuel de Sousa Moreira da Silva, and Lucas Filipe Martins da Silva. Development of a Split Hopkinson Bar type machine to measure the impact resistance of bonded joints, 2018.
- [18] Jianming LIU, Zhiyong BAO, Yang LIU, Zhenchun WANG, and Fucui LIU. A Measuring Method About the Bullet Velocity in Electromagnetic Rail Gun. *Sensors and Transducers*, 165:29–34, February 2014.
- [19] Zhanwei Liu, Ximin Chen, Xintao Lv, and Huimin Xie. A mini desktop impact test system using multistage electromagnetic launch. *Measurement*, 49:68–76, March 2014.
- [20] Matthew Schwartz. Lecture 21: The Doppler effect.
- [21] Dipl.-Ing (FH) Christian Wolff. Radartutorial. Publisher: Dipl.-Ing. (FH) Christian Wolff.
- [22] Jonathan D. Kaunitz. The Doppler Effect: A Century from Red Shift to Red Spot. *Digestive diseases and sciences*, 61(2):340–341, February 2016.
- [23] Joseph DuCarme. Developing effective proximity detection systems for underground coal mines. In *Advances in Productive, Safe, and Responsible Coal Mining*, pages 101–119. Elsevier, 2019.
- [24] Lewis Lea and Andrew Jardine. Application of photon Doppler velocimetry to direct impact Hopkinson pressure bars.
- [25] Abdessamad Klilou, Assia Arsalane, Mohamed Najoui, and Said Belkouch. An efficient and scalable parallel mapping of pulse-Doppler radar signal processing chain on a multi-core DSP. *Microprocessors and Microsystems*, 85:104302, September 2021.
- [26] Michael Parker. Chapter 19 - Pulse Doppler Radar. In Michael Parker, editor, *Digital Signal Processing 101 (Second Edition)*, pages 241–251. Newnes, January 2017.

- [27] Alexander Vargas, Robin Álvarez, Pablo Lupera, and Felipe Grijalva. SDR-based Speed Measurement with Continuous-Wave Doppler Radar. In *2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, pages 401–406, September 2021.
- [28] Davi V. Q. Rodrigues and Changzhi Li. A Review on Low-Cost Microwave Doppler Radar Systems for Structural Health Monitoring. *Sensors*, 21(8):2612, January 2021. Number: 8 Publisher: Multidisciplinary Digital Publishing Institute.
- [29] Kookjin Sung and Manoranjan Majji. Spacecraft Proximity Navigation using the iVisNav Sensor System. February 2020.
- [30] Lukasz Farbaniec, Yuan Xu, Junyi Zhou, Sophoclis Patsias, Duncan Macdougall, Julian Reed, Nik Petrinic, Clive Siviour, Antonio Pellegrino, and Daniel Eakins. Application of the Photon Doppler Velocimetry (PDV) technique in tension-torsion Hopkinson bar experiments. *EPJ Web of Conferences*, 250:01025, January 2021.
- [31] D. T. Casem and M. B. Zellner. Kolsky Bar Wave Separation Using a Photon Doppler Velocimeter. *Experimental Mechanics*, 53(8):1467–1473, October 2013.
- [32] Shamal L. Chinke, Solomon Berhe, and Prashant S. Alegaonkar. High Speed Projectile Sensor: Design, Development and System Engineering. *IEEE Sensors Journal*, 21(23):27062–27068, December 2021.
- [33] Lewis Lea and Andrew Jardine. Two-wave photon Doppler velocimetry measurements in direct impact Hopkinson pressure bar experiments. *EPJ Web of Conferences*, 94:01063, January 2015.
- [34] William B. Ribbens. Chapter 6 - Sensors and Actuators. In William B. Ribbens, editor, *Understanding Automotive Electronics (Seventh Edition)*, pages 233–307. Butterworth-Heinemann, Oxford, January 2013.
- [35] Kevin M. Lynch, Nicholas Marchuk, and Matthew L. Elwin. Chapter 21 - Sensors. In Kevin M. Lynch, Nicholas Marchuk, and Matthew L. Elwin, editors, *Embedded Computing and Mechatronics with the PIC32*, pages 317–340. Newnes, Oxford, January 2016.
- [36] Tarun Agarwal. Different Types of Optical Sensors and Applications, March 2017. [<https://www.rigol-uk.co.uk/pdf/Rigol-MSO5000-User-Guide.pdf>, Accessed: 4.03.2023].
- [37] Diffused, through-beam and retroreflective sensors - what's the difference? - element14 Community, November 2019. [<https://community.element14.com/products/manufacturers/bulgin/b/blog/posts/diffused-through-beam-and-retroreflective-sensors—what-s-the-difference>, Accessed: 4.03.2023].
- [38] High Strain Rate Testing Speed Sensor | REL. [<https://www.relinc.com/speed-sensor>, Accessed: 3.3.2023].
- [39] Bing Pan, Kemao Qian, Huimin Xie, and Anand Asundi. Two-dimensional digital image correlation for in-plane displacement and strain measurement: a review. *Measurement Science and Technology*, 20(6):062001, April 2009.
- [40] Hubert Schreier, Jean-José Orteu, and Michael A. Sutton. *Image Correlation for Shape, Motion and Deformation Measurements: Basic Concepts, Theory and Applications*. Springer US, Boston, MA, 2009.

- [41] TCRT5000, TCRT5000L Optical Sensors | Vishay. [<https://www.vishay.com/en/product/83760/>, Accessed: 5.3.2023].
- [42] TCRT5000 Line Following/ Optical Sensor Module | Alphasatronic. [<https://alphatronic.lk/product/tcrt5000-line-following-optical-sensor-module/>, Accessed: 5.3.2023].
- [43] Texas Instruments . TDC7200 data sheet, product information and support.
- [44] Atmel. Atmel ATmega640/V-1280/V-1281/V-2560/V-2561/V 8-bit Atmel Microcontroller with 16/32/64KB In-System Programmable Flash.
- [45] High-Speed Counter/Timer Devices with Digital I/O. [<https://www.mccdaq.com/usb-data-acquisition/USB-CTR-Series.aspx>, Accessed: 15.3.2023].
- [46] Bernardo Santos Moreira. Development, assembly, testing and validation of a Split-Hopkinson pressure bar testing equipment, September 2022.
- [47] What is a Microcontroller and How Does it Work? [<https://www.techtarget.com/iotagenda/definition/microcontroller>, Accessed: 5.3.2023].
- [48] Everything You Need To Know About Microcontrollers | RS. [<https://ie.rs-online.com/web/generalDisplay.html?id=ideas-and-advice/microcontrollers-guide>, Accessed: 15.3.2023].
- [49] What is Arduino? [<https://www.arduino.cc/en/Guide/Introduction>, Accessed: 23.2.2023].
- [50] Sketch build process - Arduino CLI. [<https://arduino.github.io/arduino-cli/0.33/sketch-build-process/>, Accessed: 23.2.2023].
- [51] ATmega2560-Arduino Pin Mapping | Arduino Documentation. [<https://docs.arduino.cc/hacking/hardware/PinMapping2560>, Accessed: 23.2.2023].
- [52] ARD Mega2560R3 | Joy-IT. [<https://joy-it.net/en/products/ARD-Mega2560R3>, Accessed: 23.2.2023].
- [53] Gammon Forum : Electronics : Microprocessors : Timers and counters. [<http://www.gammon.com.au/timers>, Accessed: 3.3.2023].
- [54] Atmel. Atmel ATmega328P 8-bit AVR Microcontroller with 32K Bytes In-System Programmable Flash.
- [55] What is a Pulse Width Modulation (PWM) signal?, February 2022. [<https://www.circuitbread.com/ee-faq/what-is-a-pwm-signal>, Accessed: 23.3.2023].
- [56] Janet Heath. PWM: Pulse Width Modulation: What is it and how does it work?, April 2017. [<https://www.analogictips.com/pulse-width-modulation-pwm/>, Accessed: 23.3.2023].
- [57] Rob Toulson and Tim Wilmshurst. Chapter 4 - Analog Output. In Rob Toulson and Tim Wilmshurst, editors, *Fast and Effective Embedded Systems Design*, pages 57–75. Newnes, Oxford, January 2012.
- [58] Basics of PWM (Pulse Width Modulation) | Arduino Documentation. [<https://docs.arduino.cc/learn/microcontrollers/analog-output>, Accessed: 25.3.2023].

- [59] Simulate pulse width modulation (PWM) output from hardware - Simulink.
- [60] Sustainablelab. Arduino Diecimila and the Atmega168: Changing PWM Frequency on the Arduino Diecimila and the Atmega168, November 2008. [<http://usethearduino.blogspot.com/2008/11/changing-pwm-frequency-on-arduino.html>, Accessed: 20.3.2023].
- [61] LTD. RIGOL TECHNOLOGIES CO. Rigol MSO5000 Series Digital Oscilloscope User Guide, May 2020. [<https://www.rigol-uk.co.uk/pdf/Rigol-MSO5000-User-Guide.pdf>, Accessed: 15.06.2023].

Appendix A

Developed codes

A.1 Final Arduino code

```
1
2 #include "Arduino.h"
3
4 volatile boolean first;
5 volatile boolean triggered;
6 volatile unsigned long overflowCount;
7 volatile unsigned long startTime;
8 volatile unsigned long finishTime;
9 volatile unsigned long elapsedTime;
10
11 const int pinIRd = 49;
12 const int pinIRa = A0;
13
14
15 // timer overflows (every 65536 counts)
16 ISR(TIMER4_OVF_vect)
17 {
18     overflowCount++;
19 } // end of TIMER4_OVF_vect
20
21
22
23 ISR(TIMER4_CAPT_vect)
24 {
25     // grab counter value before it changes any more
26     unsigned int timer4CounterValue;
27     timer4CounterValue = ICR4; // ver datasheet, icr4 corresponde ao icp4, pin 49
28     unsigned long overflowCopy = overflowCount;
29
30     // if just missed an overflow
31     if ((TIFR4 & bit(TOV4)) && timer4CounterValue < 0x7FFF)
32         overflowCopy++;
```

```
33
34
35  if (first)
36  {
37      startTime = (overflowCopy << 16) + timer4CounterValue;
38      first = false;
39      return;
40  }
41
42  finishTime = (overflowCopy << 16) + timer4CounterValue;
43  elapsedTime= finishTime-startTime;
44  startTime = timer4CounterValue;
45  overflowCount =0;
46  overflowCopy = 0;
47  triggered = true;
48
49 } // end of TIMER4_CAPT_vect
50
51
52
53 void prepareForInterrupts()
54 {
55     noInterrupts(); // protected code
56     first = true;
57     triggered = false; // re-arm for next time
58     // reset Timer 1
59     TCCR4A = 0;
60     TCCR4B = 0;
61
62     TIFR4 = bit(ICF4) | bit(TOV4); // clear flags so we don't get a bogus interrupt
63     TCNT4 = 0; // Counter to zero
64     overflowCount = 0; // Therefore no overflows yet
65
66     // Timer 4 - counts clock pulses
67     TIMSK4 = bit(TOIE4) | bit(ICIE4); // interrupt on Timer 4 overflow and input
        capture
68     // start Timer 4, no prescaler
69     TCCR4B = bit(CS40) | bit(ICES4); // plus Input Capture Edge Select (rising on D49
        )
70
71     interrupts();
72 } // end of prepareForInterrupts
73
74 void setup()
75 {
76     Serial.begin(115200);
77
78     pinMode(pinIRd, INPUT);
79
```

```

80
81  int myEraser = 7;    // this is 111 in binary and is used as an eraser
82  TCCR2B &= ~myEraser; // this operation (AND plus NOT), set the three bits in
    TCCR2B to 0
83
84  int myPrescaler = 6; // this could be a number in [1 , 6]. In this case, 3
    corresponds in binary to 011.
85  TCCR2B |= myPrescaler; // this operation (OR), replaces the last three bits in
    TCCR2B with our new value 011
86
87  const int thisPin = 9;
88  pinMode(thisPin, OUTPUT);
89  analogWrite(thisPin, 255 * .5);
90
91  prepareForInterrupts();
92 } // end of setup
93
94 void loop()
95 {
96     //interrupts();
97
98     if (!triggered)
99         return;
100
101     triggered=false;
102
103     Serial.println((elapsedTime * 62.5 / 1000), 6); // each tick is 62.5 ns at 16
        MHz
104
105 }

```

A.2 Windows Forms code

```

1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10 using System.IO.Ports;
11 using System.Threading;
12 using System.Drawing.Printing;
13 using System.Windows.Forms.DataVisualization.Charting;
14 using System.Globalization;
15 using System.IO;

```

```
16 using CsvHelper;
17 using System.Windows.Forms.VisualStyles;
18 using MathNet.Numerics.Statistics;
19
20 namespace tese_interface
21 {
22     public partial class Form1 : Form
23     {
24         public Form1()
25         {
26             InitializeComponent();
27             FecthAvailablePorts();
28         }
29
30         private SerialPort SerPort;    //serial port
31         private string ReceivedData;    //received data from serial port
32
33         void FecthAvailablePorts()
34         {
35             String[] ports = SerialPort.GetPortNames(); //get available com ports
36             ComboBoxPort.Items.AddRange(ports);
37         }
38
39         private void Form1_Load(object sender, EventArgs e)
40         {
41             ButtonConnect.Enabled = true;
42             ButtonDisconnect.Enabled = false;
43             ButtonCsv.Enabled = true;
44             LabelDisconnected.Text = "Disconnected";
45             LabelDisconnected.ForeColor = Color.Red;
46
47             ComboBoxBaudRate.Text = "115200";
48
49             chart1.ChartAreas[0].CursorX.IsUserEnabled = true;
50             chart1.ChartAreas[0].CursorX.IsUserSelectionEnabled = true;
51             chart1.ChartAreas[0].AxisX.ScaleView.Zoomable = true;
52             chart1.ChartAreas[0].AxisX.ScrollBar.IsPositionedInside = true;
53             chart1.ChartAreas[0].CursorX.Interval = 0.01;
54
55             chart1.ChartAreas[0].CursorY.IsUserEnabled = true;
56             chart1.ChartAreas[0].CursorY.IsUserSelectionEnabled = true;
57             chart1.ChartAreas[0].AxisY.ScaleView.Zoomable = true;
58             chart1.ChartAreas[0].AxisY.ScrollBar.IsPositionedInside = true;
59         }
60
61         private void ButtonConnect_Click(object sender, EventArgs e)
62         {
63             SerPort = new SerialPort();
64             SerPort.BaudRate = Convert.ToInt32(ComboBoxBaudRate.Text);
```

```
65     SerPort.PortName = ComboBoxPort.Text;
66     SerPort.Parity = Parity.None;
67     SerPort.DataBits = 8;
68     SerPort.StopBits = StopBits.One;
69     SerPort.ReadBufferSize = 20000000;
70     SerPort.DataReceived += SerPort_DataReceived;
71     currentIndex = 0;
72
73     textBox1.Clear();
74     textBox2.Clear();
75     textBox3.Clear();
76     textBox4.Clear();
77     ReceivedDataBox.Clear();
78
79     chart1.Series[0].Points.Clear();
80
81     string[] valuesString = new string[2000];
82     int currentIndex = 0;
83     double[] TimeList = new double[2000];
84     double[] timeArray = new double[2000];
85
86     try
87     {
88         SerPort.Open(); //We open the port
89
90         ButtonConnect.Enabled = false;
91         ButtonDisconnect.Enabled = true;
92         ButtonCsv.Enabled = false;
93         LabelDisconnected.Text = "Connected";
94         LabelDisconnected.ForeColor = Color.Green;
95     }
96     catch (Exception ex)
97     {
98         MessageBox.Show(ex.Message, "Error!");
99     }
100 }
101
102 private void ButtonDisconnect_Click(object sender, EventArgs e)
103 {
104     if (SerPort.IsOpen)
105     {
106         try
107         {
108             SerPort.Close(); //We open the port
109
110             ButtonConnect.Enabled = true;
111             ButtonDisconnect.Enabled = false;
112             ButtonCsv.Enabled = true;
113             LabelDisconnected.Text = "Disconnected";
```



```

114         LabelDisconnected.ForeColor = Color.Red;
115     }
116     catch (Exception ex)
117     {
118         MessageBox.Show(ex.Message, "Error!");
119     }
120 }
121 }
122
123 private void Form1_FormClosing(object sender, FormClosingEventArgs e)
124 {
125     if (SerPort.IsOpen)
126     {
127         try
128         {
129             SerPort.Close(); //close the port on from closing
130         }
131         catch (Exception ex)
132         {
133             MessageBox.Show(ex.Message, "Error!");
134         }
135     }
136 }
137
138 string[] valuesString = new string[2000];
139 int currentIndex = 0;
140 double[] TimeList = new double[2000];
141
142 private void SerPort_DataReceived(object sender,
SerialDataReceivedEventArgs e)
143 {
144     if (SerPort.IsOpen)
145     {
146         ReceivedData = SerPort.ReadLine(); //We read the serial port
147         if (currentIndex > valuesString.Length - 1) currentIndex = 0;
148         valuesString[currentIndex] = ReceivedData;
149         TimeList[currentIndex] = (10 * 1000)/Convert.ToDouble(ReceivedData
, CultureInfo.InvariantCulture)/1000; // /
150
151         this.Invoke(new Action(()=> { ProcessingData(TimeList[currentIndex
], currentIndex, TimeList); })); //execute the delegate (ProcessingData)
152         currentIndex++;
153     }
154 }
155
156 double[] timeArray = new double[2000];
157
158 private void ProcessingData(double timing, int index, double[] timevector)
159 {

```

```
160 ReceivedDataBox.AppendText (timing.ToString() + Environment.NewLine);
161 double time;
162
163 try
164 {
165     if (timing > 200) time = 0;
166     else time =timing;
167
168     chart1.Series[0].Points.AddXY(currentIndex, time);
169 }
170
171 catch (Exception ex)
172 {
173     MessageBox.Show(ex.Message, "Error!");
174 }
175
176 timeArray = timevector.Where(T => T != 0).ToArray();
177 timeArray = timeArray.Where(T => T != 0).ToArray();
178
179 double meanTime = Statistics.Mean(timeArray);
180 double medianTime = Statistics.Median(timeArray);
181 double maxTime = Statistics.Maximum(timeArray);
182 double minTime = Statistics.Minimum(timeArray);
183
184 textBox1.Text = Convert.ToString(meanTime);
185 textBox2.Text = Convert.ToString(medianTime);
186 textBox3.Text = Convert.ToString(maxTime);
187 textBox4.Text = Convert.ToString(minTime);
188 }
189
190 private void ButtonCsv_Click(object sender, EventArgs e)
191 {
192     string filepath;
193     string filename = "test";
194     string currentFile = filename;
195     int j = 0;
196
197
198     while (File.Exists($"test\\{currentFile}.csv"))
199     {
200         j++;
201         currentFile = filename + j.ToString();
202     }
203
204     filepath = $"test\\{currentFile}.csv";
205
206     var writer = new StreamWriter(filepath);
207     var csvwriter = new CsvWriter(writer, CultureInfo.InvariantCulture);
208
```

```
209         csvwriter.WriteRecords(TimeList);
210
211         csvwriter.Dispose();
212         writer.Dispose();
213     }
214 }
215 }
```