Czech Technical University
Faculty of Transportation Sciences

Department of Mechanics and Materials

Study Programme: Technology in Transportation and Telecommunications

Field of Study / Specialization: Air Transport

Automated Image Processing Methods for Impact Dynamics Experiments

BACHELOR'S THESIS

Author:	Bc. Jan Stoklasa
Supervisors:	Ing. Tomáš Fíla, PhD.; Ing. Jan Šleichrt
Year:	2023

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta dopravní
děkan
Konviktská 20, 110 00 Praha 1

K618 .... **Ústav mechaniky a materiálů**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení studenta (včetně titulů):

Jan Stoklasa

Studijní program (obor/specializace) studenta:

bakalářský – LED – Letecká doprava

Název tématu (česky): **Automatizované metody zpracování obrazu pro rázové experimenty**

Název tématu (anglicky): Automated image processing methods for impact dynamics experiments

Zásady pro vypracování

Při zpracování bakalářské práce se řiďte následujícími pokyny:

- Cílem bakalářské práce je vytvořit nástroj pro automatizované zpracování obrazu nasnímaného během rázového experimentu pomocí rychloběžné kamery. Vytvořte nástroj tak, aby umožňoval automatizovaně sledovat vybrané prvky v obrazu a na základě tohoto sledování následně umožnit vyhodnocení vybrané veličiny. Vzhledem k proměnným parametrům měření během rázových experimentů bude za splnění bakalářské práce považováno i ověření spolehlivosti metod automatizovaného zpracování obrazu na vybraných datech.
- Typickou aplikací pro uvažovaný softwarový nástroj je automatizovaná detekce pseudo-náhodného obrazce připevněného na dopadajícím projektilu a měření jeho rychlosti na základě sledování jeho polohy v obrazu a v čase. V současné době je za tímto účelem využívána metoda digitální korelace obrazu, přičemž zpracování dat vyžaduje řadu manuálních vstupů a je časově náročné.
- Další typickou aplikací je identifikace a oprava chyb obsažených v obraze způsobených nedokonalou geometrickou kalibrací celého systému, optickým zkreslením jednotllivých optických komponent, nedostatečnou hloubkou ostrosti či výskytem úlomků během rázů.

Rozsah grafických prací: není stanoven

Rozsah průvodní zprávy: minimálně 35 stran textu (včetně obrázků, grafů a tabulek, které jsou součástí průvodní zprávy)

Seznam odborné literatury: Himanshu Singh, Practical Machine Learning and Image Processing, APress, 2019

Gonzalez, Rafael C., Digital Image Processing, Global Edition, Pearson Education Limited, 2017

Blanchet Gerard, Digital Signal and Image Processing Using Matlab, Volume 1: Fundamentals, Wiley, 2014

Vedoucí bakalářské práce: **Ing. Tomáš Fíla, Ph.D.**

Ing. Jan Šleichrt

Datum zadání bakalářské práce: **7. října 2022**

(datum prvního zadání této práce, které musí být nejpozději 10 měsíců před datem prvního předpokládaného odevzdání této práce vyplývajícího ze standardní doby studia)

Datum odevzdání bakalářské práce: **7. srpna 2023**

a) datum prvního předpokládaného odevzdání práce vyplývající ze standardní doby studia a z doporučeného časového plánu studia

b) v případě odkladu odevzdání práce následující datum odevzdání práce vyplývající z doporučeného časového plánu studia

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta dopravní

L.S.

prof. Ing. Ondřej Jiroušek, Ph.D.
vedoucí
Ústavu mechaniky a materiálů

prof. Ing. Ondřej Přibyl, Ph.D.
děkan fakulty

Potvrzuji převzetí zadání bakalářské práce.

Jan Stoklasa
jméno a podpis studenta

V Praze dne .... 7. října 2022

Declaration

I have no relevant reason against using this work in the sense of §60 of Act No. 121/2000 Coll. on the Copyright and Rights Related to Copyright and on the Amendment to Certain Acts (the Copyright Act).

I declare I have accomplished my final thesis by myself and I have named all the sources used in accordance with the Guideline on the ethical preparation of university final theses.

In Prague, August 7, 2023 ....

Bc. Jan Stoklasa

Acknowledgements

I would like to express my sincere gratitude to my supervisors, Ing. Tomáš Fíla, Ph.D., and Ing. Jan Šleichrt, for their invaluable and constructive discussions about my thesis. Furthermore, I extend my appreciation to doc. Ing. Daniel Kytýř, Ph.D., for introducing me to the essentials of experimental mechanics. Additionally, I am grateful to all the members of the department for sustaining a creative atmosphere.

I would also like to extend my heartfelt thanks to all my family members for their support and motivation throughout my academic journey.

The bachelor thesis was supported by the Czech Science Foundation (project Junior Star no. 22-18033M).

Bc. Jan Stoklasa

Title: **Automated Image Processing Methods for Impact Dynamics Experiments**

Author: Bc. Jan Stoklasa

Study programme: Technology in Transportation and Telecommunications

Study field: Air Transport

Degree: Bachelor's Thesis

Year: 2023

Supervisors: Ing. Tomáš Fíla, PhD.; Ing. Jan Šleichrt
Department of Mechanics and Materials, Faculty of Transportation Sciences, Czech Technical University

Abstract: This thesis introduces new software for the detection and recognition of randomly generated patterns, which are utilized for the processing of image data during impact dynamics experiments. The tool enables the identification of the pattern's position and its further utilization in tracking its motion during the experiment, as well as its automatic integration into subsequent image processing methods. The automation of individual processes significantly minimizes the manual workload in the evaluation process. Employing multiple independent image processing techniques provides possibilities to review accuracy and reliability of the results and also provides result redundancy. The method has been evaluated on sample datasets, representing not only potential usage of the method, but also interesting materials with potential applications in various fields, including aviation. The sample datasets demonstrate that under suitable conditions, the method's outcomes are fully comparable with alternative image processing approaches, such as digital image correlation, as well as non-image data obtained from strain gauges. However, in certain cases, particularly when debris is present in the pattern region, the method exhibits systematic failure, yielding results that vary in quality compared to image correlation.

Keywords: Pattern detection, Image processing, Digital Image Correlation

Název: **Automatizované metody zpracování obrazu pro rázové experimenty**

Autor: Bc. Jan Stoklasa

Studijní program: Technika a technologie v dopravě a spojích

Obor: Letecká doprava

Druh práce: Bakalářská práce

Rok vydání: 2023

Vedoucí práce: Ing. Tomáš Fíla, PhD.; Ing. Jan Šleichrt
Ústav mechaniky a materiálů, Fakulta dopravní, České vysoké učení technické v Praze

Abstrakt: Tato práce představuje nový software pro detekci a rozpoznání náhodně generovaného obrazce, který je využíván pro zpracování obrazových dat při rázových experimentech. Nástroj umožňuje zjištění polohy tohoto obrazce a její další využití pro zjištění pohybu během experimentu, a také jako vstup do dalších metod zpracování obrazu. Automatizace jednotlivých procesů umožňuje omezení opakujících se manuálních činností při vyhodnocování experimentálních dat. Využití více nezávislých metod zpracování obrazu naskýtá možnost vyhodnocení správnosti získaných dat a poskytuje redundanci výsledků. Metoda byla testována na vzorových setech dat, které reprezentují nejen jednotlivé možnosti metody, ale rovněž zajímavé materiály s potenciálem použití v celé řadě oblastí, včetně letectví. Vzorové sety dat ukazují, že při příhodných podmínkách jsou výsledky této metody plně srovnatelné s alternativní metodou zpracování obrazu, digitální korelací obrazu, a také s neobrazovými daty získanými z tenzometrů. V některých případech, zejména když jsou přítomny trosky vzorku v oblasti s obrazcem, metoda systematicky selhává a v závislosti na charakteru dat metoda poskytuje horší i lepší výsledky než korelace obrazu.

Klíčová slova: Detecke obrazce, Zpracování obrazu, Digitální korelace obrazu

Contents

1. Introduction

Dynamic impacts, the high velocity events, characterized by high-rate loading have a critical impact to various technologically significant applications, spanning a diverse spectrum of fields. Examples of such applications include armor penetration, the crashworthiness of vehicles, high-speed machining, etc. Within the aviation industry, high-rate loading is a crucial consideration for different aircraft components, such as landing gear during landings, airframes in the event of bird strikes or hail strikes (or any crash scenario), and more [1].

When high-rate loading occurs, the mechanical response of materials generally differs from cases involving lower loading rates (rates close to quasi-static loading). The study of this phenomenon relies on experimental capabilities to achieve high-rate loading. However, the knowledge, that the material properties under impact are different from those under static loading, has been known since the 19th century [2]. In this context, J. Hopkinson's work in 1872 demonstrated that metals can withstand significantly higher impulsive loads in simple tension compared to static loading conditions [3]. The commencement of more sophisticated high-rate loading experiments can be traced back to approximately the half of the 20th century [4].

High-speed cameras were introduced to the experimental setup quite early in the development of high-rate loading experiments. However, due to the camera's low resolution, low frame rate, and limited computational power for image processing, the camera's data were not utilized for obtaining experimental data, but only for observation purposes. Recently, with advancements in camera and IT equipment technology, high-speed cameras can now serve as a relevant source of experimental data [5]. Nevertheless, this approach to high-rate loading experiments imposes high requirements for image data quality, which cannot always be met in many cases, such as when there is the presence of sample debris covering a large area of the captured picture.

The conventional approach to processing image data from experiments involves using the digital image correlation method [6]. With recent advancements in information technology, new opportunities have arisen for the integration of machine learning and deep learning techniques into data processing of experimental data. These new approaches can serve as complementary methods to the conventional approach [7], or even provide an alternative technique that yields comparable results to the standard methods [8].

Another trend in experimental mechanics is the development of automatic fault recognition methods during image processing. These methods are essential for automatic image processing techniques as they can recognize instances when the method fails and can potentially correct the results [9].

Additionally, in context of introduction above, in this thesis I focus on the development of an automated pattern detection software. The software will be used during post-processing of data acquired from impact dynamics experiments.

2. Methods

2.1 Digital Image Processing

Image processing encompasses a wide range of techniques and algorithms aimed at improving, manipulating, and analyzing images. These techniques enable us to extract meaningful information, enhance the visual quality, and find patterns in the image. Digital image processing can be classified into several distinct classes of operations, each serving a specific purpose: image enhancement, image restoration, image analysis, and image compression. Image enhancement techniques manipulate images mostly by using heuristic approaches to obtain images from which human viewers can extract more information. These techniques aim to improve the visual quality and perceptual characteristics of the image. Image restoration methods utilize statistical and mathematical models to predict missing or damaged parts of an image caused by degradation. By establishing statistical and mathematical models of the degradation process, these methods aim to restore the image to its original state. Image analysis techniques facilitate the automated extraction of information from images through a systematic processing approach. These techniques encompass various operations that enable the delineation and characterization of image components and features. Examples of image analysis include image segmentation, edge extraction, and texture and motion analysis. Within the scope of this thesis, the image analysis component of image processing, especially image recognize (computer vision), is identified as the most valuable and relevant part [10].

To provide image processing techniques, the MATLAB Image Processing Toolbox (MIPT) was used [11] and some of the functions included in MIPT that were utilized in the thesis will be described in following chapters (2.1.2 to 2.1.5) . The image processing methods discussed in the following sections were implemented using Matlab® software [12], version R2022b. The objective of the process is to achieve pattern detection of a randomly generated pattern physically embedded within the image data. More details were are listed in Computational protocol, Section 2.7.

2.1.1 Logic of the Process

The process of calculating the pattern mask from the gray scale image dataset is schematically shown in Figure 2.1. Firstly, local statistical values are calculated. Different patterns can be localized using the fact that each pattern (or background, etc.) has distinct statistical values in the local area of its presence. The appropriate statistical variable is mostly chosen empirically to best fit the investigated pattern. Secondly, the image of statistical values is binarized into two sets: the pattern (foreground) and the background. In cases where any pattern area encompasses a recognized background, this background is moved to the recognized pattern set. Mostly, the foreground area consists of several non-connected regions. If they cannot be connected via the previous step (Area Closure), *the best* foreground has to be selected.

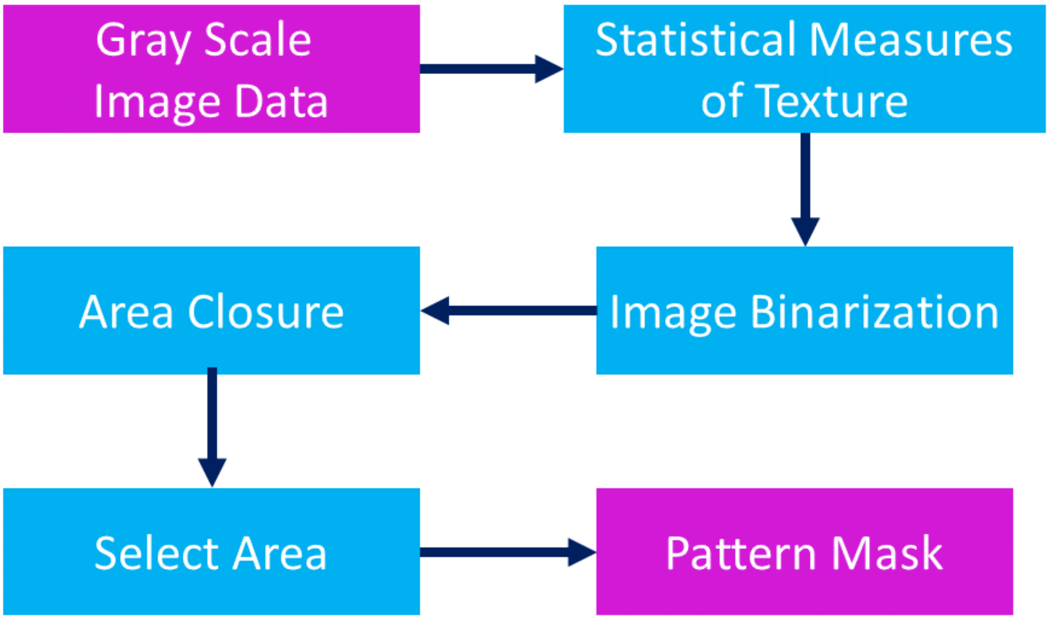


Figure 2.1: The process of mask recognition

After obtaining the pattern mask, the pattern location is included in it. Then, with knowledge of the pattern's location, an area including the pattern can be chosen as the input for the Digital Image Correlation (DIC) method. Generally, the pattern can be located in every captured image of the dataset, and independently, the pattern's movement can be tracked using DIC. From the movement, specifically the displacement, the velocity of the pattern, or respectively the sample, can be easily calculated, providing valuable experimental results. Additionally, other non-image data, such a strain gauge measurements, can be utilized to compare displacement or velocity values. This process is schematically shown in Figure 2.2.

The steps mentioned in the previous paragraphs are detailedly described in the following sections.

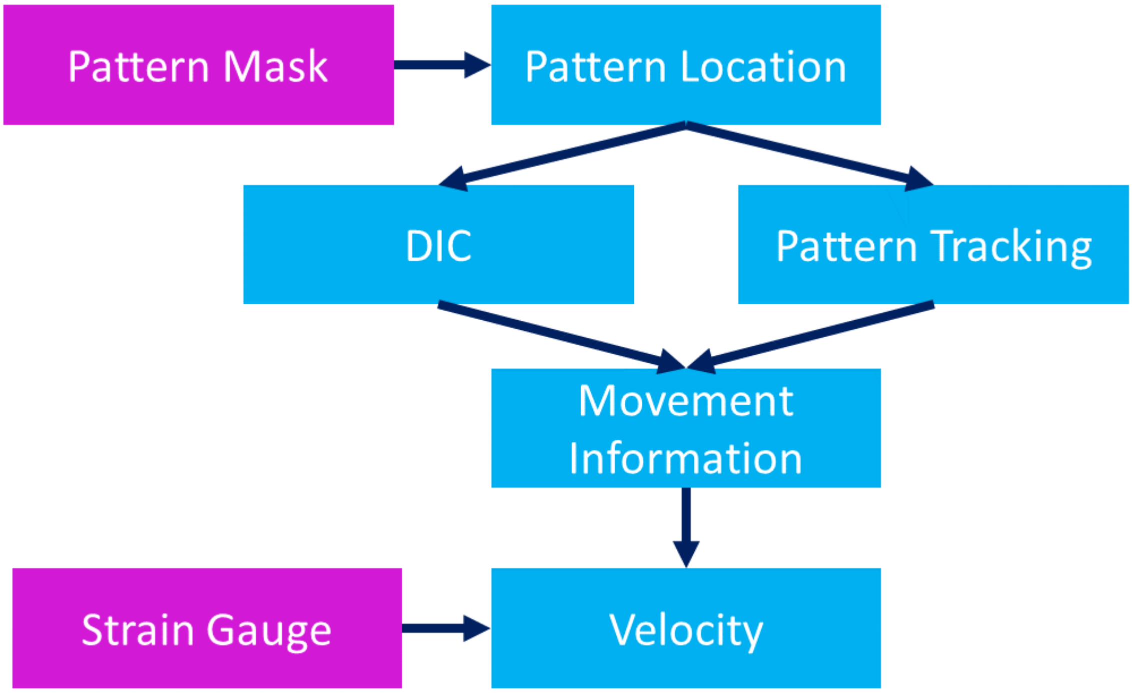


Figure 2.2: The process of image processing

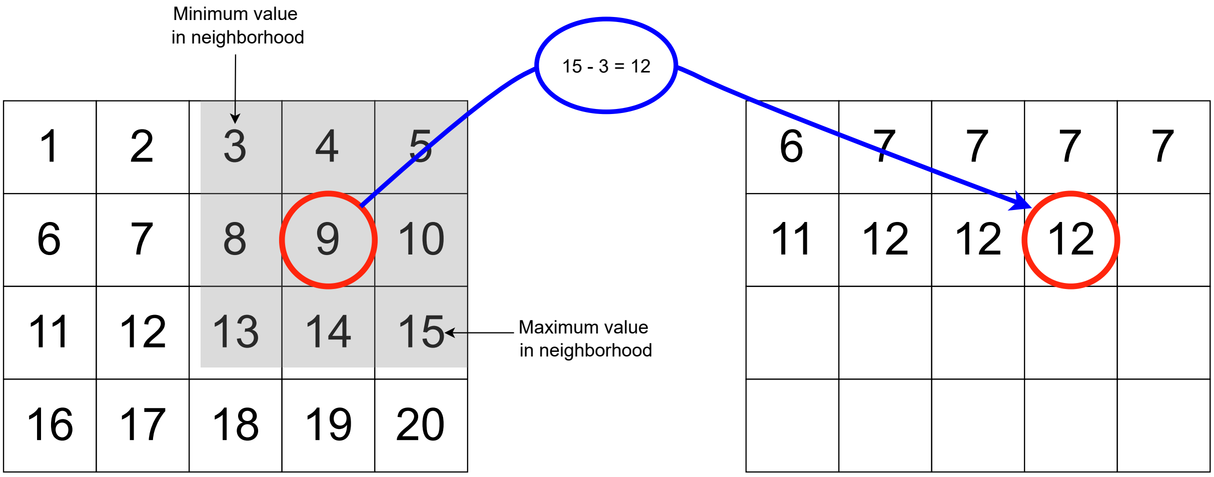
2.1.2 Calculation of Statistical Measures of Texture

When using MIPT (MATLAB Image Processing Toolbox), there are several possibilities used to calculate the local statistical values within an image. MIPT provides the following functions: `rangefilt`, `stdfilt` and `entropyfilt` function.

The `rangefilt` function calculates the local range of the pixel intensities, while the `stdfilt` function calculates the local standard deviation. Additionally, the `entropyfilt` function calculates the local entropy, which represents the statistical measure of randomness in the image [11].

To utilize these functions, the input parameters, apart from the original image, include the specification of the neighborhoods where the local statistical values should be computed. As illustrated in Figure 2.3, an example demonstrates the calculation of a local statistical value within the pixel's neighborhood.

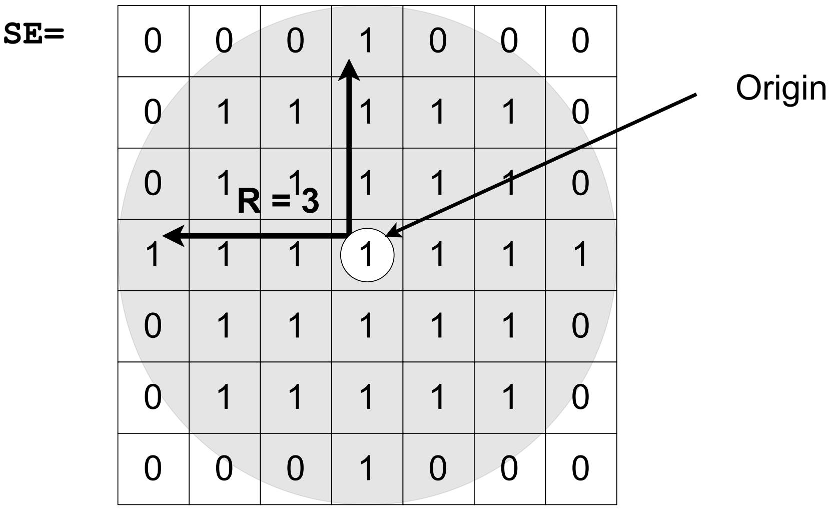
The neighborhood mask can be configured into various shapes utilizing functions such as `strel`. In Figure 2.4, an illustrative example showcases the usage of the `strel` function to generate a circular mask [13].



1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20

6	7	7	7	7
11	12	12	12	

Figure 2.3: Example of the calculation of the local range in the image using the `rangefilt` function with 3x3 neighborhood [11]



0	0	0	1	0	0	0
0	1	1	1	1	1	0
0	1	1	1	1	1	0
1	1	1	1	1	1	1
0	1	1	1	1	1	0
0	1	1	1	1	1	0
0	0	0	1	0	0	0

Figure 2.4: Example of creating the *disk* neighborhood mask [13]

2.1.3 Binarized Image

To define the region of interest (ROI), the binary mask is used. A mask with pixel values of 1 represents image pixels belonging to the ROI, while a mask with pixel values of 0 denotes image pixels that correspond to the background. One of the possibilities in MIPT to generate binary mask of ROI is to use the `imbinarize` function. In this case, the ROI is generated by definition of the threshold for the pixel's intensity. By default, the `imbinarize` function uses Otsu's method which selects the threshold value to minimize the intraclass variance in the ROI and background pixels. In addition, the `imbinarize` function allows for the input of a user-defined threshold value [14].

2.1.3.1 Otsu's Method of Binarization

The method employed, Otsu's method, is a non-parametric and unsupervised approach utilized for the automatic threshold selection in the binarization process. It aims to determine an optimal threshold that maximizes the separation between the two resulting classes based on discriminant criteria. The process is straightforward and will be described in this section [15].

The equations below are based on Otsu's article [15]. Let the image have N pixels at L gray level, so at the i level n_i pixels can be found . The histogram is normalized and regarded as a probability distribution:

$$p_i = \frac{n_i}{N}. \tag{2.1}$$

An obvious corollary of the aforementioned fact is that the sum of p_i over all L levels equals 1. Additionally, it is expected that the probability of occurrence in level i, denoted as p_i, is greater than or equal to zero. When segmenting pixels into two classes, denoted as C_0 and C_1, using a threshold at level k, C_0 represents the pixels at levels ranging from 1 to k, while C_1 represents the pixels at levels ranging from $k+1$ to L. Subsequently, the probability of occurrence in class C_0 is denoted as ω_0, while the probability of occurrence in class C_1 is denoted as ω_1. Furthermore, the mean class values, represented by μ_0 and μ_1, can be defined in a similar manner.

$$\omega_0 = \mathrm{P}(C_0) = \sum_{i=1}^{k} p_i = \omega(k) \tag{2.2}$$

$$\omega_1 = \mathrm{P}(C_1) = \sum_{i=k+1}^{L} p_i = 1 - \omega(k) \tag{2.3}$$

$$\mu_0 = \sum_{i=1}^{k} i \cdot \mathrm{P}(i|C_0) = \sum_{i=1}^{k} i \cdot \frac{p_i}{\omega_0} = \frac{\mu(k)}{\omega(k)} \tag{2.4}$$

$$\mu_1 = \sum_{i=k+1}^{L} i \cdot \mathrm{P}(i|C_1) = \sum_{i=k+1}^{L} i \cdot \frac{p_i}{\omega_1} = \frac{\mu_T - \mu(k)}{1 - \omega(k)} \tag{2.5}$$

where

$$\omega(k) = \sum_{i-1}^{k} p_i \tag{2.6}$$

$$\mu(k) = \sum_{i-1}^{k} i \cdot p_i \tag{2.7}$$

and μ_T is the total mean level $\mu_T = \sum_{i=1}^{L} i \cdot P_i$. Then the class variances σ^2 are given by the following equations.

$$\sigma_0^2 = \sum_{i=1}^{k} (i - \mu_0)^2 \cdot \mathrm{P}(i|C_0) = \sum_{i=1}^{k} (i - \mu_0)^2 \frac{p_i}{\omega_0} \tag{2.8}$$

$$\sigma_1^2 = \sum_{i=k+1}^{L} (i - \mu_1)^2 \cdot \mathrm{P}(i|C_1) = \sum_{i=k+1}^{L} (i - \mu_1)^2 \frac{p_i}{\omega_1} \tag{2.9}$$

To evaluate the *goodness* of the selected threshold, Otsu introduced 3 variables

$$\lambda = \frac{\sigma_B^2}{\sigma_W^2} \qquad \kappa = \frac{\sigma_T^2}{\sigma_W^2} \qquad \eta = \frac{\sigma_B^2}{\sigma_T^2} \tag{2.10}$$

where σ_W^2 is the within-class variance, σ_B^2 is the between-class variance, and σ_T^2 is the total variance of the levels.

$$\begin{aligned} \sigma_W^2 &= \omega_0 \sigma_0^2 + \omega_1 \sigma_1^2 \\ \sigma_B^2 &= \omega_0(\mu_0 - \mu_T)^2 + \omega_1(\mu_1 - \mu_T)^2 \\ &= \omega_0 \omega_1 (\mu_1 - \mu_0)^2 \\ \sigma_T^2 &= \sum_{i=1}^{L} (i - \mu_T)^2 p_i \end{aligned}$$

Due to fact that the terms in Equation 2.10 follow the relationship $\sigma_W^2 + \sigma_B^2 = \sigma_T^2$, the optimal freehold k is the one that maximizes one of the variables and the maximization of η is the simplest measure with respect to k.

2.1.4 Area Closure

The `imclose` function in MIPT performs morphological closing on the gray scale or binary images. It accomplishes this by utilizing a specified structuring element neighborhood, which aims to enhance the object connectivity and preserve the structural information. The primary objective of this function is to fill gaps and holes in the image [16].

2.1.5 Selecting the ROI

The MIPT incorporates the `bwpropfilt` function, which facilitates the selective filtering and extraction of the connected components based on specific properties. This function offers various parameters, including the area, filled area, eccentricity of the ellipse, Euler's number, etc. In this thesis, the area parameter was employed to extract the largest area within a binary mask [17].

2.2 Digital Image Correlation

Digital Image Correlation (DIC) is a computational optical method utilized for tracking and image registration techniques widely used for image data processing, applicable to both 2D and 3D datasets [18]. One possible definition for the two-dimensional discrete cross-correlation coefficient $\gamma(u, v)$ is provided by Equation 2.11, where f is a image, $\bar{t}$ is the mean of the template and $\bar{f}_{u,v}$ is the mean of $f(x, y)$ in the region under the template [19].

$$\gamma(u,v) = \frac{\sum_{x,y}[f(x,y) - \bar{f}_{u,v}][t(x-u,y-v) - \bar{t}]}{\sqrt{\sum_{x,y}[f(x,y) - \bar{f}_{u,v}]^2 \sum_{x,y}[t(x-u,y-v) - \bar{t}]^2}} \tag{2.11}$$

In this thesis, the Digital Image Correlation (DIC) method serves as both a comparative and supplementary technique to the presented image recognition method. Firstly, DIC is employed to verify the accuracy of the displacement results obtained through the image recognition method. Secondly, image recognition techniques are utilized to define the input files required for automating the DIC process.

2.2.1 RaDIC Software

RaDIC is an in-house developed digital image correlation software used for pattern tracking. In this thesis, it is utilized for a comparative analysis with the developed pattern position tracking approach. One of the main purposes of the developed software is to create an initial input file for the RaDIC software by utilizing the knowledge of pattern position, thus automating the process [20].

2.2.2 Differences Between DIC and Image Recognition

Despite utilizing the same input data and providing information about the displacement, the methods employed in this thesis differ in their underlying principles. The DIC method operates by searching for a predefined set of pixels from the previous image in the subsequent image and subsequently may update this set. On the other hand, the image recognition method focuses solely on detecting a randomly generated pattern within the image. The movement data is captured based on the location of the right or left edge of the recognized pattern mask.

2.3 Data Source

In high-rate loading dynamics experiments, the Split Hopkinson Pressure Bar (SHPB) is a commonly used experimental approach. The SHPB setup consists of several bars that, at high velocity, collide with the sample (and each other as well). The setup includes several strain gauges, and their measurements are combined to provide the displacement/velocity data of the bars. Additionally, the region of the bar's impact is monitored by a high-speed camera. All the strain gauge and camera measurements are synchronized. The setup can be illustrated in Figure 2.5.

The datasets used in this thesis were obtained during SHPB experiments using a high-speed camera to capture the region of impact of the bars on the sample.

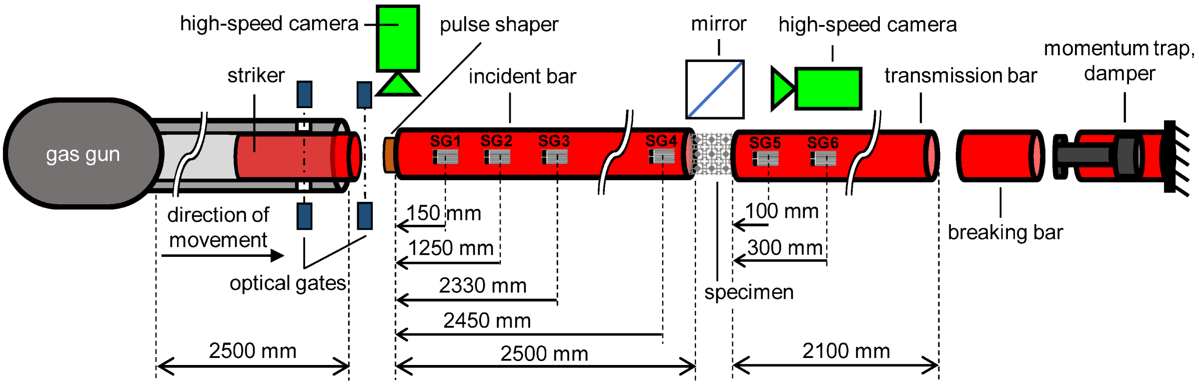


Figure 2.5: Experimental setup of the split Hopkinson pressure bar test [21]

Note: SG – strain gauge

2.4 Data Processing

According to the image recognition process definition, the method yields discrete displacement data for whole numbers of pixels. However, one of the primary objectives of obtaining movement data is to calculate the velocity of the pattern or the sample containing the pattern. The velocity is determined as the first derivative of the displacement. In scenarios involving "*slow*" movement[1], instances occur where the calculated velocity is zero between certain pairs of consecutive images, while it is overestimated in other pairs. To address this issue, a weighted average approach is introduced to mitigate the effect of zero velocity intervals and enhance the accuracy of the velocity calculation. This is achieved by extrapolating the original displacement data, increasing the number of data points, and applying a weighted average scheme.

The value of the extrapolated data point $h(i)$ for the (non)-integer frame number i is given by the term in Equation 2.12, where n is the frame number of original data-point, N is the total number of original data-points, w is the maximal affected distance for calculating average and $f(n,i)$ is the weighted function. The results of the described data processing function is displayed in Figure 2.6 (b), which shows the zoomed detail.

$$h(i) = \frac{\sum_{n=1}^{N} H(n) \cdot f(n,i)}{\sum_{n=1}^{N} f(n,i)} \tag{2.12}$$

[1] Relative to the camera's frames per second. In our specific case, *slow* movement refers to velocities on the order of tens of meters per second

where

$$f(n,i) = w - |i - n|; n \in \ (i-5, i+5)$$

$$f(n,i) = 0; n \in \ \mathbb{Z}^+ \backslash (i-w, i+w)$$

2.5 Functions Summary

This section describes the functions that were specifically developed and implemented to achieve the objectives of this thesis. These functions represent one of the key outcomes of the work. In this section, a detailed description of these functions is provided, including their input and output variables, as well as other relevant details.

2.5.1 imdetect and imdetect2

`mask = imdetect(IM, nHood, limitsum)`

`[mask1, mask2] = imdetect2(IM, nHood, detectnull)`

The output of the function is a binary mask indicating the location of the detected pattern. The mask corresponds to the recognized pattern with the largest area value. In the case of the `imdetect2` function, two regions with the largest areas.

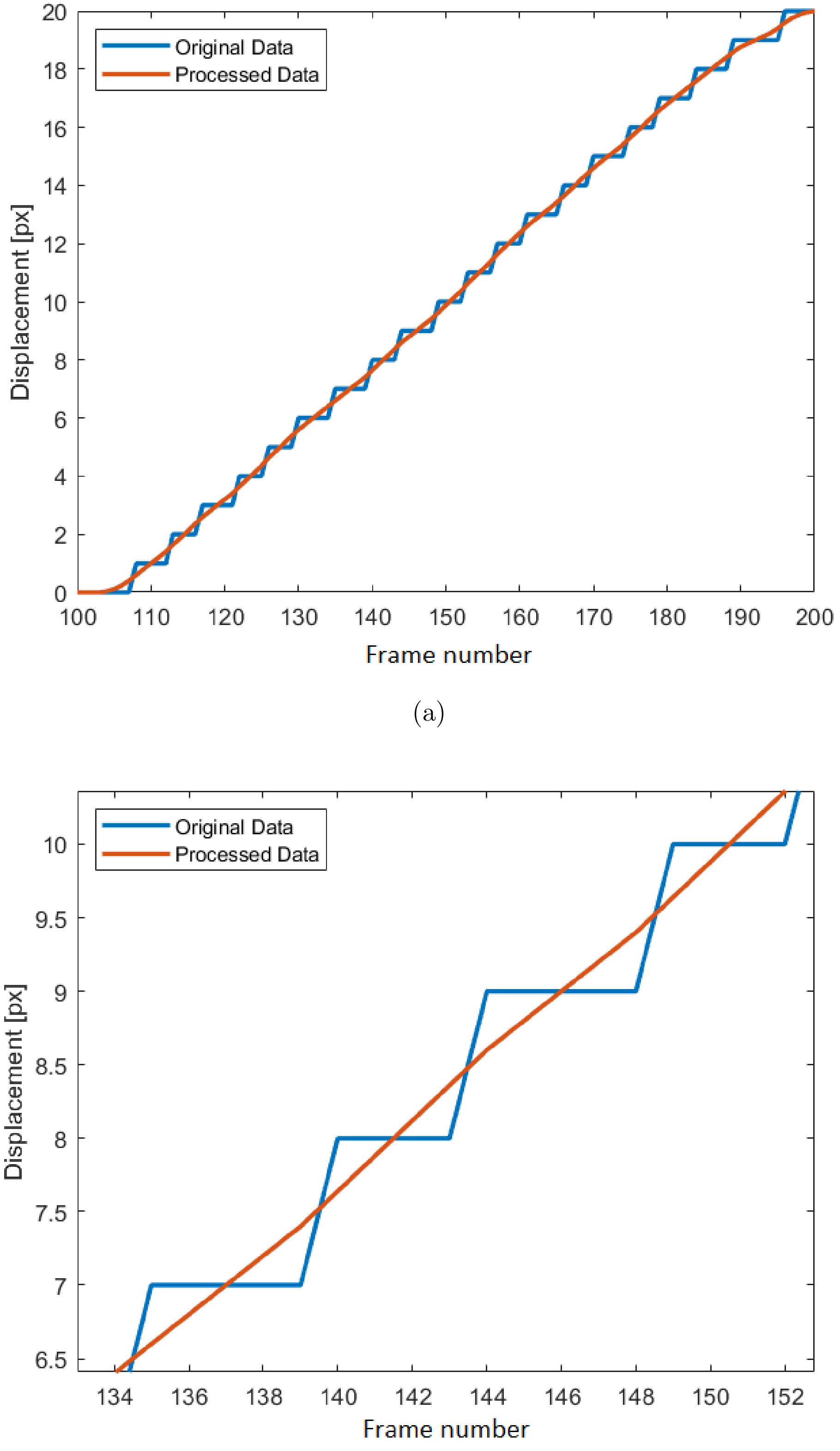
- `IM` – the original gray scale image
- `nHood` – the dimension of the square used for defining the neighborhood
- `limitsum` – the minimum number of pixels in each column in the mask

2.5.2 dataProc

`Upvysl = dataProc(Vysl,up_factor,procSize)`

The output is the processed data with an increasing number of data-points and calculated weighted average.

- `Vysl` – the original results of the displacement
- `up_factor` – factor of increasing density of the new data-points
- `procSize` – maximal affected distance for calculating the weighted average



(a)

(b)

Figure 2.6: Results of the data processing

2.5.3 makeradicfile

`makeradicfile(stIM, dataPath,row_num, col_num, row_size, col_size, startimagenumber, jobname, endingimage, exportimages,ID)`

Save to current folder initial file for SW RaDIC. This file is required for RaDIC operation.

- `stIM` – binary mask of recognized pattern
- `dataPath` – directory, where the image data are stored
- `row_num` – number of the tracking patterns in one row
- `col_num` – number of the tracking patterns in one column
- `row_size` – horizontal dimension of tracking patterns
- `col_size` – vertical dimension of tracking patterns
- `startimagenumber` – image number where the DIC begins
- `jobname` – name displayed in RaDIC SW
- `endingimage` – image number where the DIC ends
- `exportimages` – true/false - option for the RaDIC SW to generate images during the DIC process
- `ID` – integer to prevent a redundant file name

2.6 Graphical User Interface

To facilitate the operation of the functions discussed in Section 2.5, a graphical user interface (GUI) was developed. The presented GUI is one of the main outcomes of the thesis. It gathers most of the tasks that can be performed using the scripts and programs introduced in this thesis into one comprehensive program. Screenshots of the GUI are presented in Figures 2.7 to 2.11. After fulfilling the necessary prerequisites, the interface displays a series of buttons and options in a logical sequence. These elements are presented to ensure the proper flow and organization of actions within the interface.

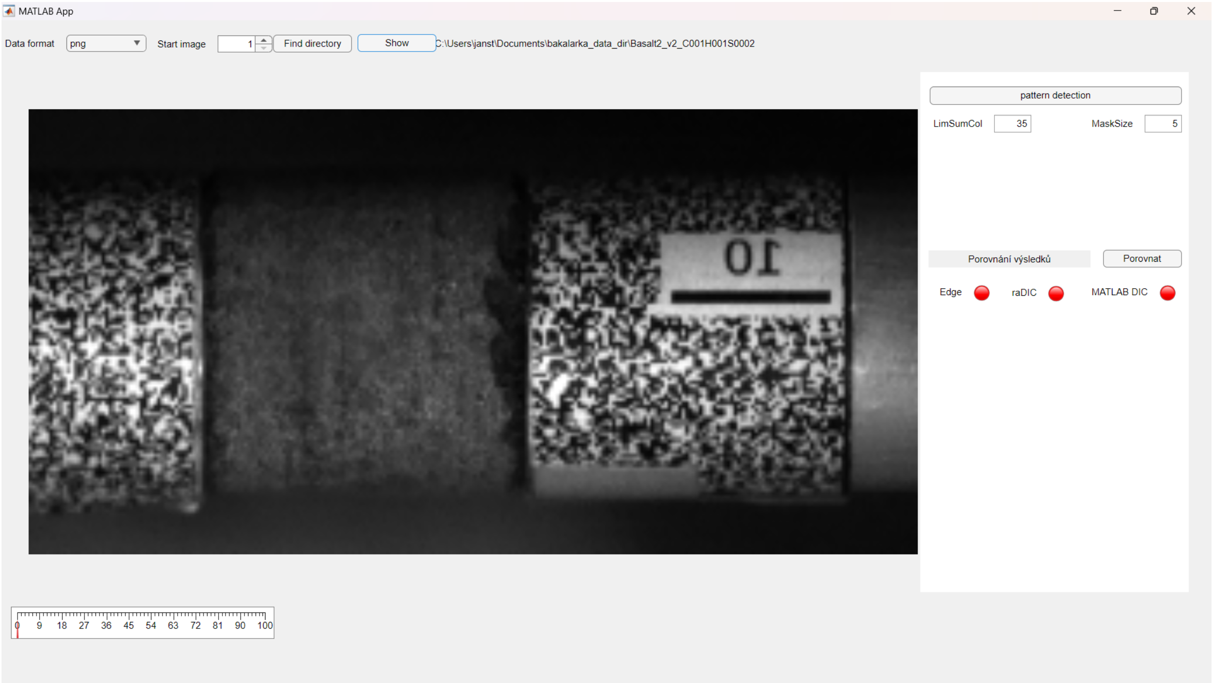
The GUI has the following tasks:

- Provide edge detection
- Export the pattern mask
- Prepare the RaDIC input file
- Provide edge detection tracking and DIC in Matlab
- Compare and display the results of the previous tasks

The comparison of the results is provided to indicate the correctness of the results and to ensure their redundancy.

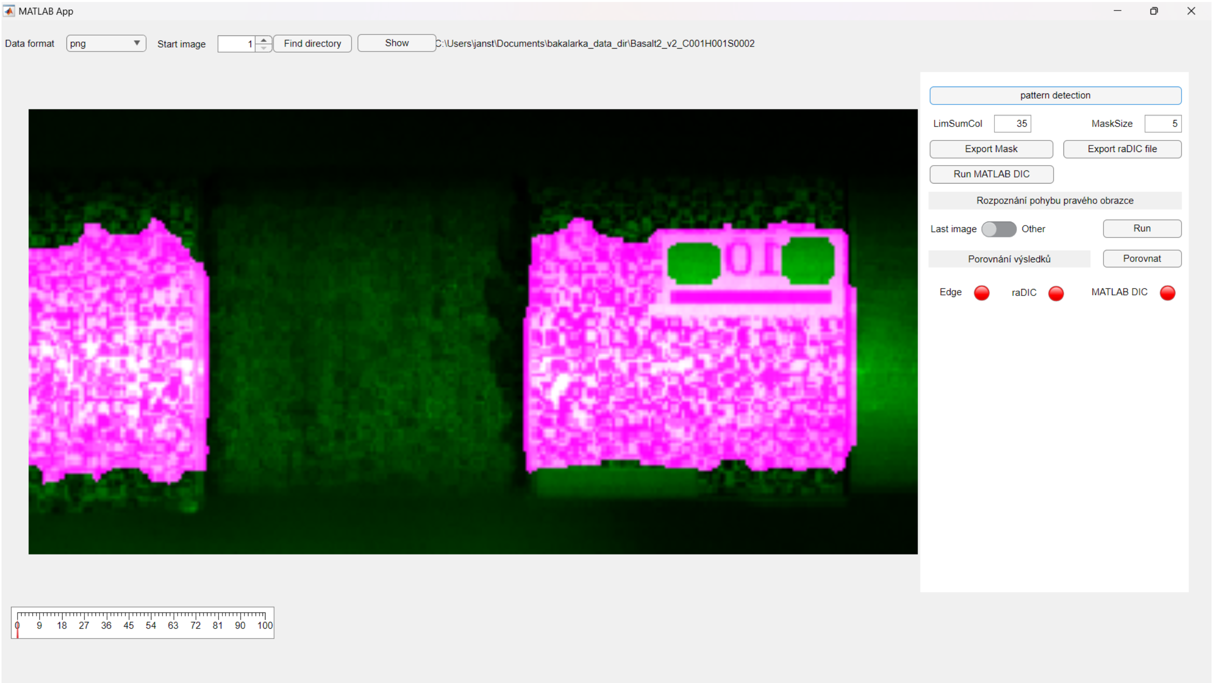
In Figure 2.7 app window is displayed. On the left side of the first line, there is a control panel dedicated to the initial steps. This control panel allows the selection of the images directory and format. After selecting the path by the *Find directory* button, the path is displayed on the right. By pushing the *Show* button, the image is displayed in the center of the app window.

A control panel dedicated to image detection is located on the right side of the GUI. The first section of this panel contains fillable fields for inserting the required parameters for the `imdetect2` function (refer to details in Section 2.5.1). These fields are pre-filled with empirical values used during experiments. Upon clicking the *Pattern detection* button, the `imdetect2` function is called and the resulting mask is displayed in the main application window. This app state is depicted in Figure 2.8. Then, several options, which require a binary mask, are displayed, e.g., run Matlab DIC, make RaDIC file or track of edge movement.



Data format: png
Start image: 1
C:\Users\janst\Documents\bakalarka_data_dir\Basalt2_v2_C001H001S0002
LimSumCol: 35
MaskSize: 5

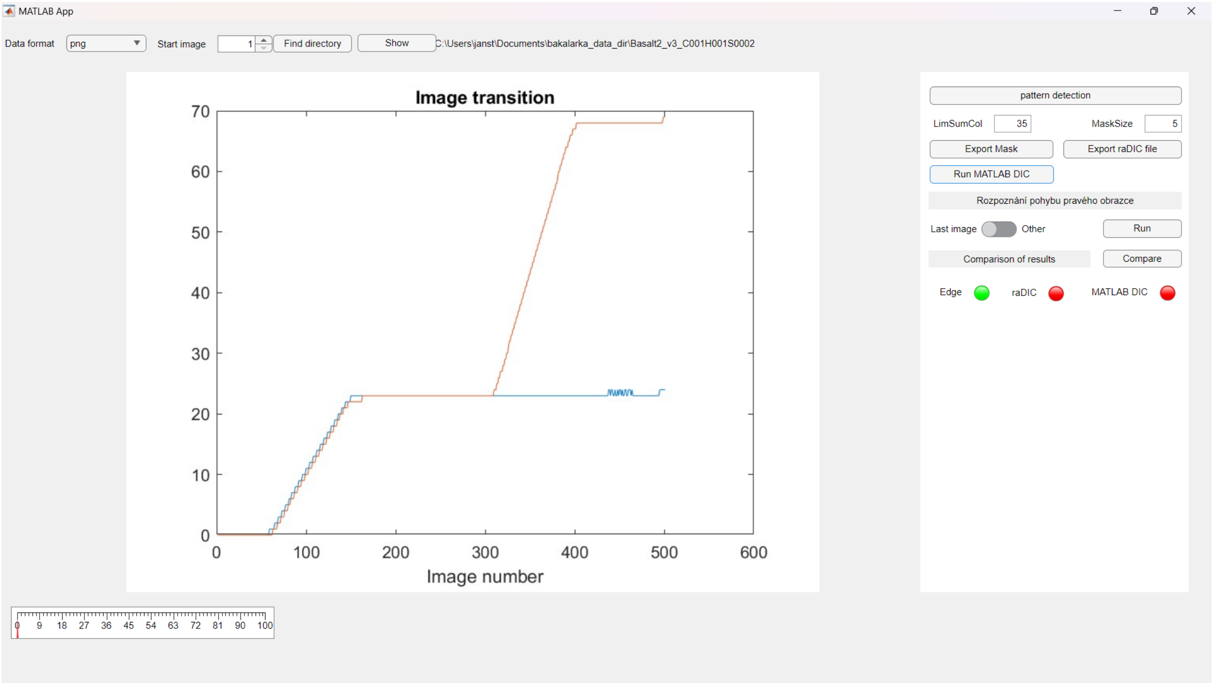
Figure 2.7: Display source data in the App



Data format: png
Start image: 1
C:\Users\janst\Documents\bakalarka_data_dir\Basalt2_v2_C001H001S0002
LimSumCol: 35
MaskSize: 5

Figure 2.8: Display detected pattern

For each of the possible image processing functions (edge detection, DIC in Matlab or in SW RaDIC), a green color is employed to indicate that the respective task has been successfully completed. It can be seen in Figure 2.9. Also, the graph of displacement dependency on the frame number is displayed in the App window.



MATLAB App

Data format: png

Start image: 1

C:\Users\janst\Documents\bakalarka_data_dir\Basalt2_v3_C001H001S0002

LimSumCol: 35

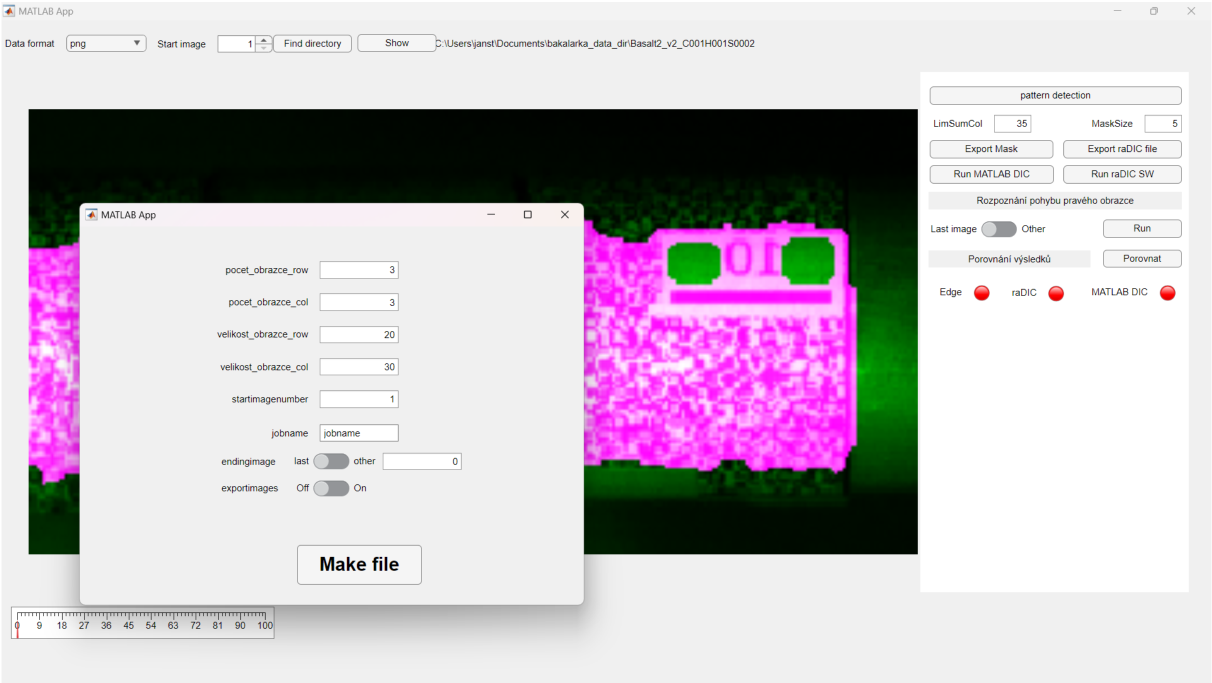
MaskSize: 5

Last image / Other

Edge ● raDIC ● MATLAB DIC ●

Figure 2.9: Screen after the edge detection movement is captured

By pushing the *Export RaDIC file* button, a new dialogue application is opened (Figure 2.10). Within this new window, some of the input parameters for the `makeradicfile` function (Section 2.5.3) can be filled. The parameters referred to pattern location are filled automatically.



MATLAB App

Data format: png

Start image: 1

C:\Users\janst\Documents\bakalarka_data_dir\Basalt2_v2_C001H001S0002

LimSumCol: 35

MaskSize: 5

Last image / Other

Edge ● raDIC ● MATLAB DIC ●

MATLAB App

pocet_obrazce_row: 3

pocet_obrazce_col: 3

velikost_obrazce_row: 20

velikost_obrazce_col: 30

startimagenumber: 1

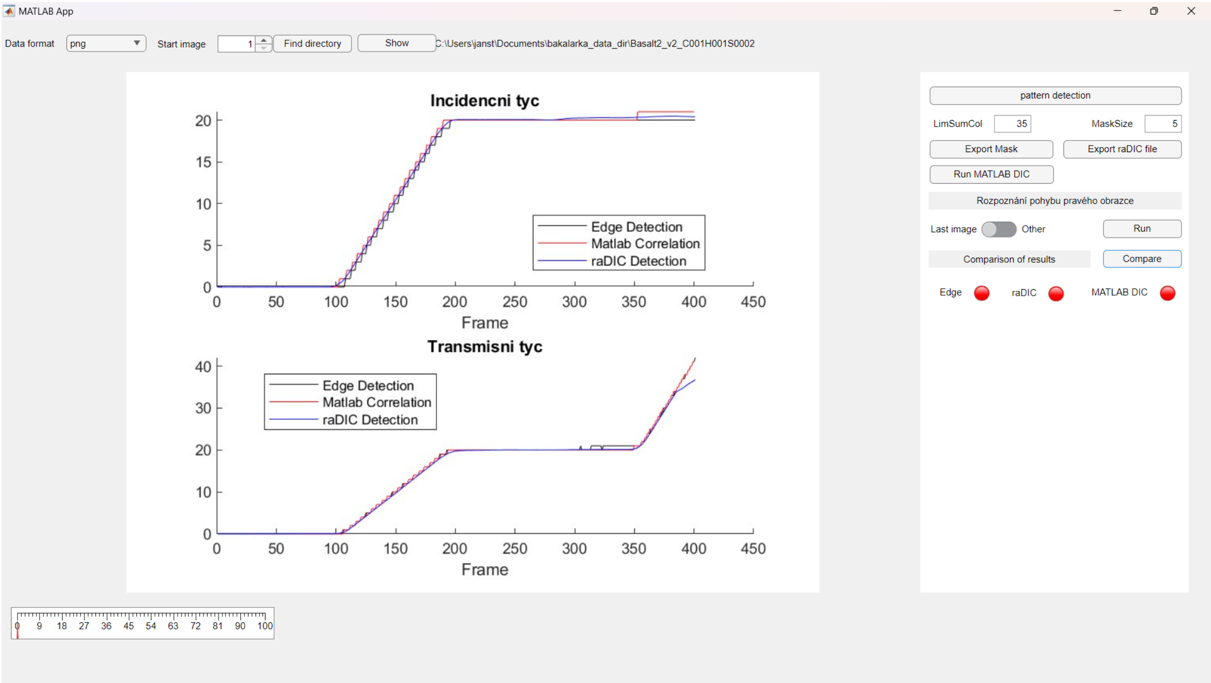
jobname: jobname

endingimage: last / other 0

exportimages: Off / On

Figure 2.10: Make RaDIC file dialog window

Upon clicking the *Compare* button, the results obtained from the edge detection, Digital Image Correlation (DIC) implemented in MATLAB, and SW RaDIC are presented in a single graph, as can be seen in Figure 2.11.



Data format png
Start image 1
C:\Users\janst\Documents\bakalarka_data_dir\Basalt2_v2_C001H001S0002
LimSumCol 35
MaskSize 5

Figure 2.11: Comparison of the results with the different methods

2.7 Computation Protocol

To showcase the accuracy of the edge detection method, three distinct datasets were utilized. These datasets were acquired during the Direct Impact Hopkinson Bar Tests conducted on various materials at the Laboratory of Experimental Mechanics, Department of Mechanics and Materials, Faculty of Transportation Sciences, Czech Technical University in Prague. Detailed information regarding these datasets is provided in Table 2.1, and examples of the corresponding image data are presented in Figure 2.12. The gray scale image data was captured using a FASTCAM SA-Z type 2100K-M-32GB camera. Independent data, such as strain gauge readings from both bars, were also obtained and used for verification of the image data. The images were saved in two formats: raw and png. For the subsequent image processing methods, the png format was utilized. Pertinent information about the camera's resolution, FPS settings, and other relevant details can be found in the cihx file, which is generated in the same folder as the captured images. The cihx file is in an XML format and contains the necessary metadata.

ID	Dataset name	Camera FPS	Resolution [px]
1	Standard concrete	300 000	256 x 128
2	3D printed bulks	300 000	256 x 128
3	PASLS X3	180 000	384 x 200

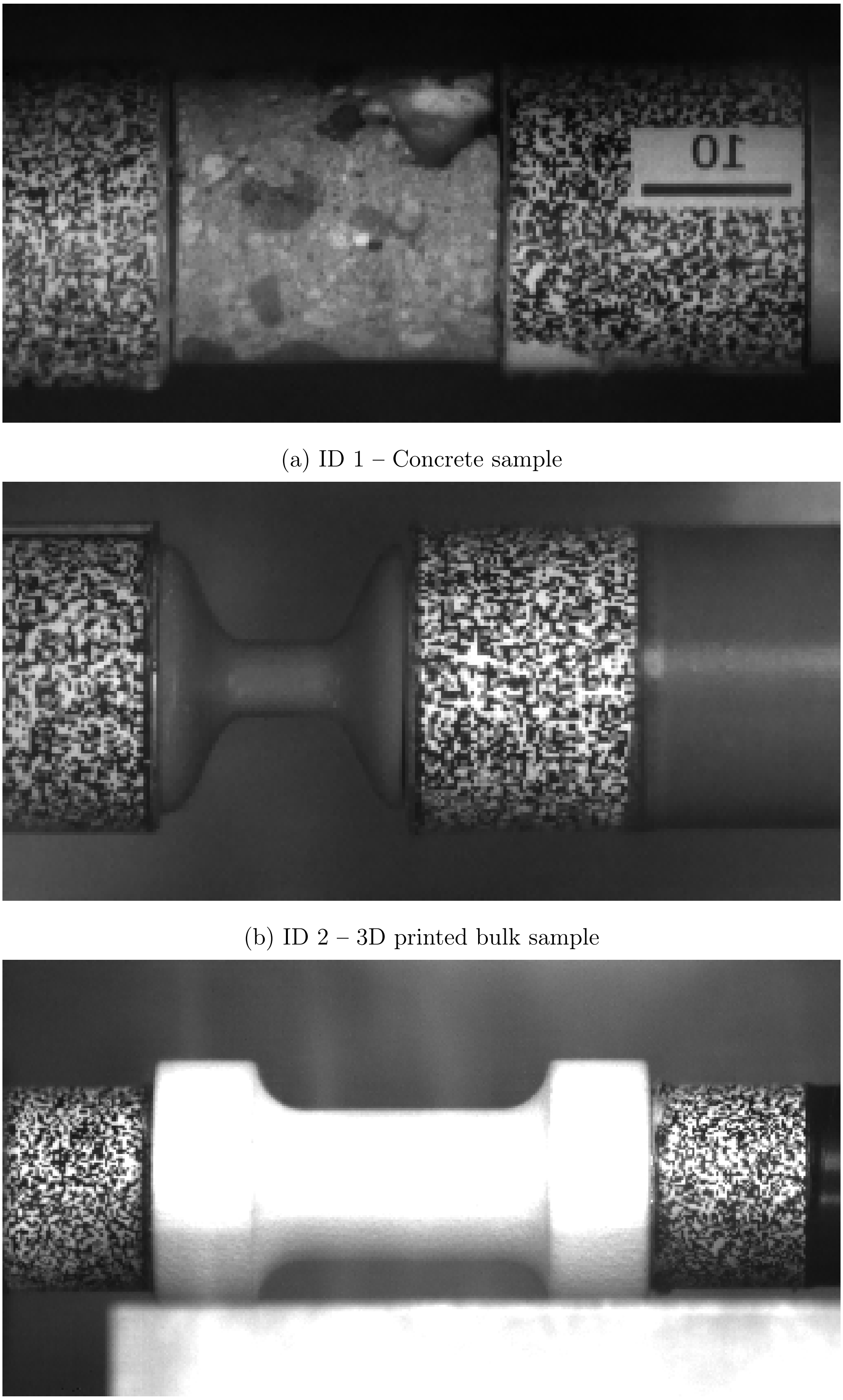
Table 2.1: Details for the datasets used in thesis

To develop the image recognition method, a dataset with an image resolution of 256 x 128 pixels was utilized. This dataset was recorded using the same camera with settings configured to capture images at a high frame rate of 300,000 frames per second (FPS). In contrast to the typical usage of the developed software, this particular dataset consists of images featuring a single random pattern that needs to be detected to optimize the development process. The dataset exclusively contains captured images featuring a single moving bar without any additional samples. An example image from this dataset is displayed in Figure 2.13.

All the elapsed times were recorded on a personal laptop equipped with an Intel i5-11300H processor, 16 GB of LPDDR4 3200 MHz RAM, and running the Windows 11 version 22H2 operating system. It is important to note that for this thesis, Matlab performance toolboxes, such as Parallel Computing Toolbox or GPU Computing, were not utilized. For the purpose of comparison, the RaDIC software, developed in Python version 3.7, was employed.

To visualize the detected mask overlaid on the original image, the built-in Matlab function `imshowpair` [22] is utilized. In this visualization, the recognized mask is displayed in magenta color, while the background is marked with green. The example is shown in Figure 3.1.

To enable a comparative analysis with the non-image data obtained from the strain gauge measurements, a conversion of the displacement graph units from the frame numbers and pixels to the time and millimeters is necessary. This conversion relies on two parameters: the camera's frame rate and the known pixelsize. The frame rate is included in the camera configuration file and is automatically extracted. The second parameter is determined by



(a) ID 1 – Concrete sample

(b) ID 2 – 3D printed bulk sample

(c) ID 3 – PASLS sample

Figure 2.12: Examples of the experiment's image data

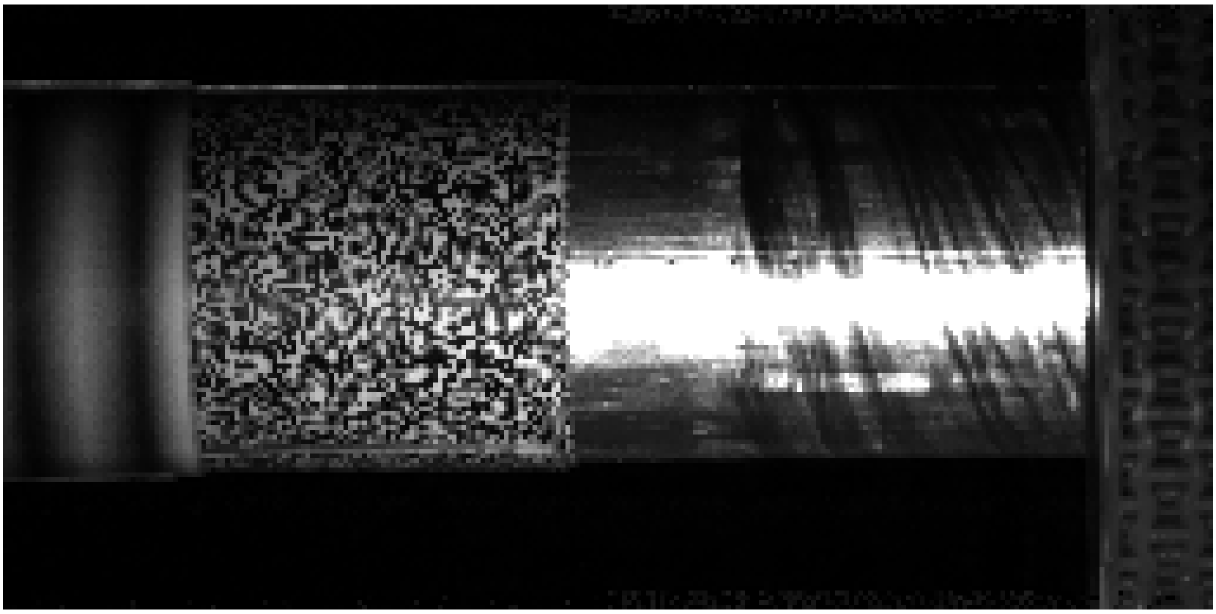


Figure 2.13: Data used for developing the image detection software

a length standard with a known dimension, e.g. 10 mm, as depicted in Figure 2.12 (a) on the right side. The conversion process is semi-autonomous, as a helpful MATLAB script has been developed, but it requires user cooperation.

The datasets were not only selected to demonstrate the ability of the developed method but also because they represent interesting materials used in transportation sciences. Concrete, for example, is commonly used in the transportation infrastructure, such as runways in aviation. 3D printed bulks represent modern materials with great potential for future applications. The PASLS samples are made of high-performance polymers, which can also be used in the aviation industry.

These materials are of interest both for their specific applications and as reference materials for testing the developed method, given their differences in geometry, pattern, contrast, etc.

3. Results

3.1 Pattern Detection

The initial approach to the pattern detection techniques yielded reasonable results, as depicted in Figure 3.1. However, there was a notable issue where the large area outside the pattern, particularly the *edge* between the light bar and the dark background, was erroneously identified as part of the pattern.

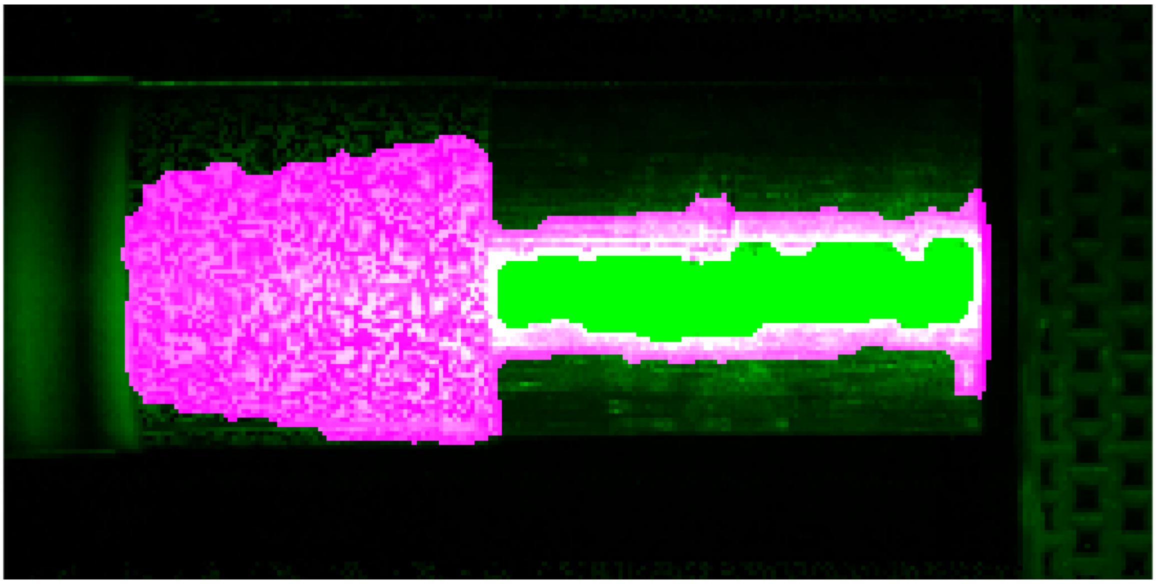


Figure 3.1: Results of the initial approach to the pattern detection

Color legend: magenta – recognized pattern; green/black – background

Subsequently, the method was updated by introducing a parameter to define the minimum number of recognized pixels in a single column, which was empirically set to 35 pixels. In general, if the sum of recognized pixels in a column is below this threshold, all the pixels in that column within the mask are assigned a value of 0. As a consequence of this approach, the vertical edges of the recognized pattern become clearly visible, although this was not originally intended, it will be used in the subsequent steps (detailed in Section 3.2). The outcomes of this modified approach are presented in Figure 3.2, and it was deemed suitable for the purposes of this thesis.

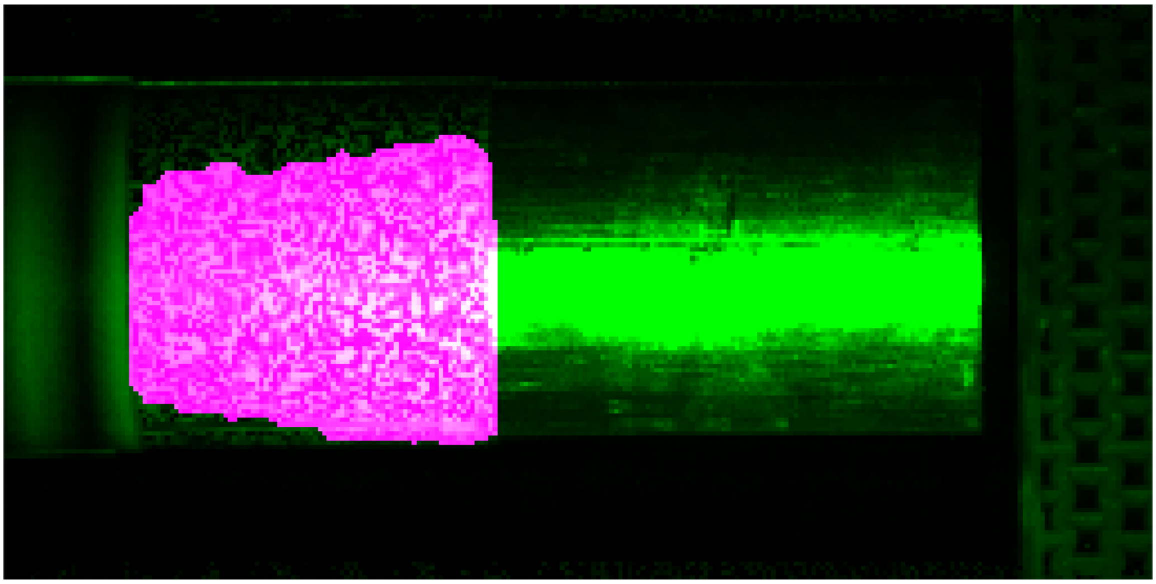


Figure 3.2: Results of the final approach to the pattern detection including the limit for the minimum pixels in a column

Color legend: magenta – recognized pattern; green/black – background

3.1.1 Quantification of Pattern Detection

Due to the high labor requirements involved in result quantification, it was performed on a limited number of data samples. Nevertheless, subjectively speaking, the results obtained using the method on different datasets exhibited similarity. Thus, the limited number of result quantifications can be deemed sufficient to demonstrate the accuracy of the approach.

Figure 3.3 displays the mask of the pattern that was recognized by the author of this thesis. The mask includes parts that are poorly visible in the shadows at the border of the pattern.

To summarize the results, a confusion matrix chart was utilized to classify the pixels, as shown in Figure 3.4. From a subjective perspective, we anticipated that there would be some parts in the mask that were labeled as the background by the function, but in reality, they corresponded to the pattern, albeit with low visibility. This observation aligns with the 75 % of the well-indicated mask pixels. This value could be improved with better lighting conditions during the experiment; however, it is not a major concern, as this specific region of the pattern is not suitable for further image processing methods such as DIC.

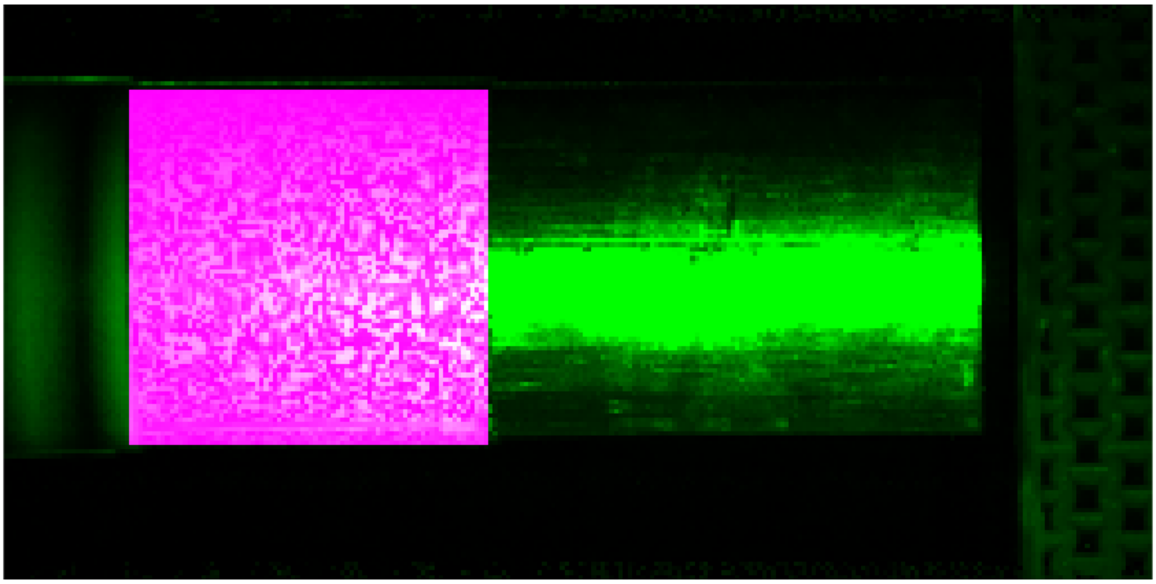


Figure 3.3: Mask detected by the user

Color legend: magenta – recognized pattern; green/black – background

More importantly, the correspondence between the predicted mask value and the real (human-indicated) mask value is highly significant, reaching 94% and 98% accuracy, respectively. This high level accuracy ensures that if the detected area is selected for DIC or any other method, there is a high degree of certainty that the chosen area contains the pattern. As previously mentioned, there is a border part of the pattern that is not detected by the function. Figure 2.13 illustrates that this low visibility portion of the pattern is predominantly situated at the top and bottom regions. Therefore, the existence of clear visible edges on the left and right sides, as mentioned in the previous section, remains unaffected by this level of accuracy.

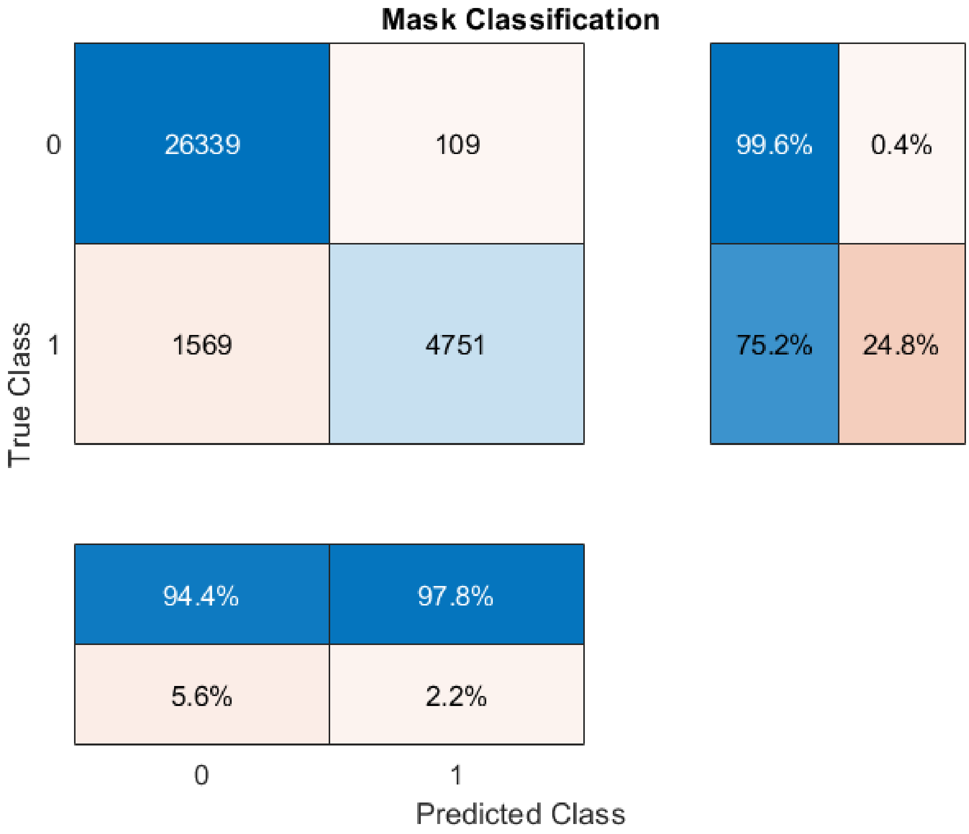
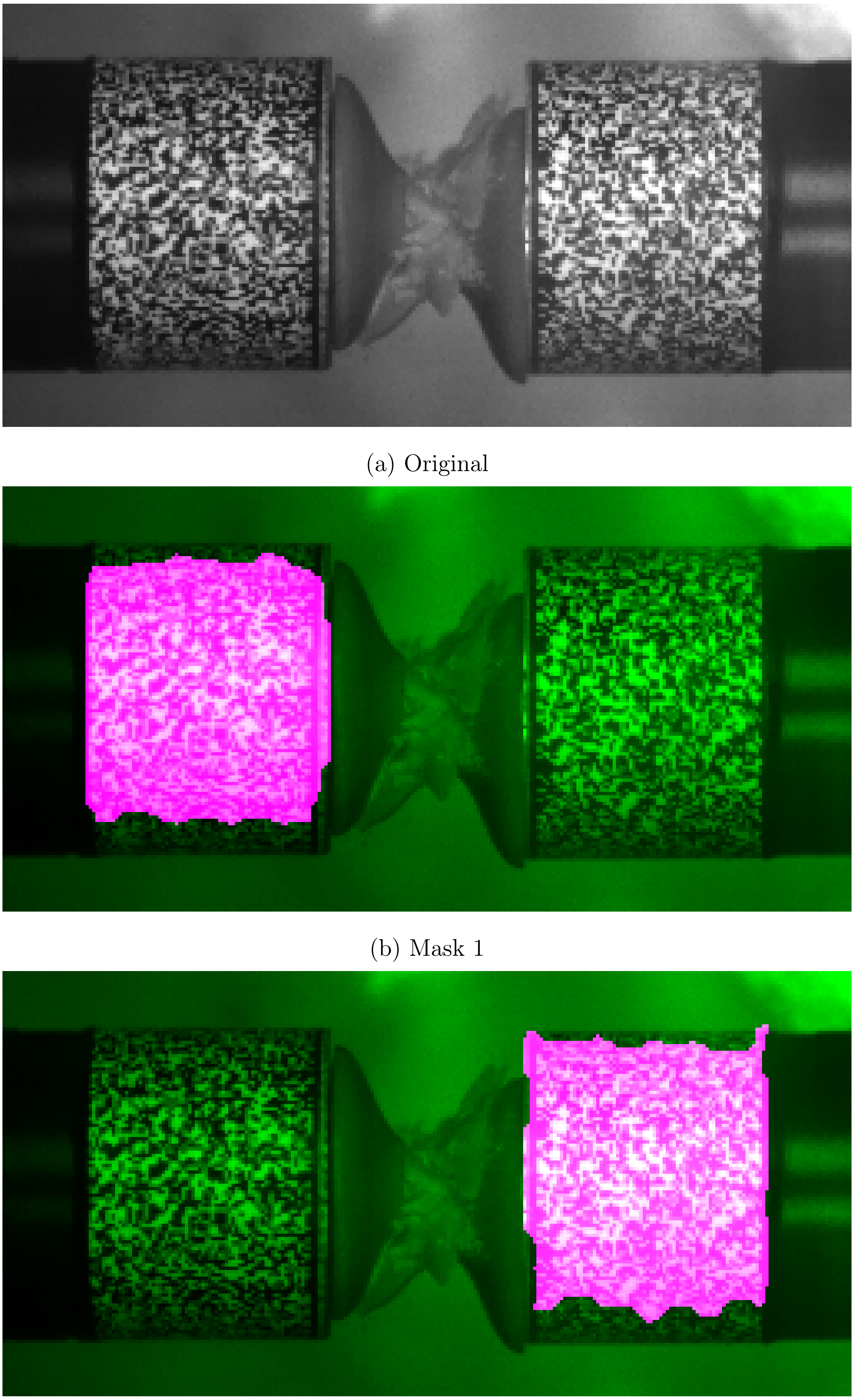


Figure 3.4: Matrix chart of the pixel's classification

3.1.2 Detection of Two Patterns

Once the method's accuracy reached a satisfactory level, the process was updated to detect two separate patterns in image data. The process remained largely the same, using the same logic and sequence of steps as for detecting a single pattern. However, it now preserved the two largest areas, rather than just the largest area. Additionally, a sorting logic was implemented, which resulted in the pattern on the left side being returned as `mask1` and the pattern on the right side being returned as `mask2`. Figure 3.5(a) shows an example of an image containing two patterns for detection. The results of the pattern detection function with the additional sorting logic are shown in Figures 3.5(b) and (c).



(a) Original

(b) Mask 1

(c) Mask 2

Figure 3.5: Two detected patterns in the image

Color legend: magenta – recognized pattern; green/black – background

3.2 Using Recognized Mask Data

In the event that the pattern is detected in the entire set of images[1], valuable information about the movement of the sample with the pattern can be extracted from the series of mask data for each image. We have two hypotheses on how to utilize these masks. Firstly, we can track the center of gravity of the recognized pattern. However, as illustrated in Figure 3.6, the black line representing the tracked center of gravity does not align well with the other independent methods. This phenomenon occurs due to the significant increase in the detected pattern size when it moves from a section with poor light conditions to a better condition. This effect is particularly observed on the sides of the picture. Consequently, in the initial frames, the displacement of the pattern calculated by this approach is notably lower. However, in the subsequent part, the trend aligns with the results obtained from the comparative methods.

As an alternative approach, we discovered that vertical edge detection (specifically the right edge of the pattern in this case) proves to be significantly more suitable for evaluating the image data. This phenomenon is caused by the minimum number of mask pixels declared in each column, as discussed in Section 3.1.

As depicted in Figure 3.6, the results obtained from RaDIC SW, DIC implemented in Matlab, and the edge detection method (with the exception of a few last images in the sequence) exhibit similar outcomes. By utilizing the upscale method for Matlab DIC and the edge detection results, as discussed in Section 2.4, the edge detection data can be regarded as the true results of processing the image data. However, it is important to consider the requirement of *good image conditions* which was formulated during the subsequent experiments with real samples. In the cases where debris from a disintegrating sample covers the pattern, especially its edges, the method may fail to provide accurate results.

[1] Assuming that the pattern remains unchanged during the movement and the camera remains in a static position

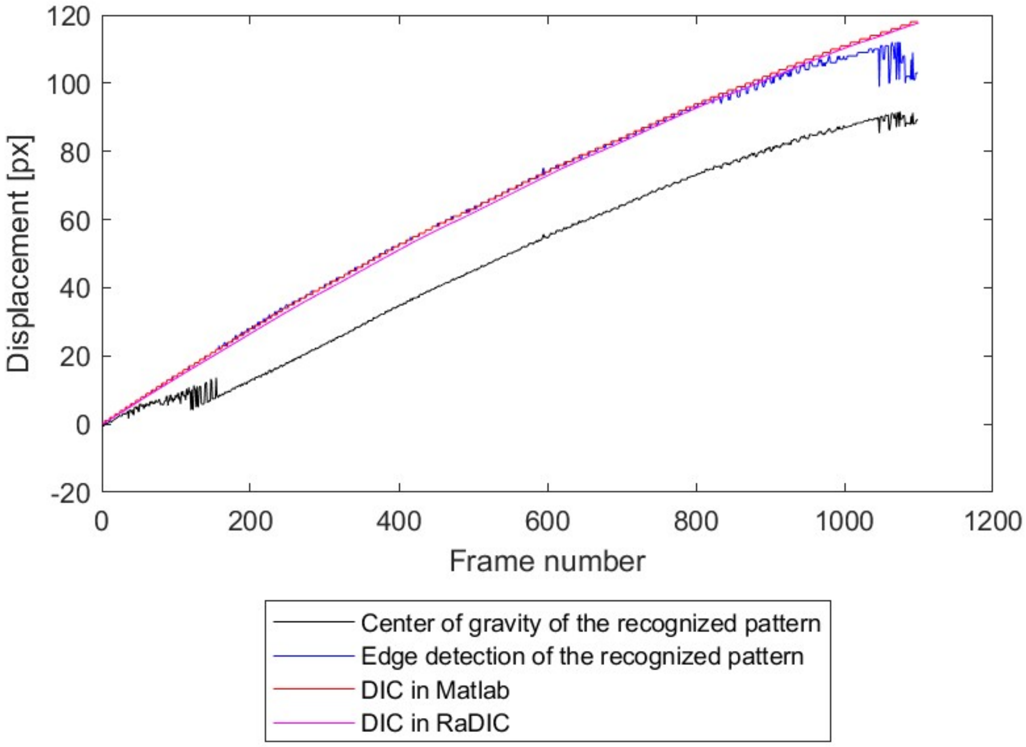
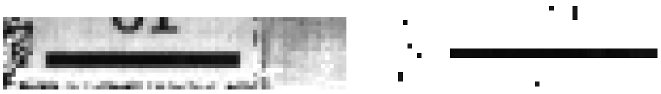


Figure 3.6: Comparison of the used methods

Dataset with 1 pattern for detection – Figure 2.13

3.3 Unit Conversion

As mentioned in Section 2.7, the unit conversion from pixel to millimeters (or equivalent length unit) is required for the application of the developed method. To achieve this, a small semi-automatic MATLAB program was written. The user inputs the image, which is then cropped to the length label section (Figure 3.7 (a)).



(a) Zoom to length label (b) Binarized image

Figure 3.7: Unit conversion program

The image is then binarized by selecting a threshold value of intensity empirically to 0.1 (Figure 3.7(b)). The sum of the pixel intensities in each column is then calculated and displayed to the user in a graph (Figure 3.8).

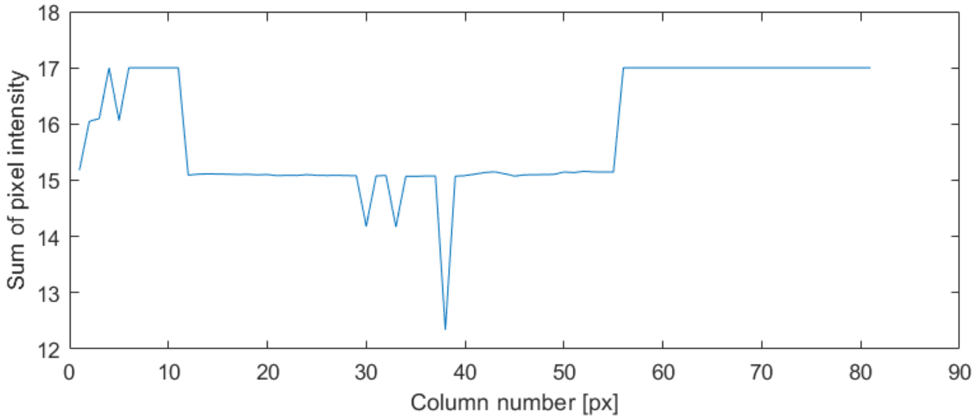


Figure 3.8: Graph of the sum of the pixel intensity in the columns

The user then defines a threshold for the sum of the pixel intensities. In the context of the graph in Figure 3.8, the threshold is set to 16. The final task of this program is to calculate the length of the continuous region below this threshold. However, there may be more than one continuous region below this threshold, and the user has to select the length value that corresponds to the length of the label of a standardized size. In this particular case, the length of the region is 44 pixels and the real length is 10 mm. With this knowledge, it is possible to provide information about the displacement/velocity in adequate units for laboratory practice.

3.4 Comparative Analysis

To conduct a comparative analysis of the described method, which involves tracking the edge of the recognized pattern, alternative approaches were employed using the same data input (image data). Specifically, the two alternate methods utilized for comparison were DIC in Matlab and RaDIC, in addition to the image-independent data obtained from strain gauge measurements during the experiments. For this purpose, several datasets, as discussed in Section 2.7, were employed. The analysis of these datasets revealed that the results, in terms of accuracy and suitability for different applications, may vary based on specific conditions.

3.4.1 Compact Concrete Sample

In this section, we employed a specific sample that did not collapse through the pattern, and an example of the corresponding image data is displayed in Section 2.7, Figure 2.12.

Figure 3.9 illustrates the dependencies of the incident (right) bar velocity over time, which were measured or processed using different approaches. Additionally, Figure 3.10 showcases the velocity of the transmission (left) bar. The velocity obtained by tracking the edge of the recognized pattern is represented by the black line. The separated curves are further delineated in Figures 3.12 and 3.13.

While a quantified comparative analysis is presented in the subsequent chapter, this section primarily focuses on visually comparing the trends in the results, which are apparent to the observer. As evidenced in Figures 3.9 and 3.10, the results obtained using all the optical approaches closely align with the velocity values measured by the strain gauge. However, it is noteworthy that visually, the values from the optical approach appear to be slightly overestimated in comparison to the strain gauge measurements. I surmise that this overestimation, which is consistent across all the image processing approaches, may be attributed to inaccuracies in the unit conversion process, which operates solely with integer values.

Furthermore, the figures reveal individual deviations from the trends of the other curves – notably, for Matlab DIC in the incident bar at times 5.4 ms and 5.5 ms, and for the edge detection in the transmission bar at time 5.4 ms. The abnormality observed in the case of the edge detection can potentially be explained by the local deterioration of the light conditions or the presence of debris near the edge of the pattern, leading to a recognition failure. However, an adequate explanation for the abnormality observed in the Matlab DIC results for this dataset was not identified.

In Figure 3.11, the detailed view of the peak observed in the incident bar velocity data is presented. A noticeable observation is that the Matlab DIC and edge detection approaches tend to preserve the waves in the peak better than RaDIC, particularly in the context of strain gauge values, as seen in the curves at time 4.85 ms. However, the question remains unanswered as to whether this phenomenon is merely a coincidence or a distinct feature of the approaches, particularly in light of the presence of *opposite* peaks at time 4.75 ms.

The absence of these peaks in the results obtained through the RaDIC approach is not particularly surprising, given that RaDIC updates the image during the DIC process, while the DIC in Matlab does not, and the edge detection approach, by its definition, lacks a similar option.

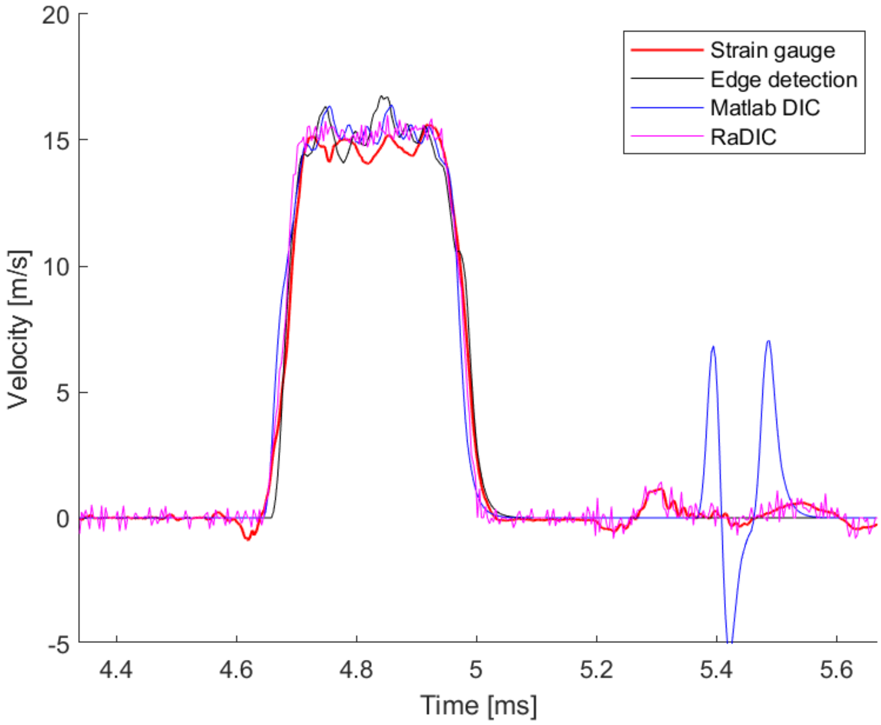


Figure 3.9: Velocity of the incident bar – concrete sample

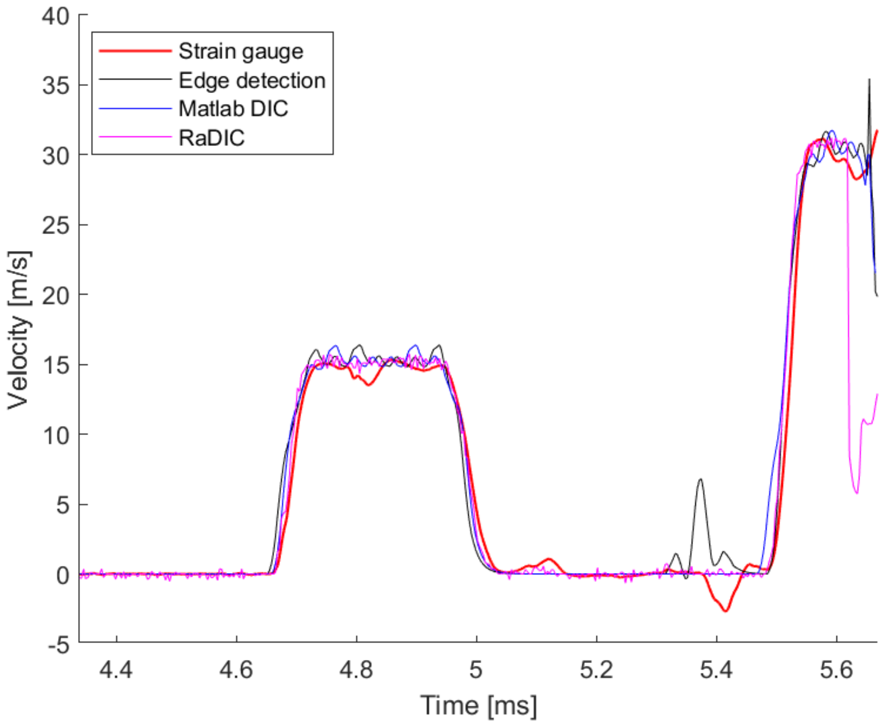


Figure 3.10: Velocity of the transmission bar – concrete sample

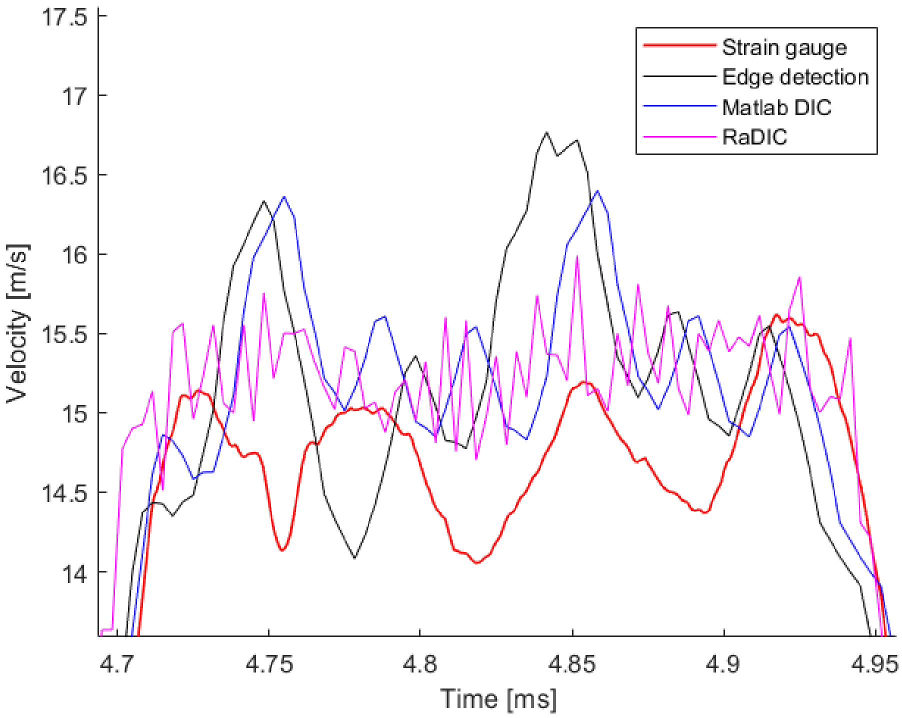


Figure 3.11: Velocity of the incident bar – concrete sample – zoom to peak

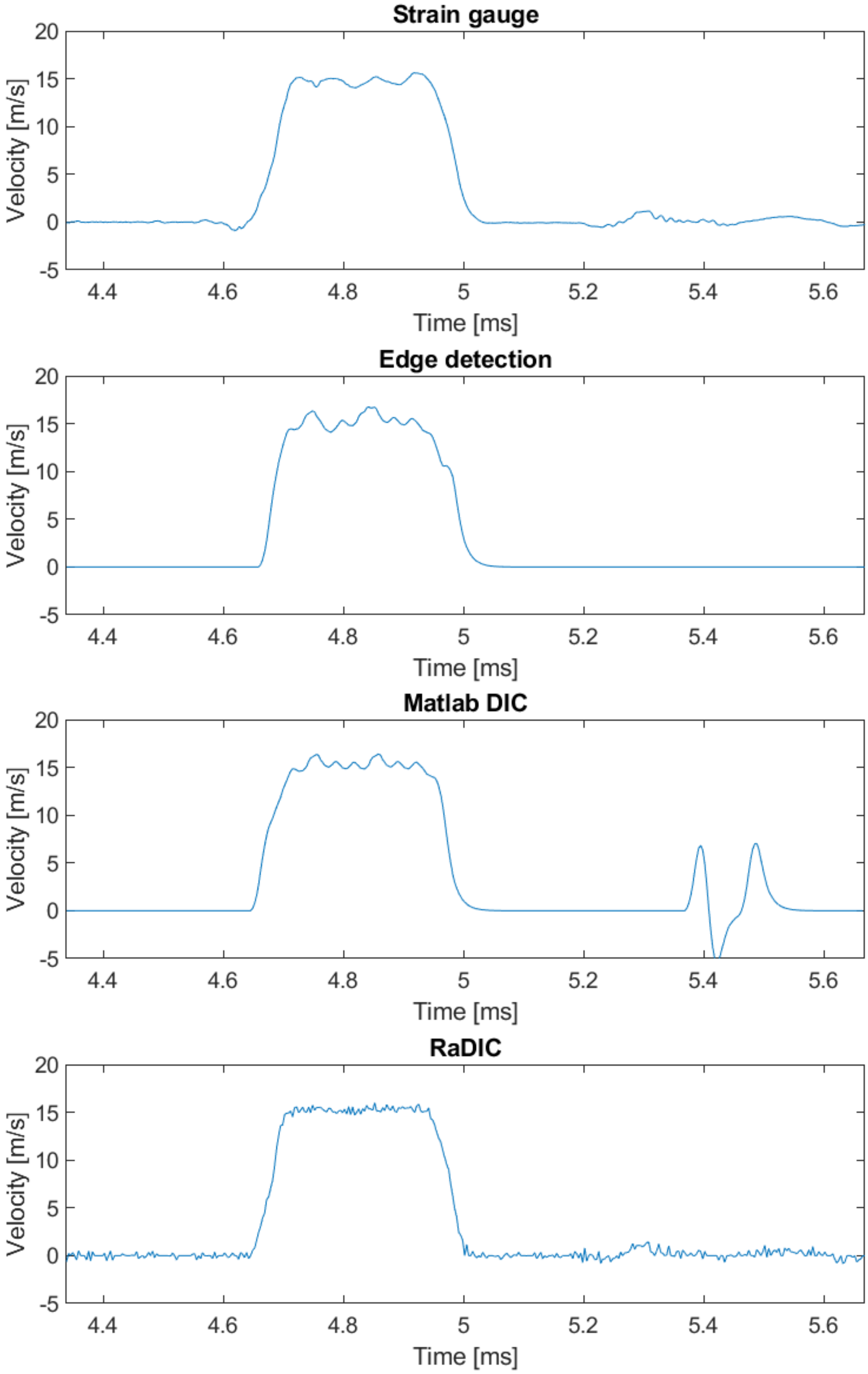


Figure 3.12: Velocity of the incident bar – concrete sample (separated curves)

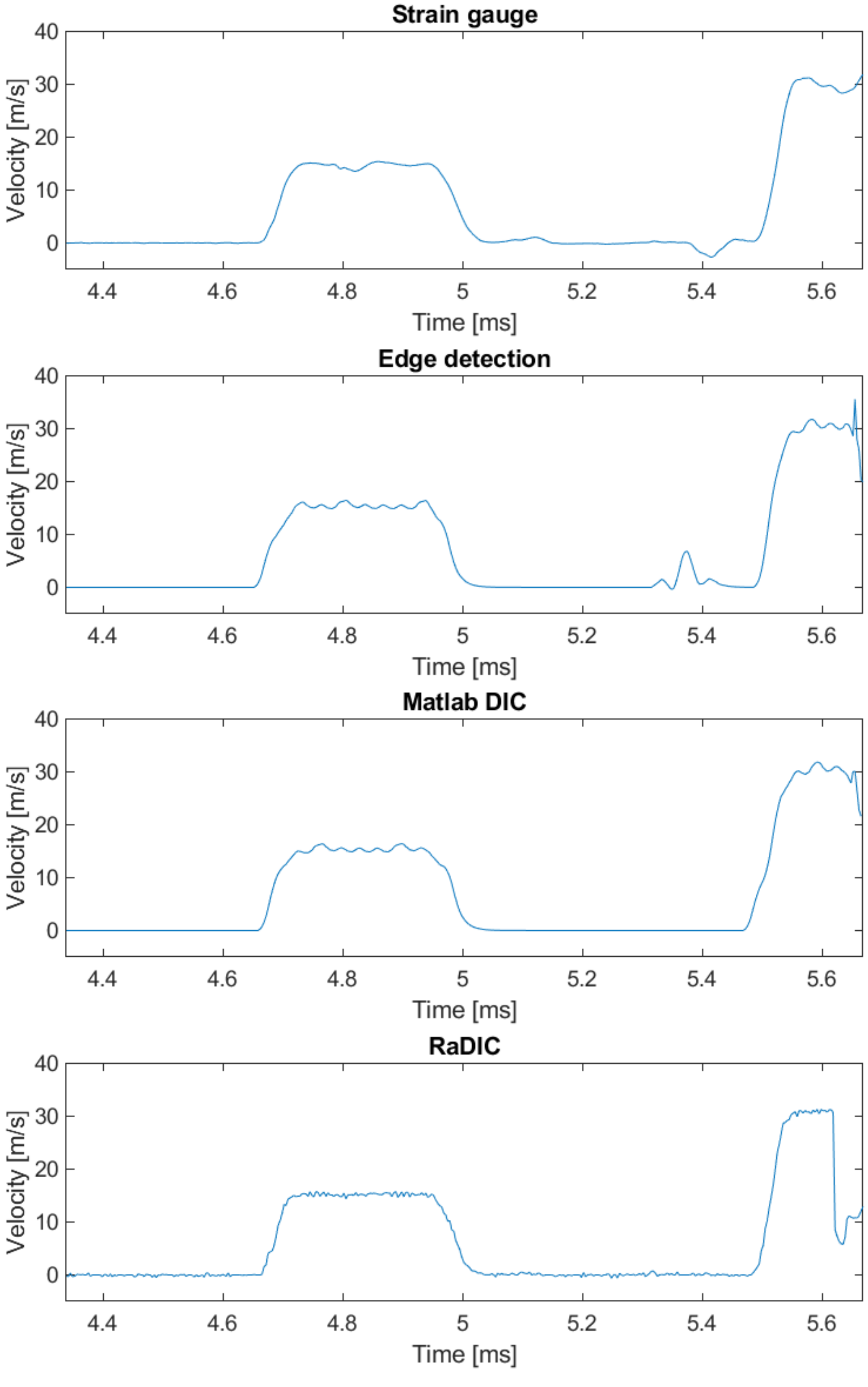


Figure 3.13: Velocity of the transmission bar – concrete sample (separated curves)

3.4.2 3D Print Bulk Sample

Similarly to the previous section, this section presents a visual comparison of the results obtained from a 3D print bulk sample, as shown in Figure 2.12 (b). In Figures 3.15 to 3.18, the individual velocity graphs of the sample are visualized.

This dataset was specifically chosen to demonstrate a phenomenon where the results for the incident bar from the edge detection approach closely correspond with the strain gauge data and other optical approaches. However, intriguingly, the results for the transmission bar do not follow the trend of the strain gauge or RaDIC results. Surprisingly, even the DIC in Matlab does not align with the other results, that are considered accurate. It should be noted that the implementation of DIC in Matlab in this study is only preliminary, and its complete integration into the *comparing software* is not within the scope of this thesis. On the other hand, the inaccuracies observed in the results from the edge detection approach can be attributed to the presence of a small amount of sample debris at the bar's pattern, as can be seen in Figure 3.14. The limited occurrence of sample debris, primarily at the onset of pattern tracking, suggests that the DIC method, especially RaDIC, can yield results that closely resemble non-optical methods.

In general, this dataset exemplifies the potential collaboration between the edge detection approach and the RaDIC software. When the edge detection approach encounters difficulties, knowledge of the pattern location from the initial image allows for the automated image processing using the RaDIC software, resulting in outcomes that closely align with the non-optical methods. This serves as a compelling example of how these two methods can effectively complement each other.

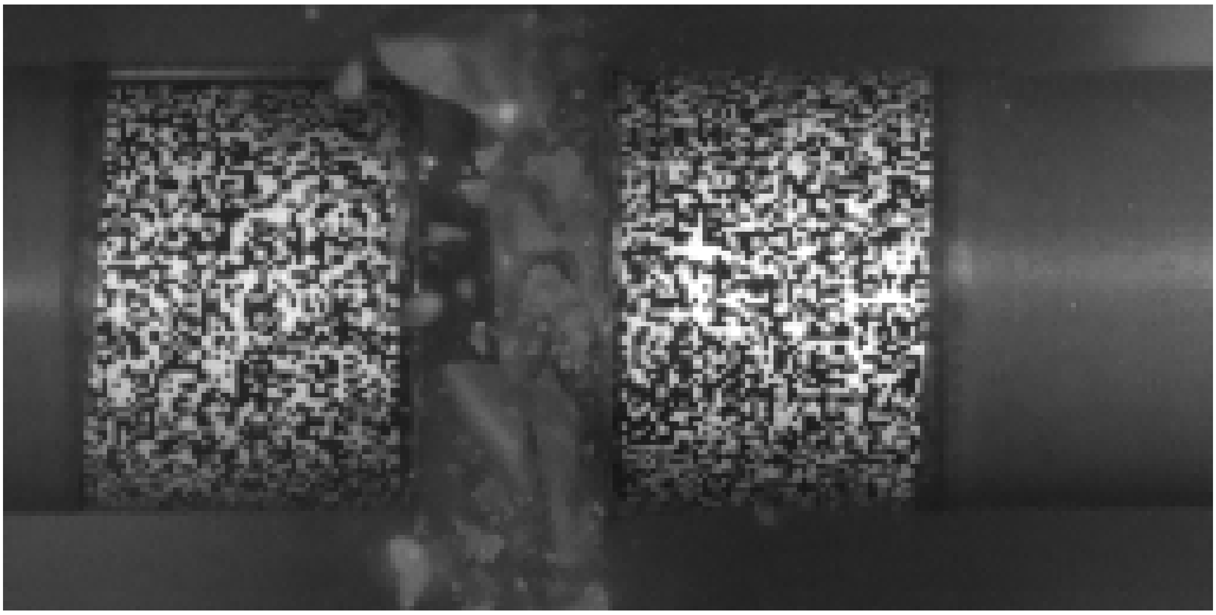


Figure 3.14: Concrete sample debris

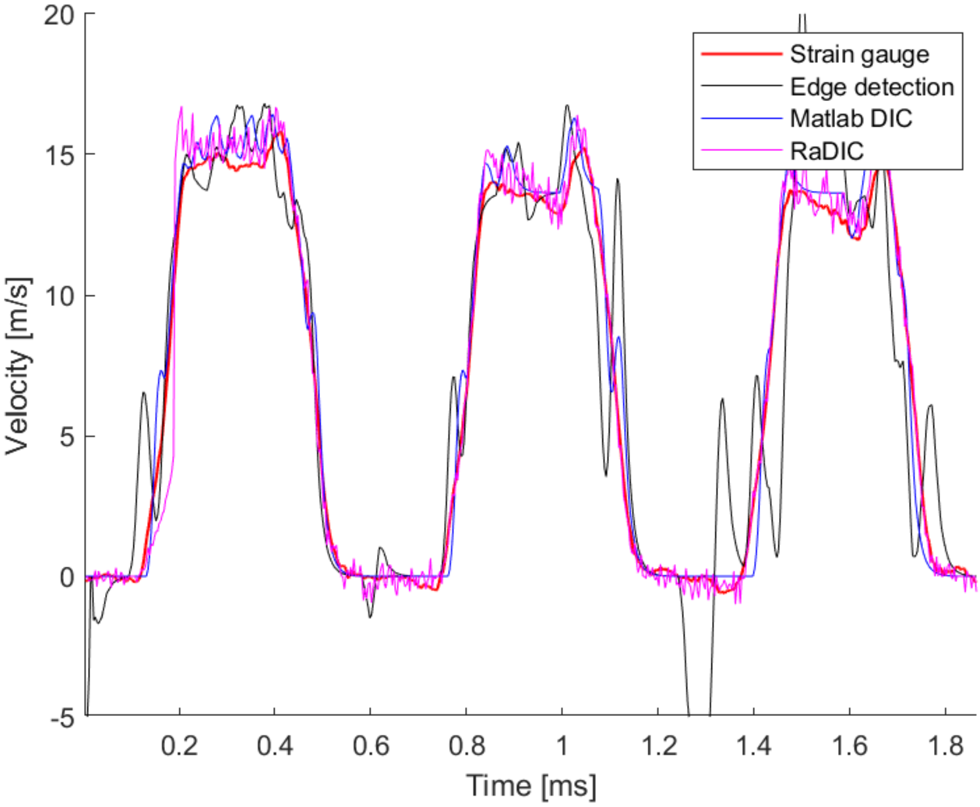


Figure 3.15: Velocity of the incident bar – 3D print bulk sample

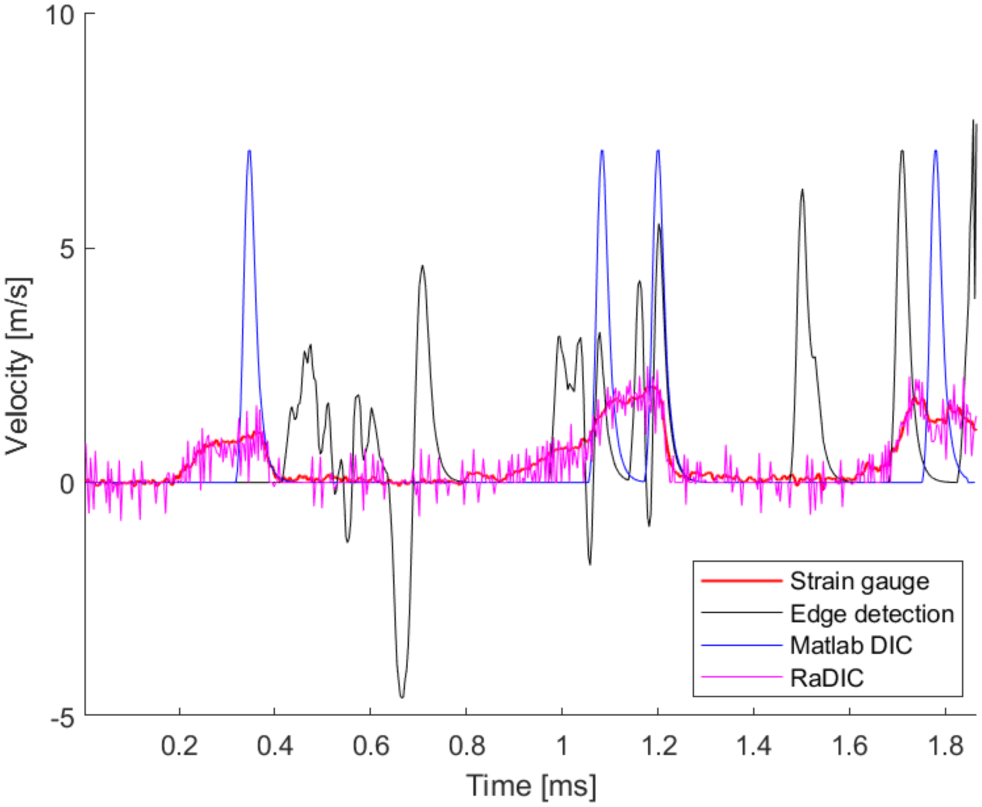


Figure 3.16: Velocity of the transmission bar – 3D print bulk sample

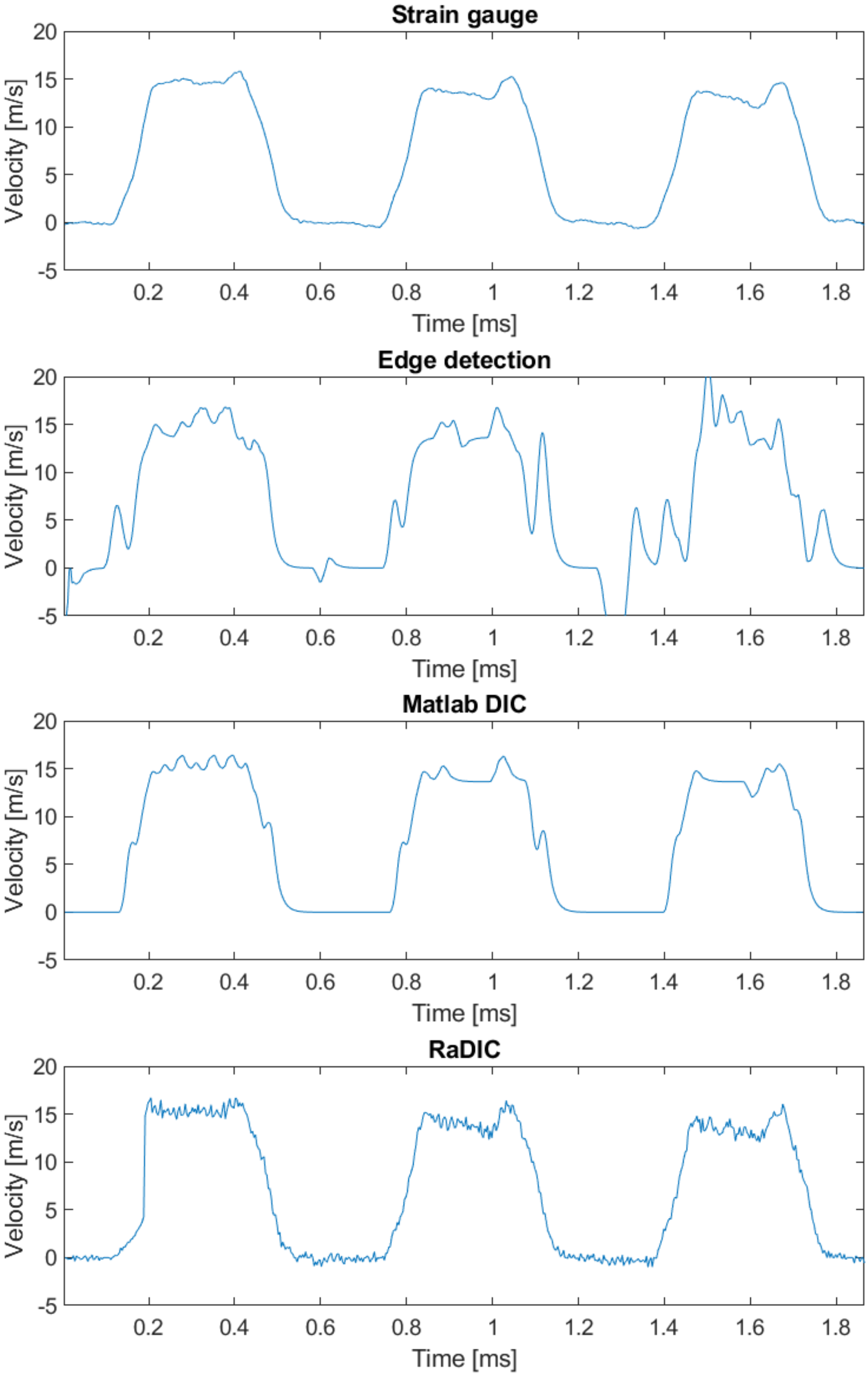


Figure 3.17: Velocity of the incident bar – 3D print bulk sample (separated curves)

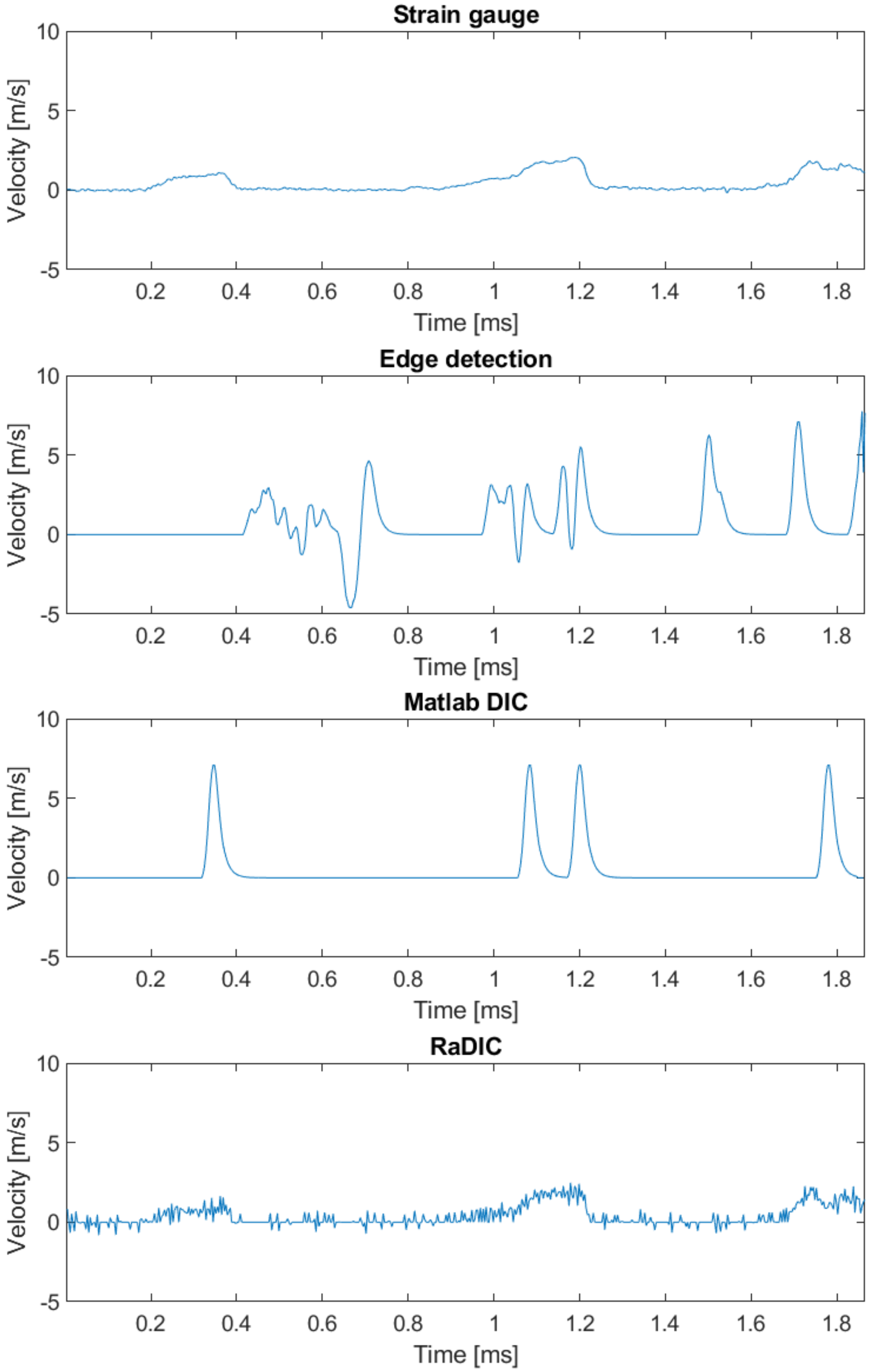


Figure 3.18: Velocity of the transmission bar – 3D print bulk sample (separated curves)

3.4.3 PASLS Sample

This dataset was chosen to demonstrate the limitations of the developed method. Since the data lacks a length label, a direct unit conversion is not feasible, leading to use bar diameter as a length label. Figures 3.19 and 3.20 present the displacement graphs for the strain gauge measurements and image data. The velocity graphs cannot be displayed as they are unreadable, and no discernible data trend can be observed within them. This dataset is not well-suited for image processing, as the presence of sample debris significantly affects a large area of the pattern, as observed in Figure 3.21. Consequently, the RaDIC method, which updates the searched pattern, loses the original pattern for tracking, as depicted in Figure 3.19 at time 1.5 ms, corresponding to Figure 3.21. According to the physical aspects of the measurements, the data measured at the time of the presence of the sample's debris is deemed uninteresting, as all the relevant material changes have already been accounted for.

However, excluding instances when sample debris covers the pattern, the edge detection method partly follows the trend obtained from strain gauge measurements. The method logic remains unaffected during short periods of pattern loss (unlike the RaDIC approach, which is significantly affected), and overall, the movement trend aligns with the strain gauge measurements.

Moreover, with additional user input, it is possible to extrapolate erroneous results around time 1.5 ms to achieve the same trend. However, the results for the transmission bar displayed in Figure 3.20 are strongly affected by the sample debris than those for the incident bar. Since the transmission bar is mostly stationary, its trend cannot be adequately compared. Nonetheless, it is noteworthy that none of the optical approaches capture the trend of increased movement at the end of the figure.

This dataset serves to illustrate that while the RaDIC approach may encounter failures in specific cases, the results obtained through the edge detection approach are more valuable, thanks to its distinct data processing logic. Additionally, in cases with excessive sample debris covering the pattern, none of the optical methods employed in this thesis yield satisfactory results.

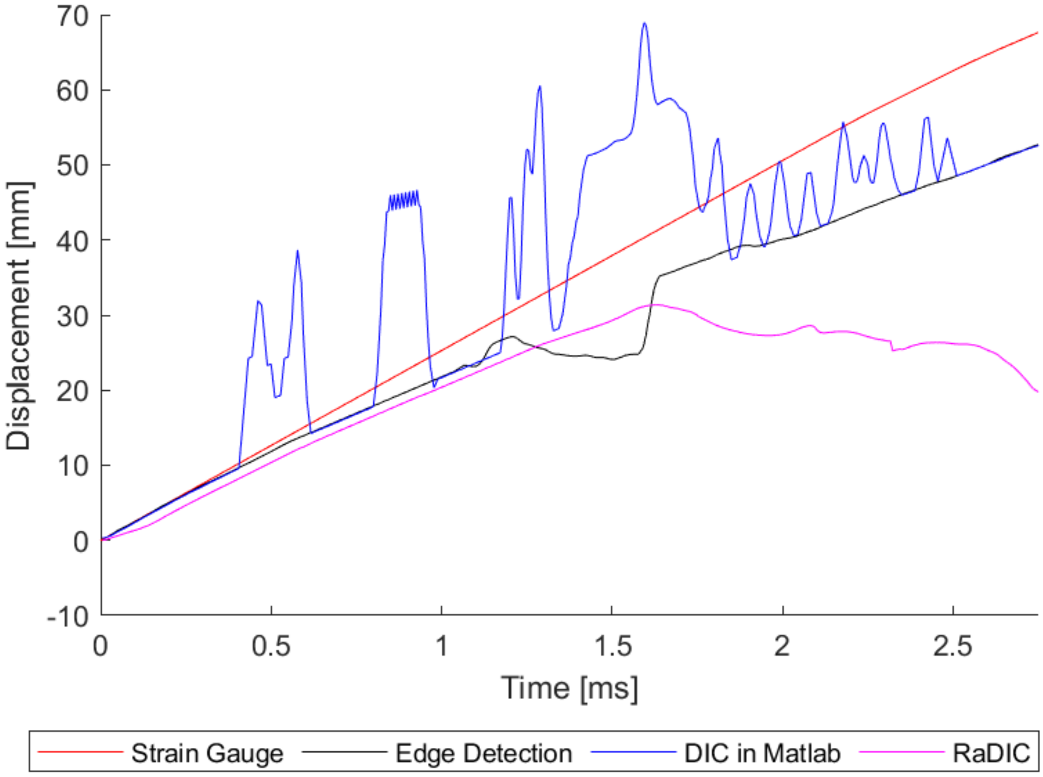


Figure 3.19: Velocity of the incident bar – PASLS sample

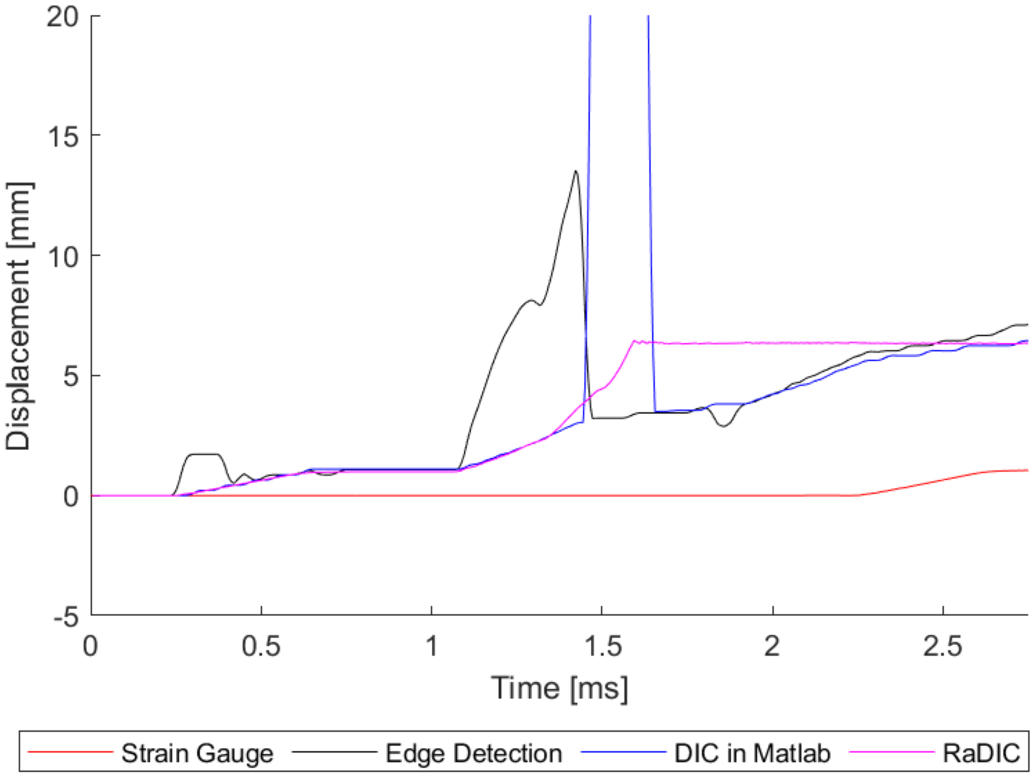


Figure 3.20: Velocity of the transmission bar – PASLS sample

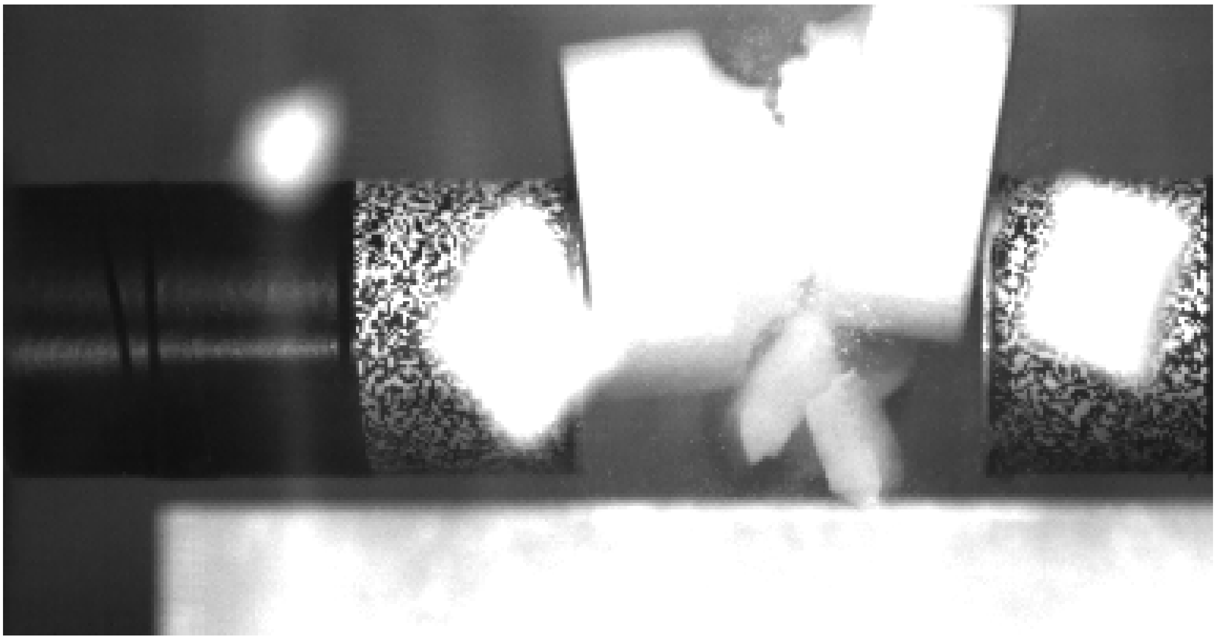


Figure 3.21: PASLS sample debris

3.5 Quantification of Results

In this section a detailed discussion of the results obtained from the concrete sample dataset presented in Section 3.4.1 is provided. This dataset was chosen for a detailed comparison due to its high trend correspondence with the strain gauge measurements. In the following section, we describe the parameters used for the quantification of the results, followed by the presentation of the numeric results.

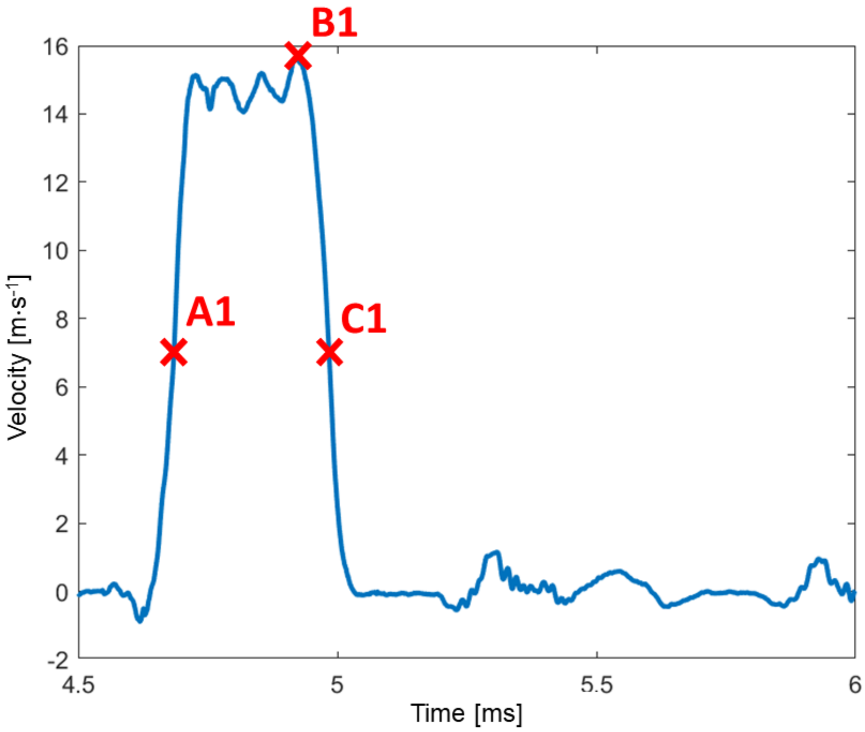
3.5.1 Definition of the Monitored Parameters

In the graphs presented in Figure 3.22, defined points are shown to provide the numerical quantification of the results. The definitions of these points are detailed in Table 3.1.

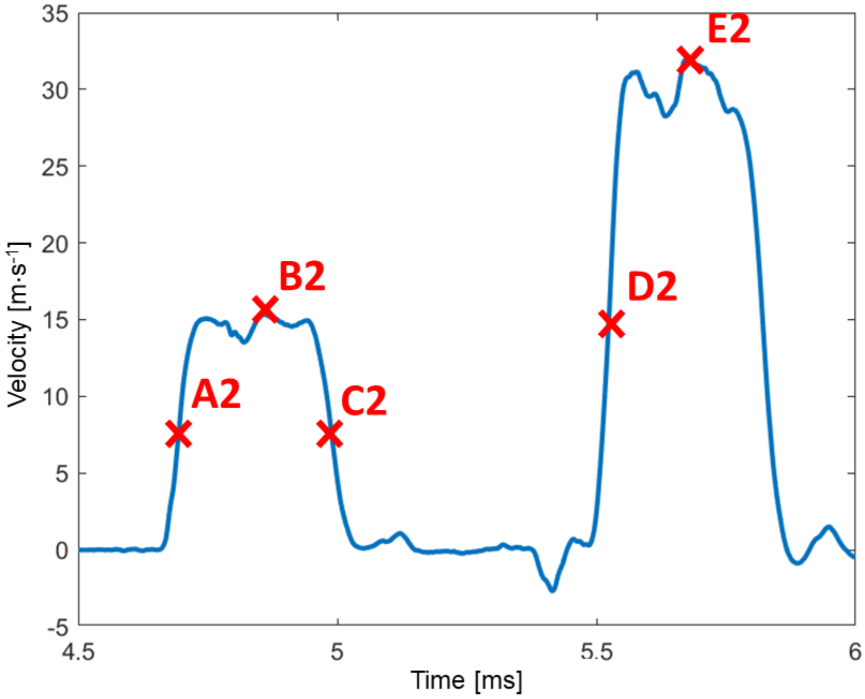
Several parameters were introduced by using points from Table 3.1. By quantifying these parameters, the accuracy and reliability of the image recognition and tracking methods used in this study can be assessed. In the following subsections, each parameter will be discussed individually.

The results are presented as absolute values to demonstrate the dimensions of the time duration or velocity used in this laboratory practice. In order to assess the accuracy of the method, we provide the ratio between the results obtained from selected method and the comparative strain gauge measurements, which are considered as the ground truth. This comparison allows us to quantify how well the image recognition and tracking methods perform in capturing the sample's movement, and provides a measure of the method's reliability and precision in relation to the reference measurements.

$$ratio = \frac{value(X)}{value(\text{Strain Gauge})} \tag{3.1}$$



(a) Incident bar



(b) Transmission bar

Figure 3.22: Points for the results quantification

Note: The point definitions can be found in Table 3.1

Point	Definition
A1	$\frac{1}{2}$ of the maximum value at B1
B1	Maximum value between A1 and C1
C1	$\frac{1}{2}$ of the maximum value at B1
A2	$\frac{1}{2}$ of the maximum value at B2
B2	Maximum value between A2 and C2
C2	$\frac{1}{2}$ of the maximum value at B2
D2	$\frac{1}{2}$ of the maximum value at E2
E2	Maximum value after D2

Table 3.1: Definition of the points in Figure 3.22

3.5.2 Peak Duration

The peak duration parameter compares the duration or *length* of the peak. For the purpose of this analysis, the peak is defined as the part of the curve between two points with their values equal to half of the maximum (which is located between these two points). In the context of the defined points from Table 3.1, it represents the difference in the x-values between C1 and A1 for the incident bar, and between C2 and A2 for the transmission bar. This parameter allows us to evaluate the time span during which the peak occurs. The values are placed in Table 3.2.

Bar/Method	Strain Gauge [ms]	Edge Detection [ms]	Matlab DIC [ms]	RaDIC [ms]
Incident	0.294	0.297	0.293	0.29
Transmission	0.293	0.293	0.297	0.293

Table 3.2: Peak duration – absolute values

Bar/Method	Strain Gauge	Edge Detection	Matlab DIC	RaDIC
Incident	1	1.009	0.998	0.986
Transmission	1	1.001	1.012	1.001

Table 3.3: Peak duration – strain gauge ratio

In this parameter, the edge detection approach exhibits an error of under 1% compared to the strain gauge measurements, indicating its high accuracy. The results are similarly accurate to the RaDIC approach, which is predominantly used as the image processing method. These very low errors demonstrate that all the methods are equally sensitive to changes in the velocity of the pattern, whether it is increasing or decreasing.

3.5.3 Peak Height

The peak height represents the velocity of the sample. In this thesis, both the maximal value within the peak interval and the average value are investigated. In laboratory practice, the average value holds greater significance. Additionally, the peak is defined in the same manner as in the previous case, considering points A1/A2 to C1/C2. The absolute values for the peak height are presented in Table 3.4, and the ratio values are shown in Table 3.5.

Bar/Method	Strain Gauge [ms]	Edge Detection [ms]	Matlab DIC [ms]	RaDIC [ms]
Incident (Max)	15.617	16.766	16.397	15.988
Incident (Avg.)	14.134	14.444	14.571	14.699
Transmission (Max)	15.346	16.399	16.397	15.744
Transmission (Avg.)	14.035	14.59	14.515	14.493

Table 3.4: Peak height – absolute values

Bar/Method	Strain Gauge	Edge Detection	Matlab DIC	RaDIC
Incident (Max)	1	1.074	1.050	1.024
Incident (Avg.)	1	1.022	1.031	1.040
Transmission (Max)	1	1.069	1.068	1.026
Transmission (Avg.)	1	1.040	1.034	1.033

Table 3.5: Peak height – strain gauge ratio

All the investigated approaches provide overestimated results in order of a few percent. The edge detection method has average values less overestimated than the maximum values, however, the RaDIC approach follows the oposite trend – the results for the average values are more overestimated than the maximum values. The overestimation is similar for both approaches. This fact was visually observed in previous sections, where it was noted that the unit conversion caused an inaccuracy due to only working with an integer number of pixels. However, the error is small enough to consider these results as similar to the strain gauge measurements.

3.5.4 Peak Distance

The second duration parameter focuses on the duration between the end of the first peak and the start of the second peak. This investigation is conducted only for the transmission bar data, as the second peak is not captured by the camera for the incident bar. Specifically, the distance is defined by points C2 and D2. The absolute values for this parameter, along with the ratios to the strain gauge values, are presented in Table 3.6.

Values/Method	Strain Gauge	Edge Detection	Matlab DIC	RaDIC
Absolute [ms]	0.538	0.543	0.537	0.533
Ratio [–]	1	1.01	0.998	0.991

Table 3.6: Peak distance for the transmission bar

Similary to the previous sections, the results obtained by the edge detection approach can be considered of very high quality in this dataset. The edge detection method provides results that are closest to the strain gauge values. Furthermore, as observed in the peak duration analysis, it can be postulated that the method is also equally sensitive to changes in the velocity of the pattern, even for two different peaks in this case.

Conclusion

The motivation behind this thesis was to develop an autonomous program capable of detecting randomly generated patterns, which are used for enabling image tracking during impact dynamics experiments. By automating the data evaluation process, the thesis effectively reduced the manual workload involved in analyzing the measurements.

This thesis has investigated the research of pattern detection in the context of laboratory practice. A software tool for pattern detection was prepared and developed, featuring a user-friendly GUI to operate the functions. The GUI also includes options to utilize different approaches for data processing, such as DIC in Matlab or RaDIC software.

In conclusion, the edge detection function demonstrated high accuracy, with the recognition of the pattern achieving an accuracy rate above 95 %. Although the function identifies approximately 75 % of the pattern pixels, it is important to note that the unrecognized pixels are mainly located in areas with poor lighting conditions, making them unsuitable for further image processing methods such as DIC. However, the recognized portion of the pattern is sufficient to meet the requirements of the potential image processing methods that may be employed.

A small semi-automatic script was introduced to perform the unit conversion, converting the pixel length to the standard metric length. When combined with knowledge of the camera's FPS settings, extracted automatically from the camera's log file, this enabled the display of velocity in meters per second over time in seconds. The converted data was then compared with the strain gauge measurements. The results of this comparative analysis revealed that the edge detection method's results closely corresponded with the image-independent data. The differences in the results were minimal, with variations of approximately a few percent, most of which remained below 2 %. Furthermore, when comparing the edge detection method with other image processing approaches such as DIC in Matlab and RaDIC software, the results demonstrated striking similarity, affirming the accuracy and reliability of the edge detection method in this context.

In the future, the method can be further improved by leveraging some of Matlab's toolboxes that offer parallel computing or computation on graphics cards. This could potentially enhance the computational efficiency and speed of the image processing tasks. Additionally, a key area for improvement lies in addressing the program's behavior when the pattern is lost due to the presence of sample debris.

Generally, the developed software has proven to be highly efficient in evaluating image data from the experiments. The automation of the data evaluation process significantly minimizes the manual workload, as most tasks are carried out automatically without requiring extensive user input. With its ability to compare results obtained from various image processing approaches, the software enables a comprehensive review of the accuracy and reliability of the outcomes. The incorporation of multiple approaches further enhances the redundancy of the results, ensuring a robust assessment of the experiment's data.

Overall, the development and refinement of this image recognition method hold great potential for advancing mechanics applications and contributing to the field of experimental mechanics.

Bibliography

[1] J. Wong, L. Ryan, and I. Y. Kim. Design optimization of aircraft landing gear assembly under dynamic loading. *Structural and Multidisciplinary Optimization*, 57(3):1357–1375, Mar 2018.

[2] John Field, S. Walley, N. Bourne, and J. Huntley. Experimental methods at high rates of strain. *Journal de Physique IV Proceedings*, 04(C8):C8–3–C8–22, 1994.

[3] John Hopkinson et al. On the rupture of iron wire by a blow. *Proc. Literary and Philosophical Society of Manchester*, 1:40–45, 1872.

[4] Q. B. Zhang and J. Zhao. A review of dynamic experimental techniques and mechanical behaviour of rock materials. *Rock Mechanics and Rock Engineering*, 47(4):1411–1478, Jul 2014.

[5] Artur Iluk. Using the high-speed camera as measurement device in the dynamic material tests. *Journal of Vibroengineering*, 14:22–26, 03 2012.

[6] Timothy E. Schmidt, John Tyson, Konstantin Galanulis, Duane M. Revilock, and Matthew E. Melis. Full-field dynamic deformation and strain measurements using high-speed digital cameras. In Dennis L. Paisley, Stuart Kleinfelder, Donald R. Snyder, and Brian J. Thompson, editors, *26th International Congress on High-Speed Photography and Photonics*, volume 5580, pages 174 – 185. International Society for Optics and Photonics, SPIE, 2005.

[7] S. Boukhtache, K. Abdelouahab, F. Berry, B. Blaysat, M. Grédiac, and F. Sur. When deep learning meets digital image correlation. *Optics and Lasers in Engineering*, 136:106308, 2021.

[8] Amir Rezaie, Radhakrishna Achanta, Michele Godio, and Katrin Beyer. Comparison of crack segmentation using digital image correlation measurements and deep learning. *Construction and Building Materials*, 261:120474, 2020.

[9] Guodong Sun, Wei Feng, Daxing Zhao, and Linjie Yang. An automatic fault recognition method for side frame key in tfds, Feb 2015.

[10] Introduction to image compression. In AL BOVIK, editor, *Handbook of Image and Video Processing (Second Edition)*, Communications, Networking and Multimedia, page 641. Academic Press, Burlington, second edition edition, 2005.

[11] Inc. The MathWorks. *Image Processing Toolbox.* Natick, Massachusetts, United State, 2022.

[12] The MathWorks Inc. Matlab version: 9.13.0 (r2022b), 2022.

[13] Inc. The MathWorks. Strel - morphological structuring element. `https://www.mathworks.com/help/images/ref/strel.html`, 2023. [Online; accessed 01-July-2023].

[14] Inc. The MathWorks. Imbinarize - binarize 2-d grayscale image. `https://www.mathworks.com/help/images/ref/imbinarize.html`, 2023. [Online; accessed 01-July-2023].

[15] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

[16] Inc. The MathWorks. Imclose - morphologically close image. `https://www.mathworks.com/help/images/ref/imclose.html`, 2023. [Online; accessed 01-July-2023].

[17] Inc. The MathWorks. Bwpropfilt - extract objects from binary image using properties. `https://www.mathworks.com/help/images/ref/bwpropfilt.html`, 2023. [Online; accessed 01-July-2023].

[18] Michael A. Sutton, Jean-Jos Orteu, and Hubert Schreier. *Image Correlation for Shape, Motion and Deformation Measurements: Basic Concepts,Theory and Applications.* Springer Publishing Company, Incorporated, 1st edition, 2009.

[19] J.P. Lewis. Fast normalized cross-correlation. *Vision Interface*, 10:120 – 123, 10 1995.

[20] Tomáš Fíla, Petr Koudelka, Jan Falta, Petr Zlámal, Václav Rada, Marcel Adorna, Stefan Bronder, and Ondřej Jiroušek. Dynamic impact testing of cellular solids and lattice structures: Application of two-sided direct impact hopkinson bar. *International Journal of Impact Engineering*, 148:103767, 2021.

[21] Michaela Neuhäuserová, Tomáš Fíla, Petr Koudelka, Jan Falta, Václav Rada, Jan Šleichrt, Petr Zlámal, and Ondřej Jiroušek. Compressive behaviour of additively manufactured periodical re-entrant tetrakaidecahedral lattices at low and high strain-rates. *Metals*, 11(8):1196, Jul 2021.

[22] Inc. The MathWorks. Imshowpair - compare differences between images. `https://www.mathworks.com/help/images/ref/imshowpair.html`, 2023. [Online; accessed 01-July-2023].

List of Figures

List of Tables

List of Abbreviations

Avg.	Average
DIC	Digital Image Correlation
FPS	Frames per second
MIPT	Matlab Image Processing Toolbox
mps	Meters per second
ROI	Region of interest
SHPB	Split Hopkinson Pressure Bar
SW	Software