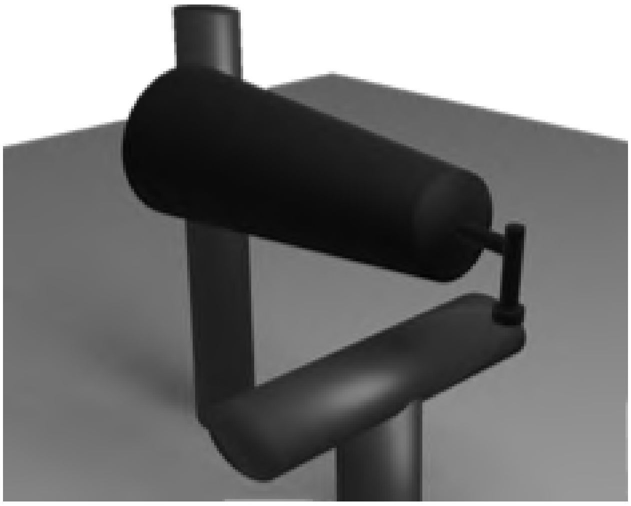


Characterization of contact interaction with an inflatable link robot

Margarida Fernandes Martinho

Thesis to obtain the Master of Science Degree in

Mechanical Engineering

Supervisor: Prof. João Carlos Prata dos Reis

Examination Committee

Chairperson: Prof. Carlos Baptista Cardeira
Supervisor: Prof. João Carlos Prata dos Reis
Member of the Committee: Prof. Alexandra Moutinho

July 2021

Acknowledgments

I would like to thank all of the people that contributed for the making of this thesis, one way or another.

Firstly, to my dissertation supervisor Prof. João Reis for the insight, support and especially patience, and always being available to help and advise me.

To my IST colleagues, who pushed me to learn, grow and persevere, for the friendship and companionship built over this era.

Also, to all my friends, for always being there for me, particularly in these pandemic times.

And finally to my family, who always supported me every step of the path I chose, with encouragement and care.

To each and every one of you - Thank you.

Abstract

This work comprises a virtual simulation study of position and contact estimation on an inflatable structure, using computer vision and machine learning tools.

The project developed is meant to be applied and validated on a prototype that will function as an additional link implemented on a commercial rigid robot manipulator. It builds on previous work, upgrading the internal pattern and the optical sensor used but not the structure geometry.

The simulation environment was built employing the Blender software, with two different optical sensors: a single camera and stereo camera set.

The images retrieved were submitted to an image segmentation algorithm to extract relevant features to be used as inputs to an artificial neural network that returns the bending curve and tip displacement. For the contact estimation, after the image segmentation step, an empirical algorithm, using the principal inertia moments information, computes the contact point and the direction.

The performance of the bending estimation algorithm was very satisfactory for both single view and stereo view. As for the contact estimation, initial tests were promising but further study is required to fully establish the method.

Keywords

Soft Robotics, Proprioception, Computer Vision, Artificial Neural Networks

Resumo

Esta dissertação consiste num estudo de simulação virtual da estimativa de posição e contato numa estrutura insuflável, utilizando ferramentas de visão computacional e *machine learning*.

O trabalho desenvolvido pretende ser aplicado e validado num protótipo que complementa um robô industrial, funcionando como um elo adicional com as características de uma estrutura insuflável. É baseado num projecto anterior, mudando o padrão interno e o sensor óptico usado, mas não a geometria da estrutura.

O ambiente de simulação foi construído recorrendo ao software Blender, com duas opções de sensor óptico: uma única câmera e um conjunto *stereo*.

As imagens obtidas foram processadas num algoritmo de segmentação de imagens para extrair características relevantes a serem utilizadas como *input* para uma rede neuronal artificial que devolve a curva de flexão e o deslocamento da ponta. Para o problema de contato, após a etapa de segmentação da imagem, o ponto de contato e a sua direção são inferidos utilizando um algoritmo empírico, através de informação dos momentos de inércia no eixo principal.

O desempenho do algoritmo de estimativa de flexão foi bastante satisfatório tanto para imagens centradas, quanto para visão *stereo*. Quanto à estimativa de contato, os testes iniciais foram promissores, mas é necessário realizar mais estudos para estabelecer o método por completo.

Palavras Chave

Robótica Macia, Propriocepção, Visão Computacional, Redes Neuronais Artificiais

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms

ANN	Artificial Neural Network
CLAHE	Contrast-Limited Adaptive Histogram Equalization
EE	Extremity Error
LMB	Levenberg-Marquardt Backpropagation
MSE	Mean Squared Error
NN	Neural Networks
ReLu	Rectified Linear Units
RMSE	Root Mean Squared Error
RNN	Recurring Neural Network
SSE	Sum of Squared Errors
TACTIP	Tactile Fingertip

1

Introduction

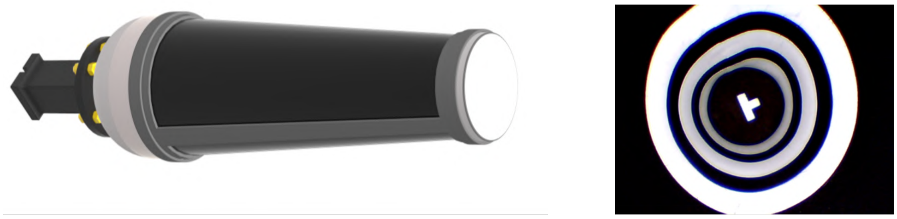
Contents

1.1 Motivation

Soft robotics is a innovative topic, redefining the terms for human and robot interaction and so, has been the subject of numerous investigation works in the last decade [1].

Soft robots have a very significant advantage over traditional robots: they are intrinsically safe to interact with people or fragile objects due to their lightweight, compliant and flexible nature. This way, there is a lot of research to improve their control, enabling them to work efficiently outside monitored environments, and to integrate them in traditional robotic manipulators. One way of achieving that is to enhance the soft robot proprioception, with tools that help to understand better their internal state.

This project builds on previous work with inflatable links [2] [3], in particular [4], in which an inflatable link prototype was created, with a built in camera centered at the base, and the interior images were used to estimate the force applied at the distal end of the link, with and without rotation. The developed algorithm proved to be capable of estimating accurately the applied force, in multiple directions, using blob analysis to evaluate deformations in the internal pattern and processing that information with neural networks.



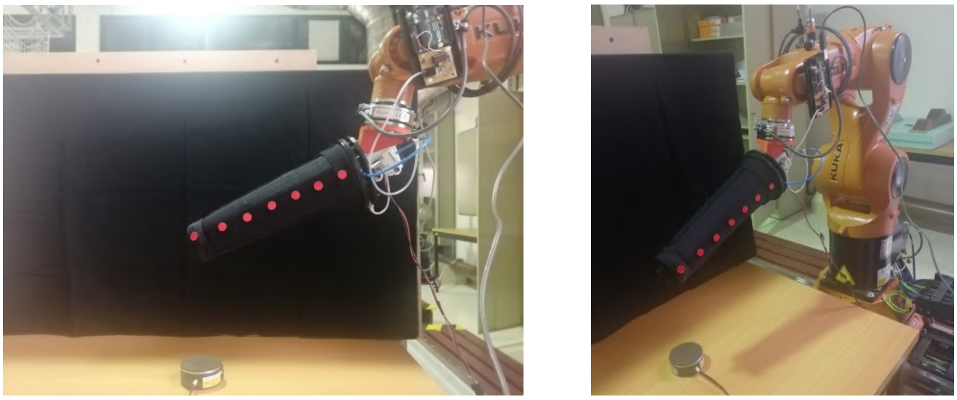
(a) Assembled Prototype in CAD Software **(b)** Interior Pattern

source: Article [4]

Figure 1.1: Previous Work Prototype

The present dissertation aims to improve and expand the previous work. The improvements comprise the change of the interior pattern, increasing the number of markers, and the image segmentation method used. The expansion occurs at the link proprioceptive functionalities, estimating the bending curve and displacement of the tip and identifying contact situations, with contact point and direction, which can eventually be used to design an evasion route. Another important aspect is the camera setup that will be used to retrieve the interior images since there is the intent of changing the sensor to a camera with a wider field of view and higher frame rate. The new sensor contains a stereo camera set, so this dissertation must also verify if every algorithm functions for stereo images. Since the simulations will take place in a virtual environment, there will be no estimation of force, this is to be validated in an experimental context, in following works.

The work developed is meant to be applied and validated on a prototype to be integrated as an additional link in a traditional robot, as shown in figure 1.2.



(a) Prototype for Validation

(b) Integration in traditional Robot

Figure 1.2: Future Work Prototype

The idea behind attaching an inflatable link to a robot manipulator is to have the predictability and controllability of a hard robot, while making sure the interaction with the environment happens safely, with the soft components. If the integration is verified and considered viable, the robot with the inflatable link might be capable of performing tasks safely, without human intervention, for example, in the biomedical engineering field, ultrasound robot assisted interventions, with the probe being held at the distal end of the link.

1.2 State of the Art

1.2.1 Soft Robotics

Soft Robotics is a field of science that borrows the mechanics of real biology. It allows for more freedom of movement, compared to traditional robotics, adding degrees of freedom to the robot, but that also makes it much harder to control. It is a rapidly improving field, subject to a significant amount of investigation since it offers plenty of advantages, like working in cluttered places or allowing for safer interactions with living beings [5].

In 2008, the term ”soft robotics” started to be used to refer to, not only rigid robots with compliant joints, but also to robots built from soft materials [6]. This allied with the development of a robotic arm inspired by an octopus tentacle [7], in 2009, started a boom in the investigation of soft robotics [1] and their potential applications to other fields, one being biomedical engineering.

The applications of soft robotics in engineering are very vast, since flexibility, freedom of movement and volume variance are extremely useful features in this context, not to mention the materials and production are usually cheaper. Soft robots can be now used as surgical tools, assistance devices or even as artificial organs [8]. There are numerous types of soft robots, reaching from grippers [9] to inflatable links [10].

Inflatable structures use internal pressure to maintain structural integrity [11]. They can be air supported - usually open structures that are continuously refilled - or air inflated - airtight structures inflated to any desired pressure, superior to the external. Inflatable links fit in the last category. Inflatable links have been used in the robotic domain to lower the contact stiffness of traditional robots, either as inflatable sleeves for a rigid robotic skeleton [12], or as replacement robotic links [13]. In addition to being safe for contact situations, inflatable links also have other advantages, such as being light, having distributed compliance, good performance post-buckling and low cost of fabrication.

It is also important to mention that the investigation on soft robotics does not aim to completely replace the traditional robot manipulators, but to associate them. In fact, there are multiple studies on how they can be used together, using the predictability and efficiency of robot manipulators and the compliance and flexibility of soft robots to create a safer work environment [14] [15].

As mentioned before, the significant disadvantage the soft robots have is the difficulty in control. As such, they rely heavily on sensors [16], to get information on the system and to develop control strategies, for example in supervising the position.

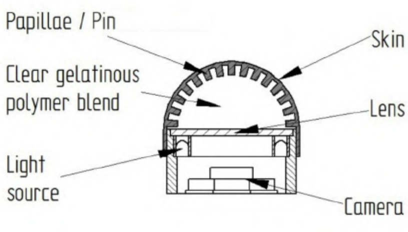
1.2.2 Tactile Sensing

Tactile sensors can be of diverse types - resistive, capacitive, optical, magnetic, inductive - but they should always provide information on three major fronts [17]:

- Response - escape in case of eminent collision [18]
- Recognition - identify objects through texture [19] , geometry [20] and other properties
- Control - estimate force [21] or contact location [22]

Optical sensors are most often cameras. They are a very popular sensor choice because of their high spatial resolution, adaptable frequency response and the fact that they are not affected by electromagnetic interferences. Their most significant setback is the size, stiffness and high processing power [23], but the first two are only a problem if the sensor is to be used in a considerably small robot and connected to its surface, the third is always a factor to consider.

The human touch sense is always preferable for tasks like object manipulation, contact sensing, pressure sensing and shear force detection [24]. Tactile sensing is evolving to try to recreate the human touch sensibility in soft robots. There are very interesting investigation works on this topic, using optical sensors, particularly the Tactile Fingertip (TACTIP) device. The TACTIP device is inspired by the anatomy of the human finger, using internal pins, mimicking epidermal papillae, to assess the surface deformation [25]. A schematic representation of the TACTIP is in figure 1.3



source: Article [25]

Figure 1.3: Cross section of the device design

This technology has been transferred to other geometries, to accommodate different needs [26]. One of particular interest based on the scope of this project is the TacCylinder. In Winstone et al. [27], there is a detailed explanation of how to use markers on a cylindrical surface (TacCylinder) to characterize (in size, shape and hardness) the objects that come into contact with it. A camera was place by one of the ends of the prototype and aligned with its central axis.

The knowledge of the position and deformation of the soft surface allows recreating the environment of interaction. In that way, it is important to create and improve the tools to comprehend the internal state of the object.

1.2.3 Proprioception

Proprioception is the understanding of the body of its internal state, it means perception of own, and is a very relevant topic in soft robotics, both to increase the knowledge of the soft robots themselves (analysing their behaviour and their responses to the environment) and to characterize, indirectly, their surroundings [28]. Proprioception is an area of undeniable importance for it is crucial to have reliable information from the robot itself if it is to be used outside of laboratory or other environments that allow monitoring.

Research on proprioception might aim to track the position of a single point, usually the extremity [29], or to be aware of the deformation of the entire surface [30].

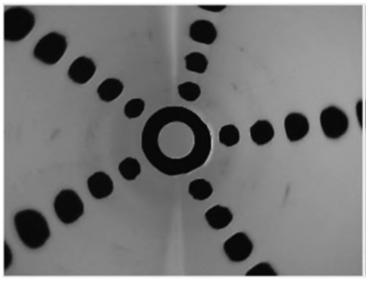
In [31] an array of optical fibers and, consequently, reflection of light is used to recognise bending and torsion in an elastomeric foam. The deformation detection tested two machine learning approaches:

a combination of a single-output classification and a single-output regression, the first predicting the movement type and the second the magnitude, and a multi-output regression, capable of simultaneously estimating bending and twisting.

The work described in [32] uses an optical sensor to control the pressure and, thus, the stiffness of a membrane. The optical sensor captures the displacement of the 3D markers to assess the current situation of the surface, estimating contact condition or detecting membrane shear.

Vision based sensing systems are particularly useful when dealing with inflatable link types, as they provide understanding on their internal state, without having to be attached to the soft surface, facilitating the creation of models and not increasing the stiffness in a local point [33].

The authors of [34] present a method to measure the tip position of an inflatable link and binary contact detection, also using a built-in camera. Here, the internal pattern consists of ellipse-shaped markers, in figure 1.4, and the position and contact detection are measure using blob analysis and geometry.

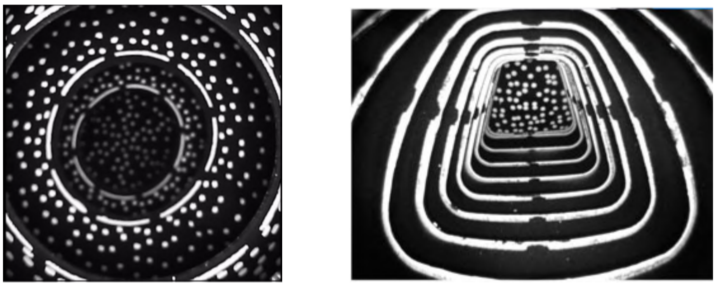


source: Article [34]

Figure 1.4: Internal Pattern of Inflatable Link

In Werner et al. [29], it is described the process of estimation of the three dimensional tip position of an inflatable actuator, using images from an integrated centered camera. The pattern applied, figure 1.5(a), uses scattered $2\ mm$ diameter dots to gather bending information according to the light density of the image.

The work presented in [33] is a continuation of [29], using the same kind of interior pattern, figure 1.5(b), not for one, but three actuators. The position of the tip is not directly measured, instead the rotational degrees of freedom of the arm are estimated using deep neural networks. As each actuator has a built-in camera, the work also suggests a sensorial fusion approach.



(a) source: Article [29] **(b)** source: Article [33]

Figure 1.5: Internal Patterns of the Inflatable Actuators

It can be extrapolated that proprioception via vision based systems usually uses one of two methods: light intensity analysis, a large number of small markers to be analysed in terms of distribution [29], or marker geometry analysis, a small number of markers where each one provides information by itself and within the group [34].

The literature described above was deliberately focused on vision based systems used for proprioceptive soft robots, with an emphasis on inflatable link types since that will be the main focus of the work.

1.3 Thesis Outline and Contributions

This section describes the organization of the present document and the contributions it brings to the state of the art.

The main contribution of this thesis is the methodology of finding the position and contact through internal pattern deformations. There are, in fact, other contributions related to the developed work in the application of computer vision and machine learning departments, in particular:

- A simulation environment for the inflatable link with built in camera, or cameras, suited to the development and validation of proprioceptive image processing algorithms;
- Design of a pattern that gathers information of both bending and contact;
- A set of algorithms and training methodologies for the neural networks to be applied in the robot prototype;
- Validation of the proposed algorithms in the simulation environment.

Following the first introductory chapter, chapter two will go over the theory elements used throughout the work, with particular emphasis on image processing and neural networks.

The simulation environment is described in chapter three, as well as the construction of the virtual inflatable link and all its integrated parts.

In chapter four, the main topic is image processing, where the manner of treating the images from simulation and the experimental work is detailed.

Chapter five presents the results and discussion of the developed project.

Finally, in chapter six, a conclusion is drawn upon this project and future work to improve its limitations or expand its functionalities is suggested.

2

Theoretical Background

Contents

This chapter will focus on the theoretical knowledge that is required prior to elaborate and analyse this project. It is given special attention to the fields of image processing and neural networks, as they are the two pillars of the work developed.

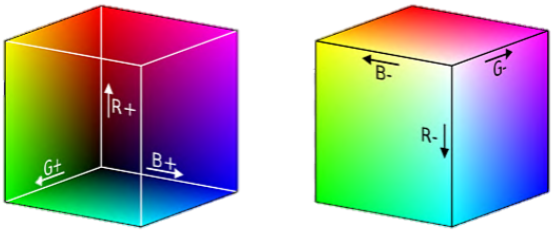
2.1 Computational Vision

Computational Vision is the field of science that deals with image processing. Identifying objects in an image, change colors, contrast, saturation; all of these operations are in the computational vision realm.

In agreement with the scope of this project, the core of this section will be shifted towards the image segmentation department.

2.1.1 color Spaces

There are several ways to organize the color information of an image, known as color spaces. One of the most commonly known is the RGB color space, figure 2.1, in which all images are $[m \times n \times 3]$ objects, they have three separate channels, each one representing red, green and blue, in that order. That means all colors are a combination of red, green and blue, with [255,255,255] being white and [0,0,0] being black.



source: PNG EGG[1]

Figure 2.1: RGB color space

Other color spaces organize the information in a different way, separating color information from intensity. Examples of this are the HUV and YIQ color spaces, that organize a colored image in hue, saturation and value and in *luma*, in-phase and quadrature, respectively, with value and *luma* representing the intensity information.

[1] https://www.pngegg.com/en/png-nolzf

2.1.2 Image Segmentation

Many times it is useful to identify and label areas of an image, for example when detecting a specific object. There are various segmentation methods, some of which are discussed in detail in this section.

2.1.2.A Binarization

In order to have a binary image, the original has to be transformed into a grayscale image and then an intensity threshold is applied to the entire matrix. Each value above the threshold will turn into 1, and each value below will turn into 0. Binary images are very useful to evaluate objects in a picture geometry wise.

Grayscale images are pictures without color information. The objects are pictured in various shades of grey, corresponding to the lightness they had originally. They have only one channel, carrying information of intensity instead of color. Notice that a grayscale image is not a black-and-white image, as the previous has many shades of gray.

The most common way to convert a RGB image to grayscale is a weighted sum of the information of the three channels, which guarantees the result will balanced and will not accentuate one color over the other. These weights come from the convertion of RGB image to the YIQ color space[2] (2.1).

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0,299 & 0,587 & 0,114 \\ 0.596 & -0.274 & -0.322 \\ 0.211 & -0.523 & 0.312 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{2.1}$$

The grayscaled image will correspond to the Y - luma - channel matrix (2.2), which gathers the brightness information of the original image.

$$Y = 0,299R + 0,587G + 0,114B \tag{2.2}$$

A binary image is a logical matrix, each pixel can only assume two values: 0 and 1, which will correspond to black and white, respectively.

There are several ways to find a suitable threshold for binarization, one of the most common being the Otsu's method [35], which is an iterative process that aims to find the value that minimizes the intra-class variance of foreground (white) and background (black). The photo ”Lena”[3] will be used for the demonstrative purpose of explaining the different parts of the binarization process, starting with a simple binarization without previous corrections, in figure 2.2.

[2] https://www.mathworks.com/help/images/ref/rgb2ntsc.html

[3] https://homepages.cae.wisc.edu/~ece533/images/

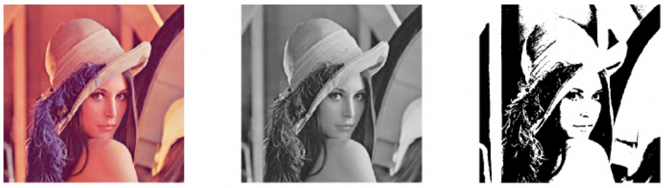


Figure 2.2: Process of Binarization

There are cases where the picture needs to be corrected so that important features are not lost in the process of binarization. This correction can often be at level of the contrast and there are many tools for it. One that is particularly useful is the histogram.

The histogram of an image plots the number of pixels by the exact shade they are. By looking at one, it is possible to understand the distribution of color in the image. Having this understanding, it is possible to change the distribution of color to better suit the needs of the task.

In figure 2.3, below, several changes to the histogram of a grayscale image are portrayed with the corresponding image.

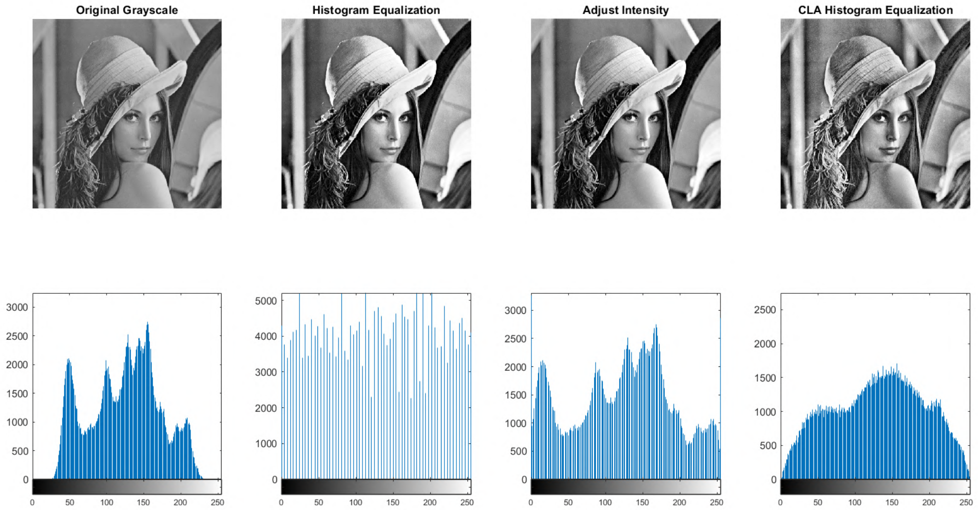


Figure 2.3: Contrast Correction - Histogram Perspective

The first column displays the original grayscale picture and the respective histogram. The second shows the effect of applying an histogram equalization, which will roughly translate to spreading out the most frequent values of intensity. In the third column is pictured the effect of adjusting the range of intensity of the image, that is, stretching the histogram to saturate the top and bottom 1% at 255 and 0, respectively. The last column corresponds to a Contrast-Limited Adaptive Histogram Equalization (CLAHE), where the image is divided into 8 by 8 tiles and an histogram equalization is applied to each, while making sure to smooth the transitions between neighbouring tiles (with a bilinear interpolation) [36].

All of these methods enhance the contrast of the picture to several degrees of success, it depends on the desired task which will be the most suitable.

2.1.2.B Blob Analysis

In computer vision, a blob is defined as a group of pixels with the same characteristics, namely color. In a binary image, a blob will be each connected group of white pixels, as the black pixels are perceived as background.

When a blob is labeled, there is plenty of information that can be obtained regarding its geometry, such as area, centroid and perimeter but also some non trivial properties like eccentricity or convex hull.

Area

Defined as the number of pixels in the blob.

Centroid

Very relevant feature because it allows the tracking of the position a blob through time. It is computed as follows:

$$C_x = \frac{\sum_{i=1}^{n} X_i}{n} \tag{2.3}$$

$$C_y = \frac{\sum_{i=1}^{n} Y_i}{n} \tag{2.4}$$

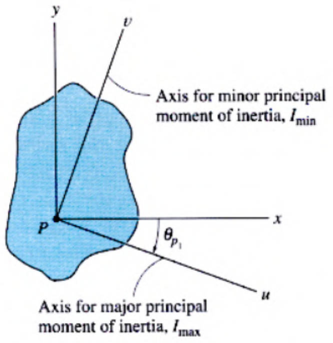
with C_x and C_y as the centroid coordinates in the x and y axis, respectively, and X_i and Y_i as the coordinates in the x and y axis of each individual blob point i out of the total number of blob points n.

Area Moment of Inertia

Henceforth, every mention of the moment of inertia will be referring to the area moment of inertia, not to be confused with the mass moment of inertia.

The second moment of area, or area moment of inertia, is a geometric property that measures the area distribution of an object relative to an arbitrary axis.

Many times, it is useful to know the moment of inertia described in the object's own reference frame (with the origin placed on its centroid) and aligning the axis system with the principal axes. These are where the moments of inertia will have the maximum and minimum values, and the product of inertia will be zero. Figure 2.4 presents the principal axes and the orientation relative to the typical x and y reference frame.


y
v
Axis for minor principal moment of inertia, I_{min}
x
P
θ_{p_1}
u
Axis for major principal moment of inertia, I_{max}

source: Lecture Notes MST University[4]

Figure 2.4: Principal Axes of Inertia

The principal moments of inertia and the orientation can be computed using the ensuing equations:

$$I_{\substack{max\\min}} = \frac{I_x + I_y}{2} \pm \sqrt{\frac{I_x - I_y}{2}^{2} + I_{xy}^2} \tag{2.5}$$

$$tan(2\theta_p) = \frac{-I_{xy}}{(I_x - I_y)/2} \tag{2.6}$$

Alternatively, it is possible to take the linear algebra approach and compute I_{min} and I_{max} using the eigenvalues of the matrix M depicted in equation 2.7.

$$\boldsymbol{M} = \begin{bmatrix} I_{xx} & -I_{xy} \\ -I_{xy} & I_{yy} \end{bmatrix} \tag{2.7}$$

The value of θ can be calculated through the eigenvectors of matrix M, by applying the inverse of the tangent.

The principal axes can be easily identify in symmetric shapes, as a symmetry axis represents a principal one.

For the example of an ellipse, the major and minor axis will be the principal axes of inertia and the respective moments will be I_{min} and I_{max}. The closer the lengths of the major and minor axes are, the more similar the values of I_{min} and I_{max} will be. For the case a circle, it is evident that $I_{min} = I_{max}$.

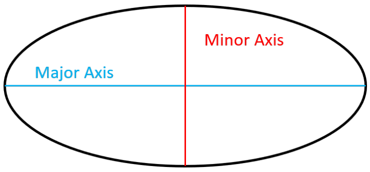


Figure 2.5: Major and Minor Axes of Ellipse

[4] https://classes.mst.edu/civeng50/lessons/28/principle_axes/index.html

2.1.2.C Boundary Analysis

There are cases where the analysis of the entire blob is not sustainable, either computation power or time-wise. There are some ways to avoid evaluating the blob in its entirety, for example, by analysing the boundary instead.

One form of tracing the boundary of a blob is the Moore-Neighbor algorithm. It consists of finding a black pixel and then evaluating its neighbourhood, figure 2.6, usually in a clockwise direction, to find another black pixel, that is saved as a new part of the boundary, and so on. The algorithm stops adding new pixels to the boundary once it reaches the first pixel for a second time [37].

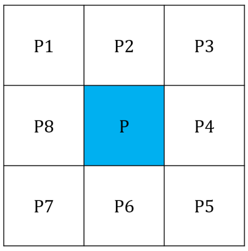


Figure 2.6: Moore Neighbourhood

There were some limitations of this method, especially concerning the stopping condition, which was improved by Jacob Eliosoff [37], when he proposed to terminate the boundary at the point in which the first pixel was reached again from the same direction as the first time.

The explained process is known as the Moore-Neighbor tracing algorithm modified by Jacob's stopping criteria.

Once the boundary is identified, it is possible to find the same geometric properties as for the blob analysis, only considering the boundary corresponds to a completely filled in shape. For the case where the blobs have holes, the boundary approach will never correspond exactly to the geometry of the blob, but it might be an adequate alternative.

2.2 Artificial Neural Networks

Artificial Neural Network (ANN), or simply Neural Networks (NN), are machine learning tools that replicate the learning process of biological neural networks through logical operations.

2.2.1 Architecture

Every ANN is composed of three elements, artificial neurons, connections and a propagation function. The neurons, or nodes, can receive multiple inputs and compute one output that is transmitted to multiple nodes ahead. The first nodes receive an external input and the last node produces the output of the NN.

The process of finding the output of the nodes is pictured in figure 2.7. It starts by associating weights to each input and adding them, which corresponds to the propagation function. A bias value can also be added to this sum. After this, the weighted sum must pass through an activation function that will assess if the value is significant enough for the node to be active. This can happen via a step function of more complex ones, like sigmoid or Rectified Linear Units (ReLu) functions. The activation function introduces a non-linear element to an otherwise linear process.

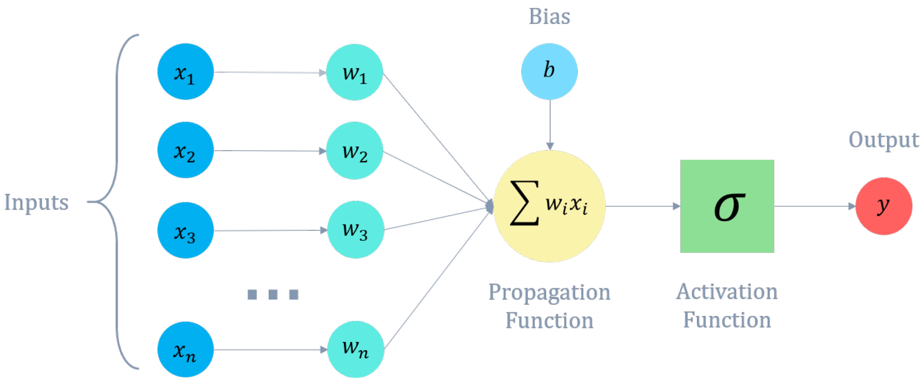


Figure 2.7: Artificial Neuron

A simple NN is organized by three layers: the input layer, the output layer and the hidden layer. As the NN grows in complexity, the number of hidden layers can increase. Note that a greater number of hidden layer does not necessarily mean the performance will be better. In figure 2.8, an example of a network is displayed, with each type of layer clearly identified.

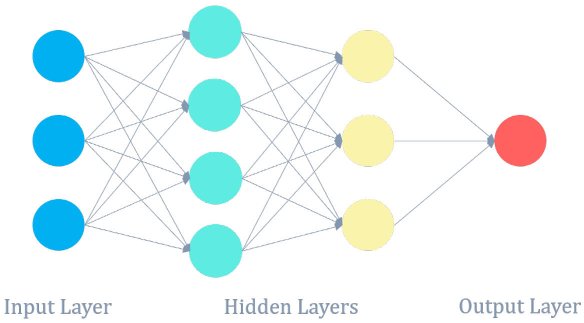


Figure 2.8: Structure of a Neural Network

As it can be observed in figure 2.8, each layer is composed of circles, which represent the artificial neurons. The arrows between layers represent connections with associated weights, that are adjusted through training. Naturally, the greater the weight value, the more important that connection is to the network performance.

The connections the network needs might differ according to the type of problem it is designed for. In fact, if a NN is meant to be used for classification and regression problems, a Feedforward Neural

Network might be the obvious choice, as it only needs to input information to find a fitting result, once the function is mapped for the specific task. On the other hand, if the task hangs on pattern recognition, like speech or handwriting, it is useful to make use of the internal state of the NN, having a notion of memory, while in the process. This is accomplished by using feedback connections, passing the information of later nodes to previous ones. These networks are called Recurring Neural Network (RNN).

2.2.2 Learning Paradigms

The learning ability is probably the most important aspect of an ANN, so it is capital to understand how the process happens.

The term ”learning” is used with the meaning of improving the performance of the network as the process continues. There are different ways the network can make used of the data to learn how to perform the task required, these approaches are called Learning Paradigms.

There are three learning paradigms used for NN, Supervised Learning, Unsupervised Learning and Reinforced Learning.

Supervised Learning is when the NN builds a set of rules to predict the result of a certain input, based on examples of input-output pairs, called the training set. The NN is submitted to a training process, which is explained in the subsequent section.
It is usually used in the scope of classification or regression problems.

Unsupervised Learning is frequently used as a first step before supervised learning. It detects patterns in the data set provided without human intervention and organizes the data into classes based on the common features found.
It is suitable for organizing into clusters and compressing data.

Reinforced Learning does not need a set of data to learn from, either labelled or not. It works via interaction with the model of the dynamic system, going through the process by trial and error and evaluating the responses of the environment to an action.
An agent is created to send an action to the environment and receive an observation and a reward (measures the degree of success). The goal of the learning process is to maximize the cumulative reward.
It is employed when dealing with typical stochastic problems, like autonomous driving or natural source management.

2.2.3 Training

The training of the NN consists of modifying the weights of the connections of the nodes based on the overall performance error until there is no significant change in the result. It is an iterative process grounded in estimation and optimization theory.

It is essential for the training process to distribute all the available data into three categories: train, validation and test. The training set is the one used to update the weights of the NN, the testing set is the one evaluating its performance and the validating set serves the purpose of avoiding overfitting of the data. There is to be no overlapping of these sets.

There are many training algorithms to guide this process, with several specialization areas, the most common being the Levenberg-Marquardt Backpropagation (LMB).

The Levenberg-Marquardt algorithm finds the minimum value of a function within an assessed region. The application of this algorithm to backpropagation results in a method of detecting the minimum value of the error.

LMB assumes that the performance measure is a variation of the sum of squared errors, so the network that uses this algorithm must evaluate the performance with either Mean Squared Error (MSE) ou Sum of Squared Errors (SSE). The steps of the algorithm [38] are subsequently explained.

1. Apply all inputs to the NN (with the randomly assigned weights) and compute the performance.

2. Compute the Jacobian matrix, employing (2.8).

$$[J]_{i,j} = \frac{\delta e_i}{\delta w_j} \tag{2.8}$$

where J is the jacobian matrix, e_i is the error vector with i iterations and w_j is the weigth value for connection j.

3. Find the new weight values, through (2.9)

$$w_{i+1} = w_i - [J^T J + \mu I]^{-1} J^T e_i \tag{2.9}$$

where μ is a damping factor.

4. Recompute performance of NN using new weights.

- if the new performance decreases, continue from 1. with a smaller value of μ;
- if the new performance increases, continue from 3. with a greater value of μ.

The algorithm is considered to have attained convergence when the gradient of the performance function is smaller than a user-predefined number.

LMB is very popular due to its speed, being often the fasted method of backpropagation, which makes a significant advantage in a process as repetitive as NN training. It is not always the best algorithm to use, as it can detect a local minimum of the error instead of the absolute, but it is a recommended first choice.

3

Simulation Environment

Contents

In this chapter, the simulation environment in the animation software will be described, as well as the steps of construction of the virtual inflatable link. The simulation to be executed is meant to be recreated experimentally. Accordingly, every aspect of the simulation must be analysed, keeping in mind that it has to be practical to be reproduced in the real world.

3.1 Simulation Environment - Blender

The software chosen for the simulations was Blender. Although Blender is primarily an animation software, it has been gradually making its presence known in the robotic simulation world [39]. Its physical properties tab has been improving substantially in the last years, being now able to emulate the behaviours of fluids, soft bodies and cloth as well as rigid bodies and interactions between them.

It is to be noted that Blender is not intended to achieve the physical accuracy of finite element analysis. It focuses primarily on the produced image, the camera setup and properties, to present a visually realistic take on the physical behaviour of the created objects, even if the process is not simulated with the most precision. The software chosen for this project must produce a visually realistic behaviour of an inflatable link and enable customization of camera properties. This way, Blender is more suitable than a finite element analysis software.

The construction of the virtual link for the software simulation was made taking into account the tasks it would perform in reality. The virtual link must behave like a beam when in a bending motion. Therefore, it is necessary to know how to cause this effect. The option chosen was to apply a displacement at the extremity of the link, keeping the base at the same place - a fixed beam with a free end.

To simplify the problem, it was decided not to define two different materials for inflatable sleeve and cover and instead, merge them into a single one. That will not change the simulation as long as the cover surface is guaranteed to be rigid, which blender does by pinning the relative positions of the face vertices.

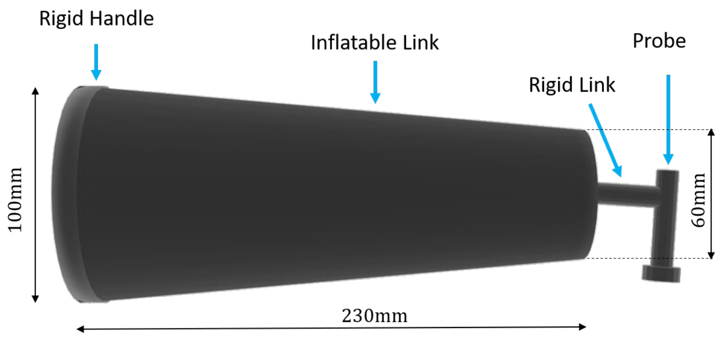


Figure 3.1: Components and Dimensions of the Simulation Link

The virtual link created is exhibited in figure 3.1, with the different components and measurements discriminated. The overall appearance, size and geometry were kept from the previous work [4], only simplified, as there is no need for an overlapping of the surfaces to close the sleeve. The changes will occur mainly at the internal pattern and the optical sensor.

It is worth noticing that the virtual link is controlled by a structure (armature in Blender designation) that encloses its physical properties. The virtual link was modelled as a cloth object and not a soft body since, in reality, it is an inflated fabric. Blender offers predefined cloth materials with every physics field filled-in to behave as expected for the material. Considering this the material ”rubber” was chosen and some values were then adjusted, such as weight and internal pressure.

The internal pressure feature was not used for the bending simulation as the link behaviour and the images produced were deemed realistic enough without it. As for the contact simulation, the pressure feature performed a prominent role since it involves collision with other objects and how the link responds to collisions will heavily depend on how pressurized it is.

It is at the armature that the displacements are to be directly applied, and it is according to the movement of the structure that the virtual link will then move, as exemplified in figure 3.2. The armature will correspond to the centerline of the soft link. As such, it is from there that the reference position values will be taken.

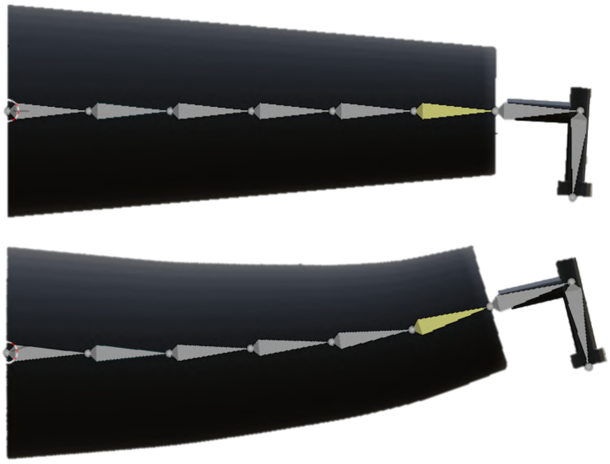


Figure 3.2: Armature of Link

The armature must be divided in enough segments to assure a smooth bending curve and the last segment and the end-effector must move together (to guarantee it behaves as a rigid link). For each subdivision of the armature, more variables are introduced to the system and the possibility of instability increases. So this number should be the smallest possible while being a multiple of three, so that there is a balance between beginning, middle and end of the curve. The centerline was divided into six parts, blocking the rotation of the last segment relatively to the two of the rigid body.

The bending motion results from an imposed displacement at the probe segment. The position of the other parts is defined by the software itself, through the use of inverse kinematics[5].

3.1.1 Contact Simulation

The goal of the contact simulation was to analyse the effects of surface interaction at the internal view. Some cylindrical objects were created to collide with the virtual link while it followed a predetermined trajectory. For this simulation, it was not as important to recognise the exact position of the soft link or how it bends, but how the interior markers deform with the contact with other objects and if it is possible to identify the location and direction of the contact.

The reference values of the contact location were retrieved computing the location of the object relative to the origin of the virtual link (located at the base).

Part of the simulation setup for contact is portrayed in figure 3.3.

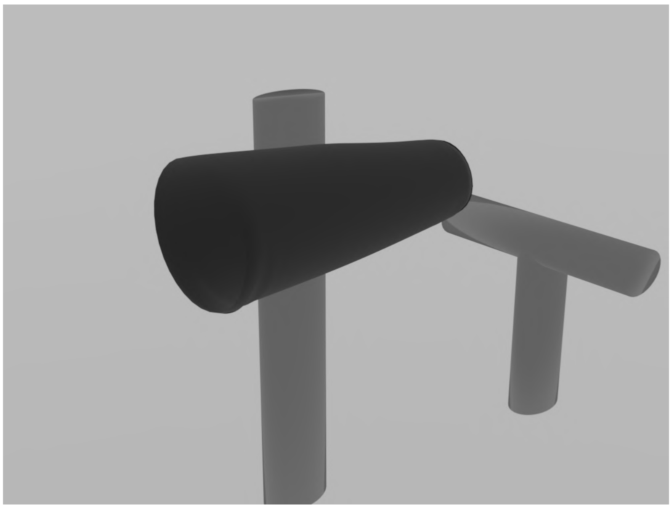


Figure 3.3: Contact Simulation Setup

3.1.2 Patterns

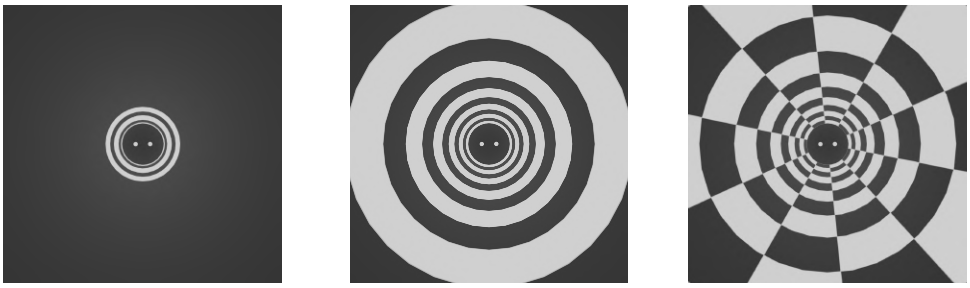
Firstly, the pattern to be applied to the inside of the link must be chosen. The chosen pattern needs to be able to provide enough information to reconstruct the bending curve. An easy way to do this is to use ring-like markers, using their centroids to estimate the curve. Three different patterns will be evaluated:

Original Work Pattern - 3.4(a) - rings covering approximately 40% of the link. Suited for the properties of the original work's camera. Used mostly for comparison purposes.

Striped Pattern - 3.4(b) - twelve rings covering the entire link, equally spaced and with the same thickness.

[5] https://docs.blender.org/manual/en/latest/animation/armatures/posing/bone_constraints/inverse_kinematics/introduction.html

Checkered Pattern - 3.4(c) - checkerboard-like pattern with the same amount of rows as the striped one.



(a) Original Markers **(b)** Ring Markers **(c)** Checkered Markers

Figure 3.4: Applied Patterns as captured by the new camera and lens

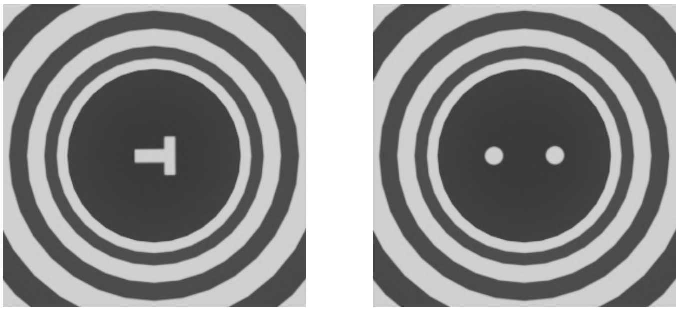
The decision of which pattern to use will be explained in chapter 4, alongside with the image segmentation process.

3.1.3 Distal Extremity Marker

The Distal Extremity Markers have two purposes: identify the position and the rotation of the end piece of the link. The markers have to be determined in a way that allows that specific information to be retrieved.

There were two distinct types of markers used:

- **T-shaped mark** - where the centroid and the moment of area indicate the position and the rotation of the centre of the piece, respectively, 3.5(a).
- **Two circular marks** - where the combined centroid of both circles shows the position and the centroids of each present the rotation, 3.5(b).



(a) Original Cover **(b)** Dotted Cover

Figure 3.5: Inside View Cover

It has to be noticed that, when the simulation goal is solely to find the position (in a pure bending motion), the type of cover marks applied will not alter the result, for the centroids of the T and the combined centroid of the circles are designed to be at the centre of the cover.

In the end, the markers chosen were the two circles, since they allow for an implementation using only the centroids of each marker as opposed to needing the entirety of the boundary.

3.2 Camera Properties

The software permits the use of several cameras for a single animation. For this case, two cameras were used: one centred at the base of the virtual link that accompanies its position and rotation and another to retrieve exterior images. This would be equivalent to the previous work configuration, in which a Trust SpotLight Pro Webcam[6] was used at the link base. The properties used for the simulation of this camera are presented in table 3.1

$Camera\ Property$	$Value$
Maximum Resolution	640×480
Focal Length / Field of View	120^o
F-stop	$f/11$

Table 3.1: Camera Properties

In figure 3.6, the referential of the camera used can be observed, in orange, as well as a global referential, in black, that will be used to present the results.

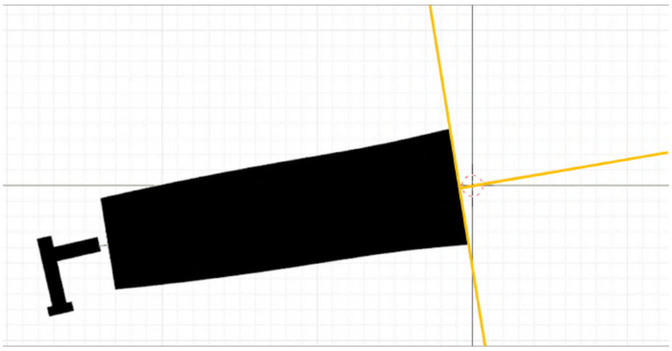
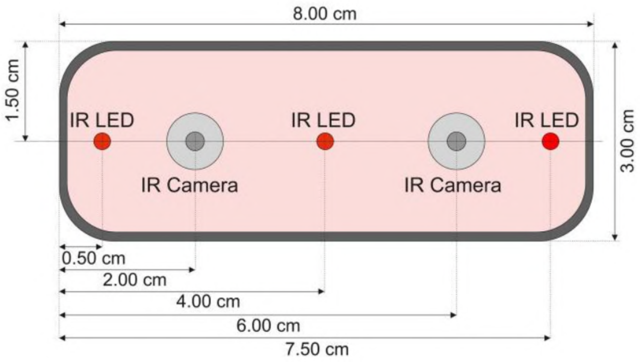


Figure 3.6: Referential of the camera

After the algorithm was developed and tested, the built-in camera at the base of the link was replaced by a stereo set, mimicking the leap motion sensor that will be used when recreating the simulation in an

[6] https://www.trust.com/en/product/16428-spotlight-pro-webcam-with-led-lights

experimental environment. At this stage, it is essential to change the settings of the interior cameras to match those of the Leap Motion Controller™ sensor.



source: Article [40]

Figure 3.7: Schematic View of Leap Motion Controller

The cameras are positioned $2cm$ before the base of the link, and $4cm$ away from each other, as seen in figure 3.7. The rest of the Camera Properties are presented in the following table, table 3.2.

$Camera\ Property$	$Value$
Maximum Resolution	640×240
Focal Length / Field of View	150^o
F-stop	$f/11$

Table 3.2: Stereo Set Camera Properties

Note that other properties of the sensor will not have a relevant role in the simulation. For example, the maximum frame rate of the sensor is an important aspect for the experimental procedure, as the trajectory has to be planned according to it.

3.3 Lighting Conditions

The animation software Blender offers the option to add an unlimited number of light sources. As such, it would have been possible to recreate lighting conditions that resembled a real experiment environment.

That said, it was decided not to follow this approach and even out the lighting in the simulation environment, for it would introduce shadows and other obstacles to image processing that might not even appear in the experimental case.

This was achieved by removing all light sources from the simulation environment and making the background white. That way, it is as if the entire background is the light source and the virtual link is receiving light from every direction homogeneously.

4

Image Processing

Contents

This chapter will focus on the processing of the images generated from the animation software until the inputs for the neural networks and the contact algorithms are attained. It is also here that the choice of internal pattern is made and the image segmentation methods are compared.

4.1 Image Segmentation

When the images from the animation software are rendered, they need to be converted into grayscale and then binary images. The grayscale conversion is made through a weighted sum of the three channel values (red, green and blue), following equation (2.2), in section 2.

After this, a threshold value is chosen, using the Otsu method, above which every pixel will take the value of one and below it, zero. The binary image is obtained.

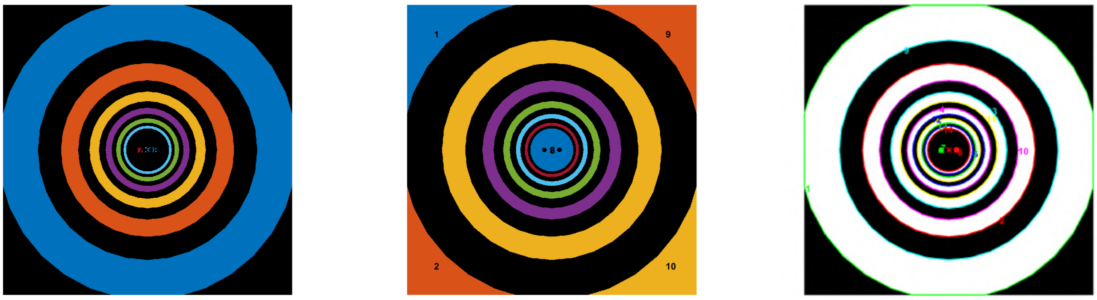
With the binary images, the process of image segmentation continues in order to use the relevant features of the image for future steps instead of the full image matrix.

Two different image segmentation methods were defined at this phase so that a comparison could be made relative to the impact on the performance estimating the bending curve: Blob and Boundary Detection. They are to be applied to all patterns so that the best combination can be selected.

Blob Detection: This method consists of dividing the image into groups of pixels - blobs - with similar characteristics, in this case, color - white or black. Then the centroids of the different blobs are computed and used to estimate the link deflection.

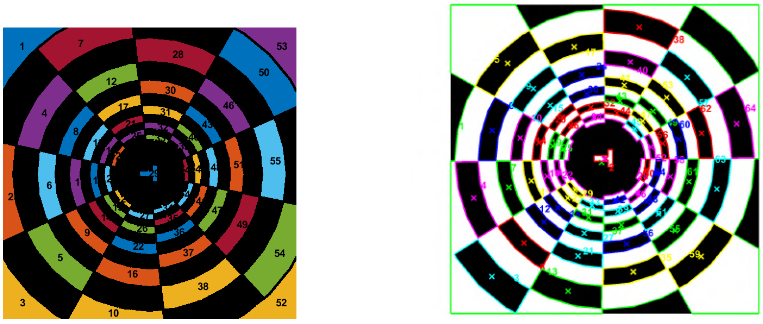
Boundary Detection: This method consists of identifying the edges of the image, which will correspond to the boundary of each blob, and then computing and using the centroid of each boundary to estimate the position of the link neutral line along the length.

Images portraying both of the methods can be found in figures 4.1 and 4.2. Note that, in method 1, both the binary image and the complementary of that, 4.1(a) and 4.1(b), must be segmented because only the white connected components, with value 1, are identified as objects since the black connected components, with value 0, are perceived as background.



(a) Blob Detection 1 **(b)** Blob Detection 2 **(c)** Boundary Detection

Figure 4.1: Segmentation Methods - Striped Pattern



(a) Blob Detection **(b)** Boundary Detection

Figure 4.2: Segmentation Methods - Checkered Pattern

At this stage, it is possible to eliminate the checkered pattern. Although it might have an advantage in identifying the location of a contact, it is more sensitive to errors due to having to group the centroids in the correct row and combine them to find the centerline of the link. Besides, for images referring to a large deflection, it is much harder to separate all the blobs or boundaries and evaluate them as different objects, with examples shown in appendix A.1.

As for the segmentation method, it is not possible to deem one alternative better than the other in terms of execution, only the computation time required. In table 4.1, the average computation time required for each method is presented, for both the case that the images are displayed in the screen and the case that they are not. The boundary segmentation is significantly faster, specifically when not showing the image assessed, when the computation time is more than one order of magnitude smaller.

$Method$	$ImageDisplay$	$Time(s)$
Blob	Yes	1.2299
Blob	No	0.8290
Boundary	Yes	0.5375
Boundary	No	0.04632

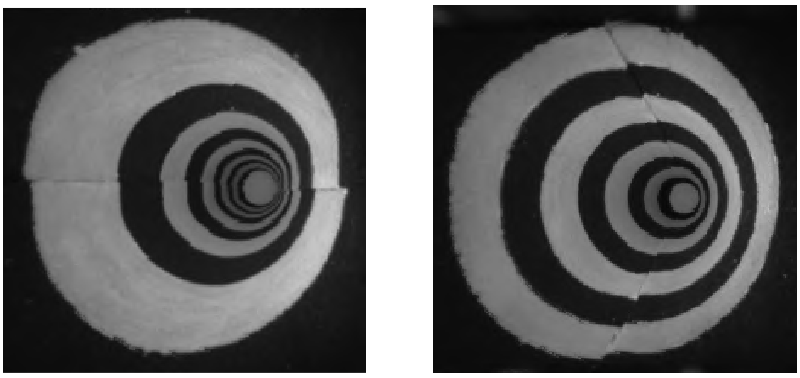
Table 4.1: Computation Time Required - Striped Pattern

Both methods provide the area within the boundary or the area of the blob, and the location of their centroids. This is the information that will be used to compute the deflection of the inflatable link. For the contact simulation the moments of inertia in the principal axis of each blob or boundary are computed, as well as their direction.

4.2 Experimental Work

When transitioning to the experimental work there were slight changes in the pattern that had to be made, so that it was possible to identify all the rings independently. That way a new pattern was created, the **Exponential Distributed Striped Pattern**, with stripes of different thickness, growing thicker with the distance from the base, so they are perceived to have the same width by the camera.

Figure 4.3 shows the comparison of the two patterns. Note that the discontinuation in the markers corresponds to a seam.

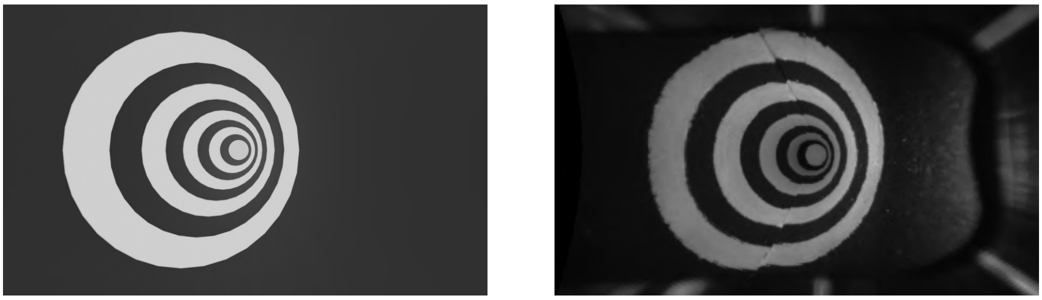


(a) Same Width Stripes

(b) Exponential Distribution Width Stripes

Figure 4.3: Comparison of Patterns

The process of treating the images retrieved from the experimental setup is slightly different mainly due to the lighting conditions. The original image from the sensor is displayed in figure 4.4(b) with the comparison of the image from the simulation, figure 4.4(a). The further away it gets from the camera, the darker it translates to the image.



(a) Simulation

(b) Experimental

Figure 4.4: Lighting - Simulation vs Experimental

There is a need to even out the lighting of the image before making it binary, at the risk of losing the markers distant from the camera. Several methods were tested to assess which allowed for the best results. The process can be observed in figure 4.5.

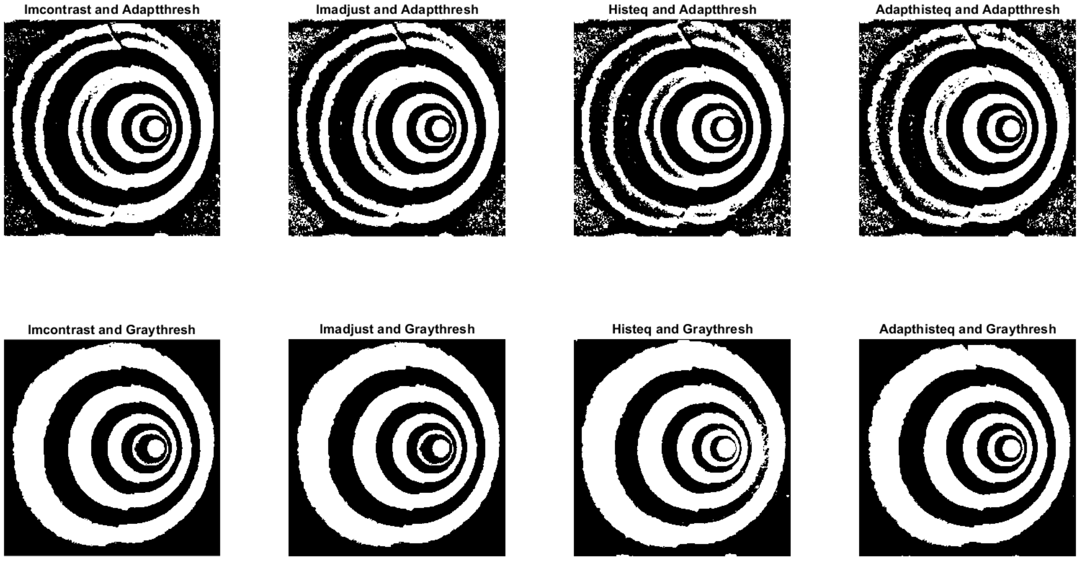

Imcontrast and Adaptthresh
Imadjust and Adaptthresh
Histeq and Adaptthresh
Adapthisteq and Adaptthresh
Imcontrast and Graythresh
Imadjust and Graythresh
Histeq and Graythresh
Adapthisteq and Graythresh

Figure 4.5: Binarization Methods

The top row shows the results of applying an adaptive threshold, when it is not one single threshold value but more, chosen accordingly to the local mean intensity in the neighbourhood of each pixel of the grayscale image. The last row presents the results of applying a single threshold value, found using Otsu's method.

The first column represents the end product of applying user defined limit values for the histogram, which represent the new values for black and white. The other columns represent methods previously discussed in section 2.1.2.A; the second column corresponding to a saturation of the 1% top and bottom values of the histogram, the third column to an histogram equalization and the fourth to an adaptive histogram equalization (CLAHE). All of the methods in figure 4.5 are identified by the corresponding MATLAB® functions for contrast correction and threshold used.

The method chosen was to use the CLAHE with a single threshold (last picture), since it is the one that mostly clearly defines the boundaries of the rings. This method is shown in more detail below, in figure 4.6.

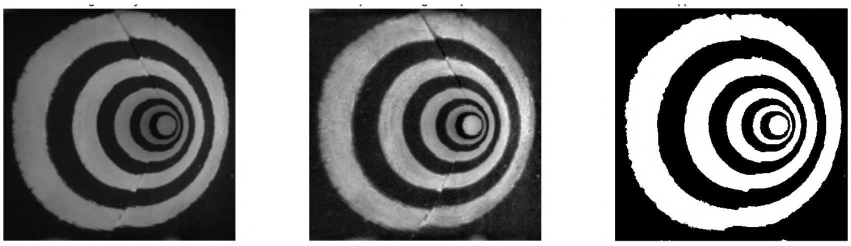


Figure 4.6: Binarization Process Chosen

After having found a suitable binarization method for the lighting conditions, the rest of the segmentation process follows the same steps reviewed previously for the simulation work. In the end, this pattern is capable of providing the same information as the one used for the virtual simulations, only for a smaller number of markers.

5

Results and Discussion

Contents

This chapter will focus on the executed simulation process for the deflection and the contact problems, explaining the several integral steps and presenting the results obtained as well as a discussion of their meaning for the project in its entirety.

5.1 Bending Problem

5.1.1 Bending Algorithm

The algorithm created to estimate the deflection uses the initial position of the centroids as the resting pose of the soft link (deflection=0m) to compare with the sequence of images, combining the displacement of the corresponding centroids with the pixel to meter conversion. This conversion is found using an established object that appears in the image, for example the tip, which has a known diameter.

The algorithm will return the values described in the link's own reference frame so, to present the results more intuitively, the bending curve needs to be corrected with the angle of the camera, allowing for the analysis to occur in a global reference frame.

This adjustment is a simple rotation of reference frame, with the rotation matrix about the angle of the camera, as depicted in (5.1).

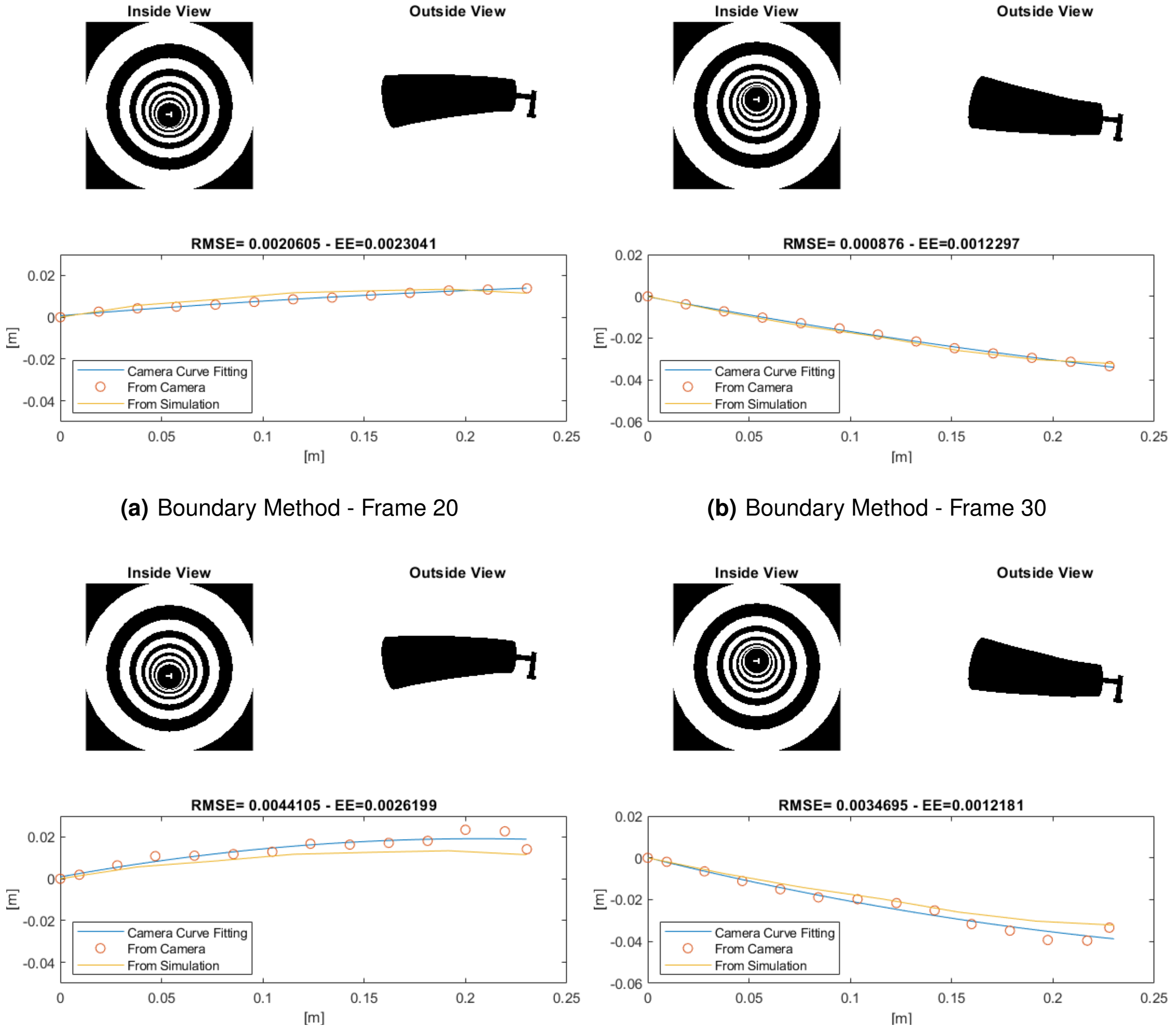
$$[\boldsymbol{R_\alpha}] = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix} \tag{5.1}$$

The algorithm developed for bending estimation is presented in algorithm 5.1.

Algorithm 5.1: Bending Estimation Algorithm

```
Input: Position of Centroids ($C_i$), in pixel, and angle of camera ($\alpha_i$)
Output: Deflection
1. Compute displacement of centroids, considering the initial position as resting pose:
$\Delta y = (C_i - C_1)$
2. Apply pixel to meter conversion previously found using a known reference:
$\Delta Y = 1.7920 \times 10^{-4} \times \Delta y$
3. Change reference frame:
$Deflection = R_\alpha \times \Delta Y_i$
```

This process was repeated for both segmentation methods and the results can be examined in figure 5.1. The deflection curve is presented as second order polynomial curve fitting, with the estimated points identified as markers, over the curve from the simulation software.



(a) Boundary Method - Frame 20

(b) Boundary Method - Frame 30

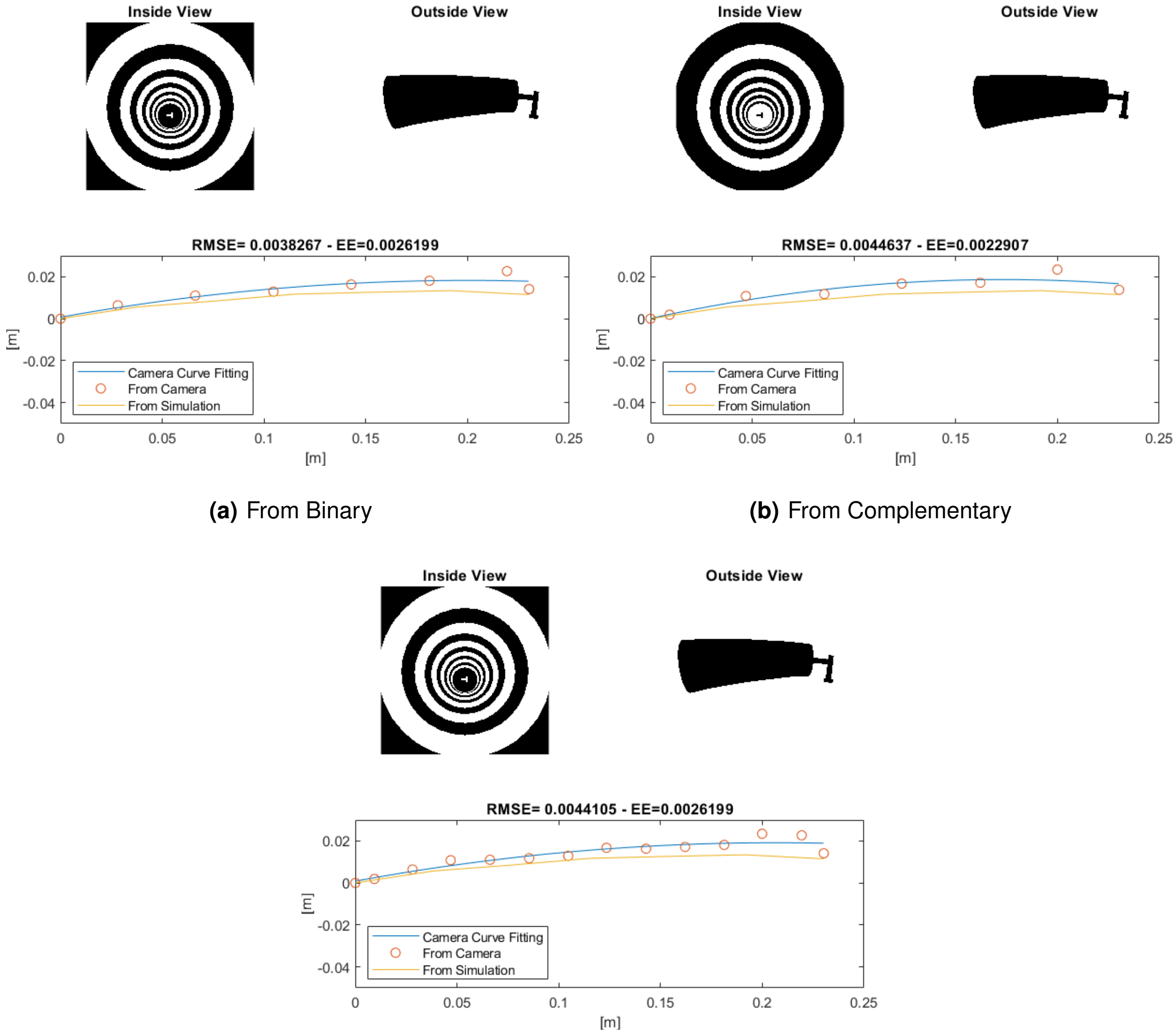
(c) Blob Method - Frame 20

(d) Blob Method - Frame 30

Figure 5.1: Boundary vs Blob Performances

The performance of the algorithm will be evaluated by 3 factors: Root Mean Squared Error (RMSE) (measured using the second order polynomial curve adaptation of the points), Extremity Error (EE) (measured using only the last value) and computation time. One aspect to take notice of is that this algorithm is very susceptible to errors for deformations where occlusion of markers exist. It loses information to work with and the information it does receive is distorted, which represent situations that the algorithm does not have the tools to compensate.

For the first segmentation method, Blob Detection, further analysis was made regarding which of the markers would be the most valuable, the ones from the binary image or the complementary image. This assessment is portrayed in figure 5.2. Although the performance differs with the markers used, none of the Blob Detection options offer a better result than the Boundary Detection.



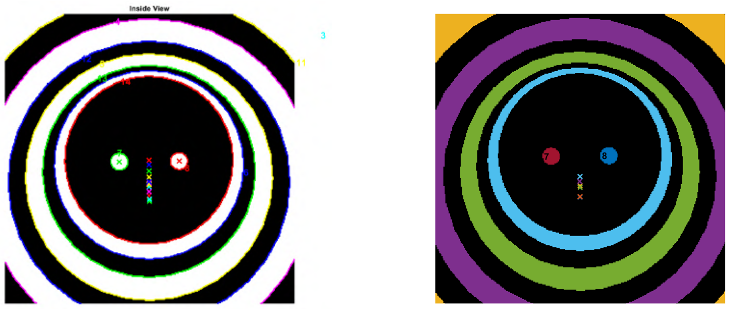
(a) From Binary

(b) From Complementary

(c) From Binary and Complementary

Figure 5.2: Blob Method choosing different marks

When examining the segmentation closely, it can be noticed that the centroids of the blobs are very affected by the distortion of the rings; as one side grows thicker and the opposite grows thinner, the centroid will shift towards the thicker side, 5.3(b). The distortion of the rings does not affect as much the boundary method, 5.3(a), since all the edges remain to be circles. This is likely the reason why the boundary segmentation method has a better performance. The effect of the distortion of the rings can be observed in figure 5.3.

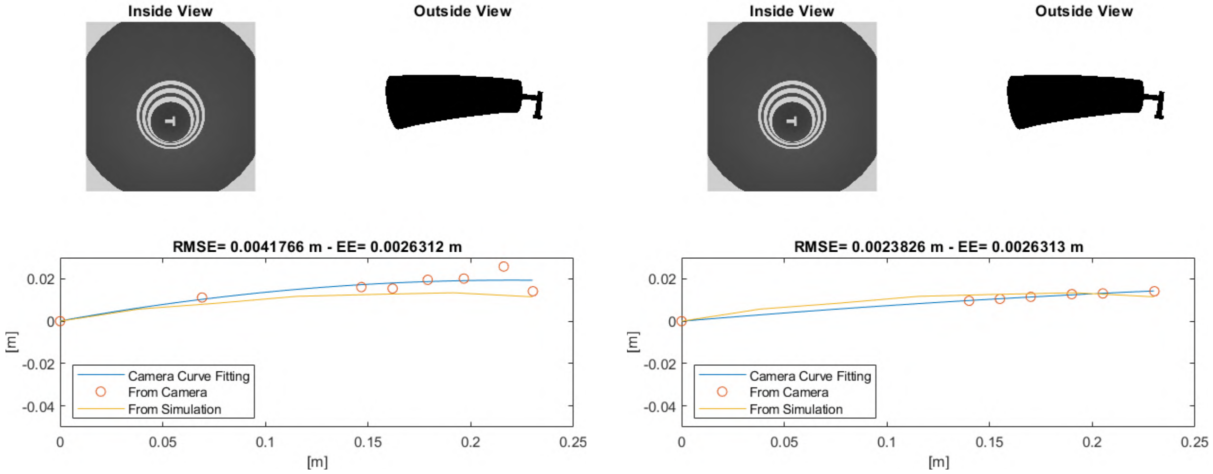


(a) Boundary Method **(b)** Blob Method

Figure 5.3: Close up on Centroids

Comparison with Original Pattern

The original pattern was also tested for this algorithm so a direct comparison between patterns was possible. The results for both segmentation methods are in figure 5.4.



(a) Blob Method **(b)** Boundary Method

Figure 5.4: Bending - Original Pattern

As expected, the error at the tip was not improved with the pattern as it still receives the same information. What is significantly better is the error along the bending curve, since the number of markers as increased. It is to be noticed that if the bending radius is significantly large (when the bending curve is not as accentuated) the algorithm produces accurate results, as depicted in figure 5.5, but the new pattern works as well for curved shapes with a smaller bending radius.

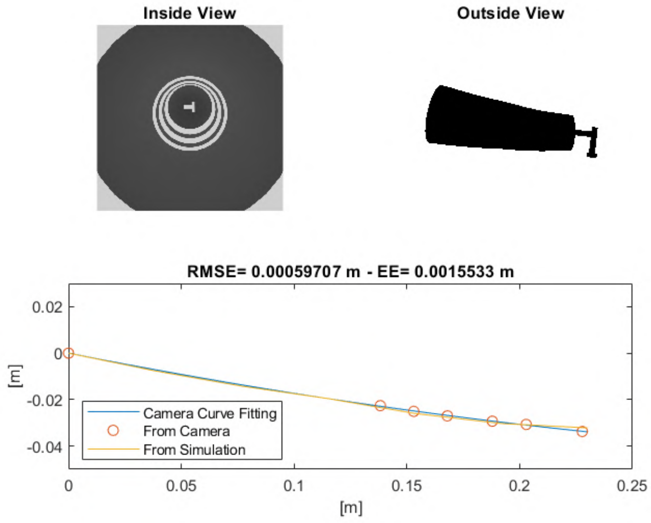


Figure 5.5: Original Pattern - Accurate Estimation

5.1.2 Neural Networks

The purpose of using NN is to establish if it can compensate the occlusion of markers or small deviations from the segmentation and, overall, if the curve error can be reduced.

The NN will take the values directly from the segmentation stage and return the bending curve. As such, this alternative will not improve the most computationally demanding part of the procedure, the segmentation itself. The inputs chosen for the NN were the marker's Area, Centroid Position (vertical only), Position along length and Angle of camera

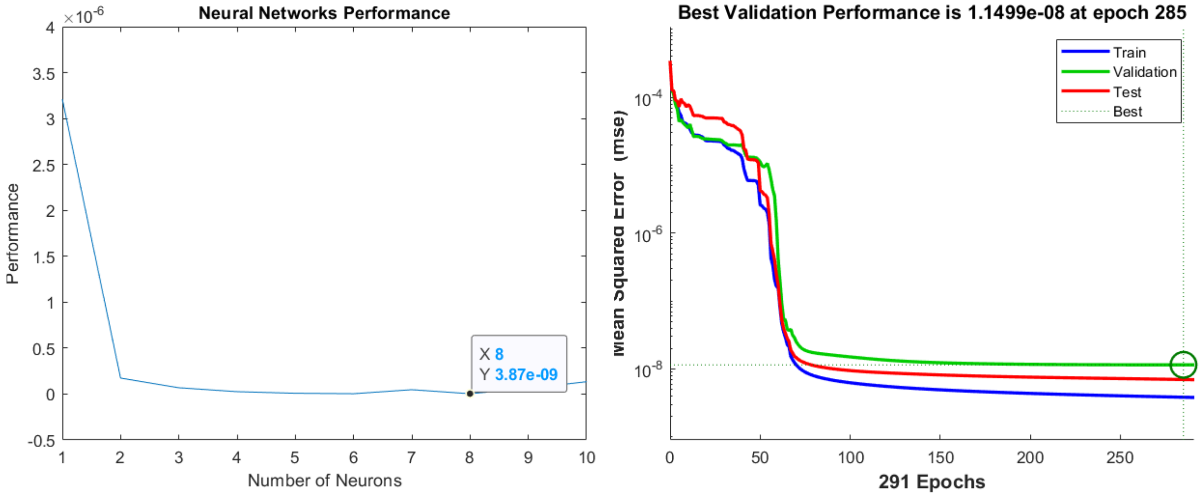
It is worth noticing that the position of the centroid is only evaluated on the $y - axis$ because the variation in x is negligible and the alpha angle is meant to adjust the deflection to the world's reference frame

The creation of the NN happens inside a training loop that increases the number of neurons of the hidden layer and compares the performance of each created network, always keeping the best one. The data that goes in the loop does not correspond to the entirety of the collected data, 10% of it is removed from the set as it will be used to test the created network. Inside the training loop, the data is divided into train and validation sets, in a ratio of 85 to 15, respectively. This means the network trains with 113 input-output pairs out of 147.

Since the training values are chosen randomly within the set and they greatly influence the performance, the NN trains several times with the same architecture (number of layers and neurons), only changing the data distribution for training and validation. This will guarantee the network several chances to find a better training set.

The network was trained varying the size of the hidden layer, the best performance being 3.87×10^{-9}

for 8 neurons. The mean performance of the NN for each hidden layer size is displayed in figure 5.6(a). The train of the best NN was finished in epoch 285, corresponding to the best validation loss. From figure 5.6(b) it is verified that all three sets (training, validation and test) present a similar convergence curve, which indicates a good network, without overfitting.



(a) NN Performance Mean with Number of Neurons 1 to 10

(b) Training Convergence for Best NN

Figure 5.6: NN Performance

The NN was configured through the MATLAB® function *fitnet*, a tool that creates a completely connected feedforward network typically used in curve fitting problems and regressions. The training function chosen was *trainlm*, which corresponds to Levenberg-Marquardt backpropagation. The NN is defined as having one hidden layer and a linear output layer, as shown in figure 5.7.

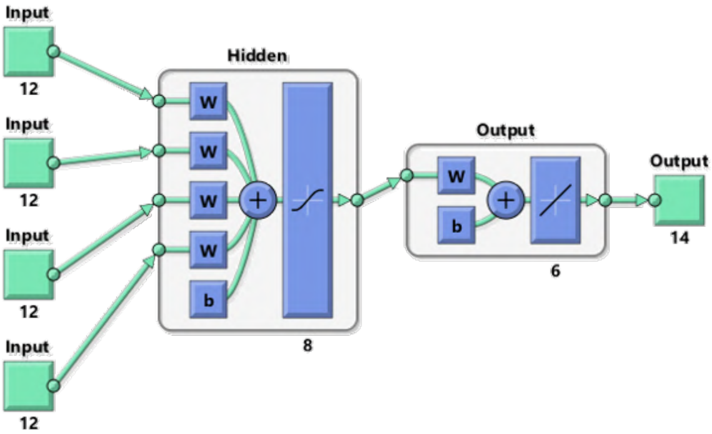
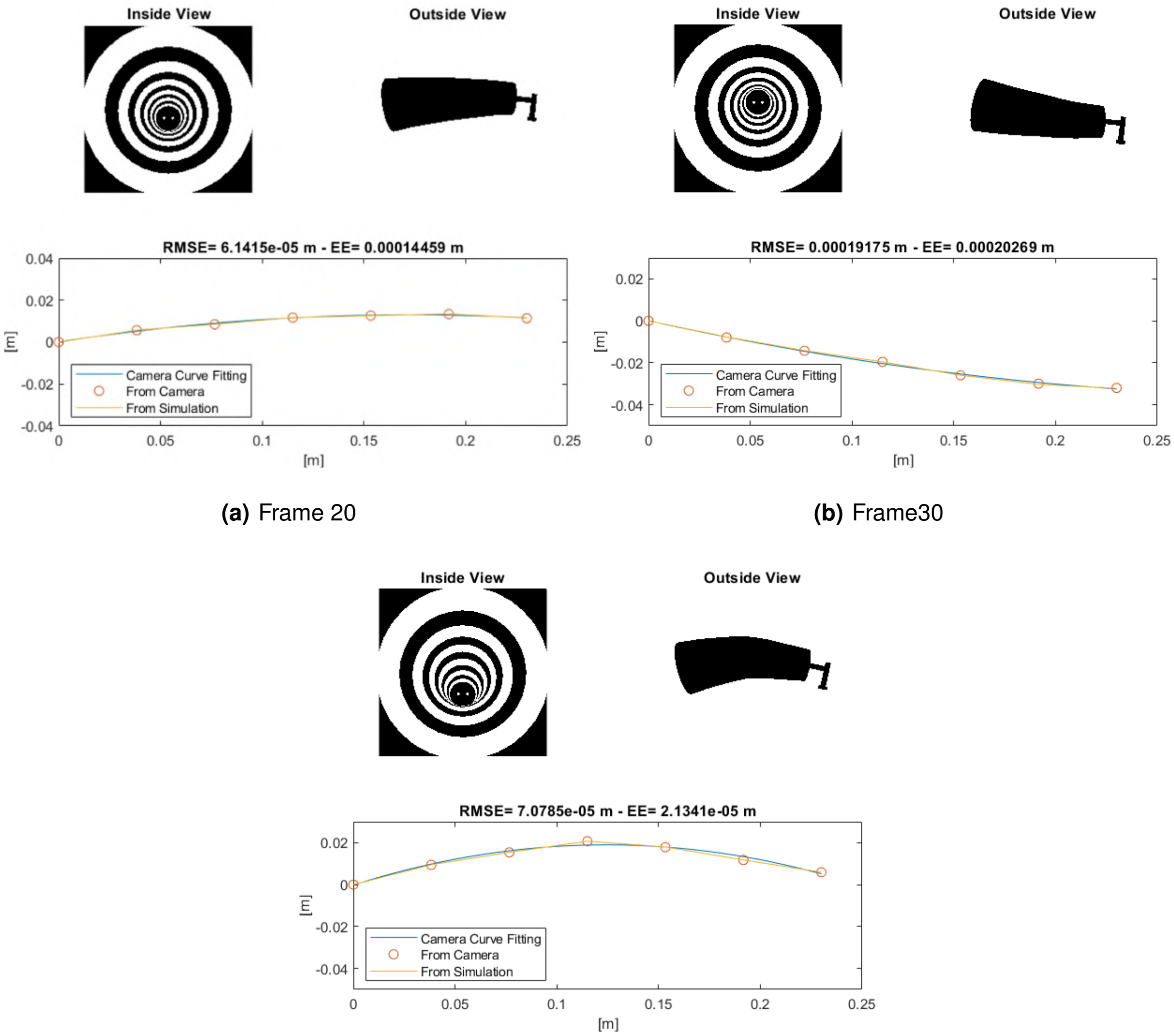


Figure 5.7: NN architecture for Bending

The performance for this NN is presented in figure 5.8, with considerable improvements from the previous method, particularly in estimating the bending curve (RMSE). In figure 5.8(c), it is portrayed an accentuated curve of the soft link, where three markers are not fully captured. The deflection results shows that the NN approach can provide precise estimations even if there is occlusion of markers.



(a) Frame 20

(b) Frame30

(c) Frame50

Figure 5.8: NN Performance - Bending in One Direction

Once it is established that the NN can calculate the deflection for bending in one direction, the ensuing step is to verify the estimation when the link bends in multiple directions.

5.1.3 Bending in Multiple Directions

To evaluate if the NN can predict the bending in various directions, different rotations are applied to the collection of images. Since the markers on the distal extremity are not processed with the geometry information, only the resulting centroid, it is not necessary to rotate the extremity back to the original position. This also enables the verification of the bending direction through the two markers, using the position of each one in x and y.

The goal of this stage will be to verify if the NN can estimate the deflection in the main bending direction, when it has to combine the displacements in the orthonormal axes. The estimated deflection will only be defined in the main direction (the target values are defined in a rotated frame as well).

In an effort to keep the NN as close as possible to the first, the inputs chosen for the NN were the marker's Area, Centroid Position (vertical and horizontal), Position along length and Angle of camera. The training process and determination of the best network created follows the same method described for the previous NN. In figure 5.9 is the architecture of the best performing NN created.

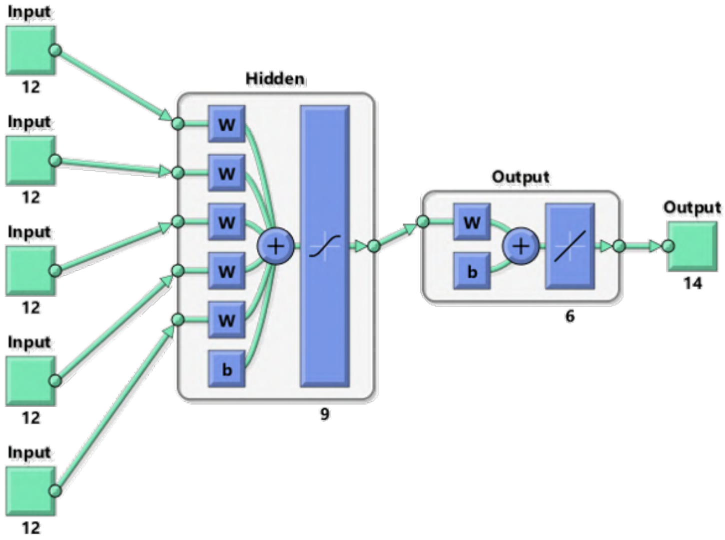
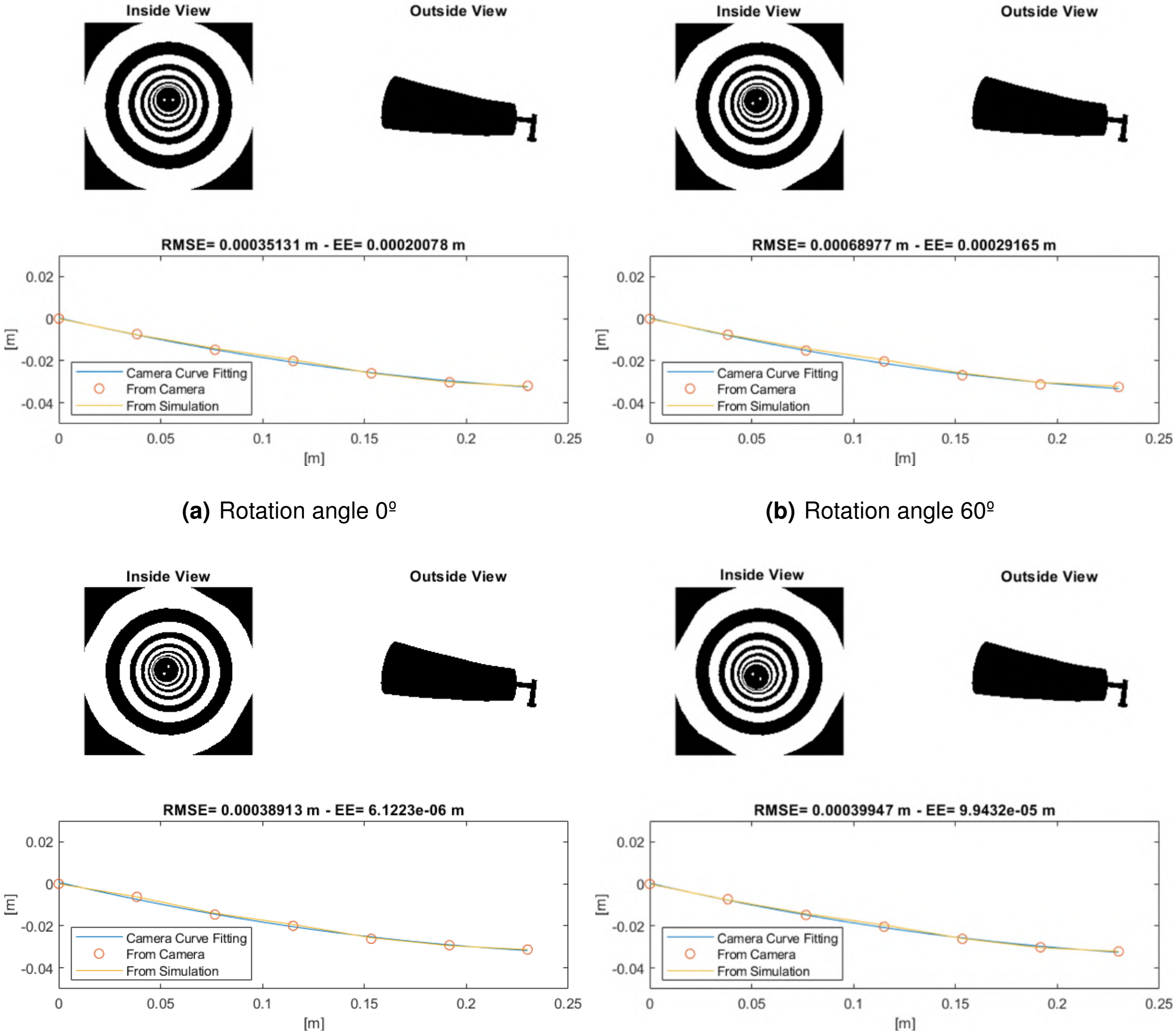


Figure 5.9: NN architecture for Bending in multiple directions

The evaluation of the results occurs with the comparison of the curves for the same image with different rotations applied. An example of this process is presented below, in figure 5.10.



(a) Rotation angle 0º

(b) Rotation angle 60º

(c) Rotation angle 120º

(d) Rotation angle 150º

Figure 5.10: NN Performance - Bending in Different Directions

5.1.4 Bending Discussion

When comparing the different methods, it's possible to see that they estimate fairly well the bending curve of the link, the NN providing the best performance (with both EE and RMSE around 10^{-4}m). As for the bending algorithm, it is noticeable that the segmentation option chosen will influence the error of the entire curve (RMSE) but not as significantly the EE. The Boundary segmentation has a clear advantage when retracing the deflection but, since the marker on the extremity does not suffer noticeable distortions, both of the methods have a similar performance on EE.

From the bending in multiple directions section, it is possible to extrapolate that the algorithm is able to recognise and assess bending in various planes (not just vertically), being able to determine accurately the deflection curve in the main direction.

In table 5.1, the performance for each of the methods can be analysed. The computation times are all very similar, all at the same order of magnitude, 10^{-5}s, that way, it is fair to choose the best method based on the accuracy alone, using the average RMSE and EE values. Additional bending results can be observed in appendix A.2.

$Method$	$Computation\,Time\,(s)$	$RMSE\,(m)$	$EE\,(m)$
Bending Algorithm + Bound	8.333×10^{-5}	0.0013	7.5314×10^{-4}
Bending Algorithm + BlobComp	8.333×10^{-5}	0.0017	7.4768×10^{-4}
Bending Algorithm + BlobBin	8.333×10^{-5}	0.0015	8.8917×10^{-4}
Bending Algorithm + BlobAll	8.333×10^{-5}	0.0019	8.7730×10^{-4}
NN (Vertical)	5.238×10^{-5}	6.6082×10^{-5}	1.1047×10^{-4}
NN (Multiple Directions)	1.020×10^{-5}	2.8817×10^{-4}	4.0691×10^{-4}

Table 5.1: Performance - Bending Curve

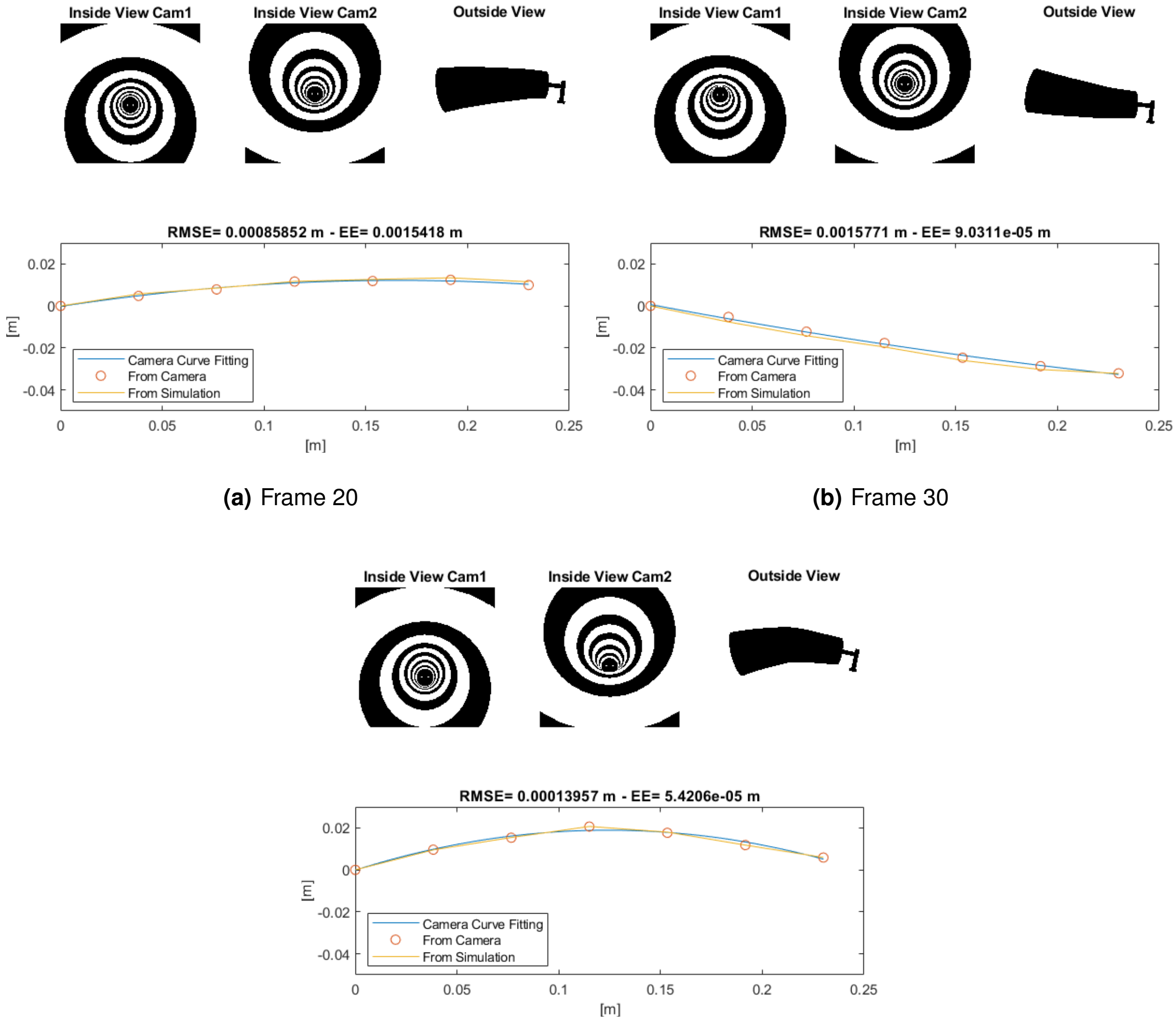
After inspecting table 5.1, it is evident that the bending algorithm does not achieve an adequate performance when compared to the NN approach. Moving forward, all simulations will be assessed using NN. Having determined that the algorithm works with a single camera centered at the base of the link, the following step is to test with two cameras, using the structure of the Leap Motion Controller™ sensor.

5.1.5 Stereo Vision

The tests made for Stereo Vision were fairly simple since the relevant part was to evaluate if the NN could still predict accurately the deflection curve of the link using two non centred images. It was decided against merging the two pictures into one that most closely resembled all of the images that had been worked on mainly because it is a process that takes much computation time and therefore is not practical for real-time simulations. There were other reasons not to merge the information, namely, the lack of unique feature points and to receive the maximum amount of information.

A new NN was created and trained using six inputs: Area and position of the centroid (vertical and horizontal) for both images. Since the NN always takes all the values from both pictures, even when there is occlusion of markers from one of the cameras, the NN is capable of getting information from the other.

The performance of the NN application in the stereo images can be seen in figure 5.11.



(a) Frame 20

(b) Frame 30

(c) Frame 50

Figure 5.11: Stereo Set - NN Performance

Below, in table 5.2, the average performance of the NN is exhibited.

$Method$	$Computation\,Time\ (s)$	$RMSE\ (m)$	$EE\ (m)$
NN Stereo Vision	6.054×10^{-5}	3.4895×10^{-4}	4.4119×10^{-4}

Table 5.2: Stereo Set Performance - Bending Curve

At this point, the method developed is adequately examined and presented a successful performance. The next step is to validate it in an experimental setting.

5.2 Contact Problem

5.2.1 Contact Algorithm

The method chosen to identify whether there was contact with the surface of the soft link was to evaluate, for each boundary, the relationship between the moments of inertia in the principal axis, with I_1 and I_2 as the minor and major principal moments of inertia, respectively.

If there is no contact, all of the boundaries will have a circular shape, approximately, which means that the ratio of moments of inertia will be close to 1. If the contact exists, then the circular shape is deformed, which will lead to an alteration of the ratio of inertia moments.

When the algorithm finds a peak value in the ratios, there is a situation of contact and the algorithm proceeds to identify the location ($CPoint$) and angle of direction ($CDirection$) of the object relative to the link.

The algorithm developed for contact detection is presented in algorithm 5.2.

Algorithm 5.2: Contact Estimation Algorithm

```
Input: Position of Centroids ($C_x$, $C_y$), Principal Inertia Moments ($I_1$, $I_2$) and respective direction
    ($\theta_1$,$\theta_2$)
Output: Contact point ($CPoint$) and Contact direction ($CDirection$)
1. Compute ratio of moments of inertia:
$Ratio = I_2/I_1$
2. Find peak value and respective index of boundary.
3. Compute Contact point:
$CPoint = max_{index} \times width_{marker}$
4. Get the directions of the minor axis ($\theta_2$) and the centroid trajectory($\theta_c$) for the boundary:
$CDirection = \theta_2$
$\theta_c = arctan(C_{y_i} - C_{y_1}, C_{x_i} - C_{x_1})$
5. Correct contact direction with a $\pi$ deviation - only if centroid trajectory direction is in the same
  quadrant:
$CDirection = CDirection + \pi$
```

The direction of the contact is retrieved from the minor axis angle (θ_2). There has to be a decision mechanism after this, to chose if the angle is correct as is or if there is a deviation of π. This is made using the neutral position of the centroids of the link (C_1). The angle of contact is opposite to the direction the centroid moves (θ_c). This effect is shown in figure 5.12.

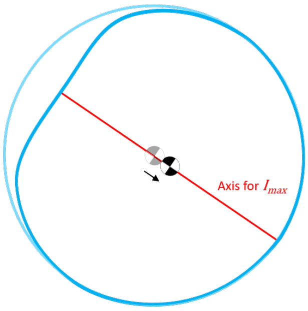
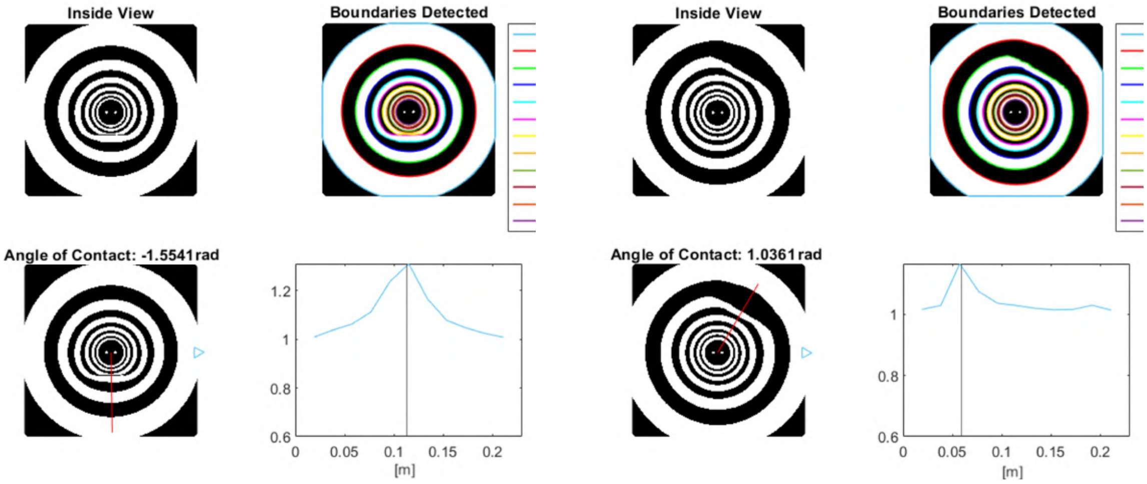


Figure 5.12: Centroid movement along principal axis

Figure 5.13 portrays the different elements involved in the contact detection, for a more intuitive identification of the estimated contact point and direction.



(a) Frame 85

(b) Frame 163

Figure 5.13: Contact Evaluation

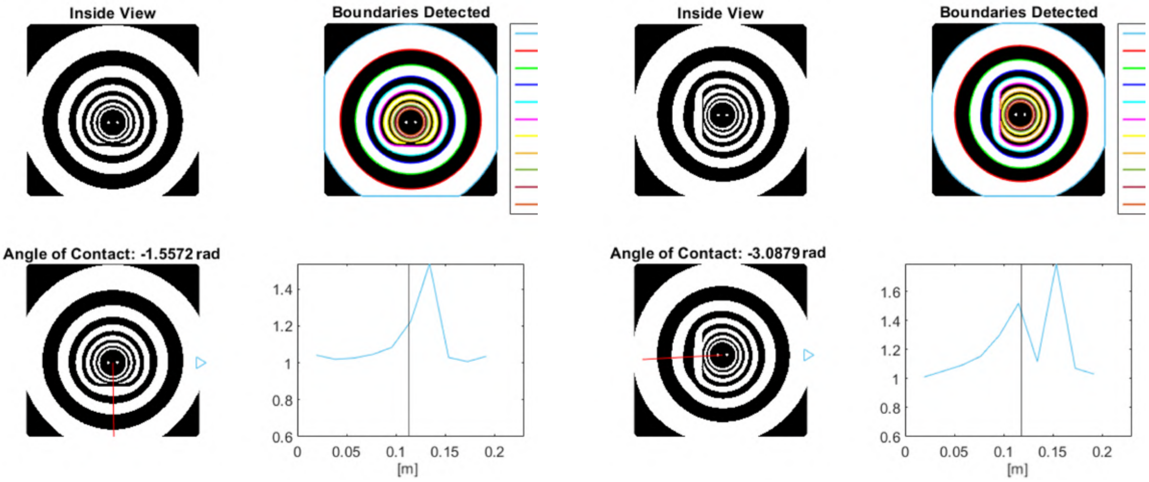
Note that the contact position received from the simulation is plotted as a vertical line for a straightforward comparison with the peak value.

In the table below, table 5.3, the performance of the developed algorithm for contact detection is exhibited.

$Method$	$Computation\, Time\ (s)$	$ImageDisplayed$	$RMSE\ (m)$
Contact Detection	0.22144	Yes	0.0169
Contact Detection	2.756×10^{-5}	No	0.0169

Table 5.3: Contact Detection Performance

This algorithm achieves a location error of $1.69\ cm$, which means that it can place the contact in the right marker, considering their width is about $1.9cm$, but can be improved if a more generalized approach is taken. For the case that an occlusion of the markers exists, the comparison is not as straightforward for the occlusion might alter the shape of the markers beyond it. In figure 5.14, two of these cases may be observed.



(a) Frame 55

(b) Frame 145

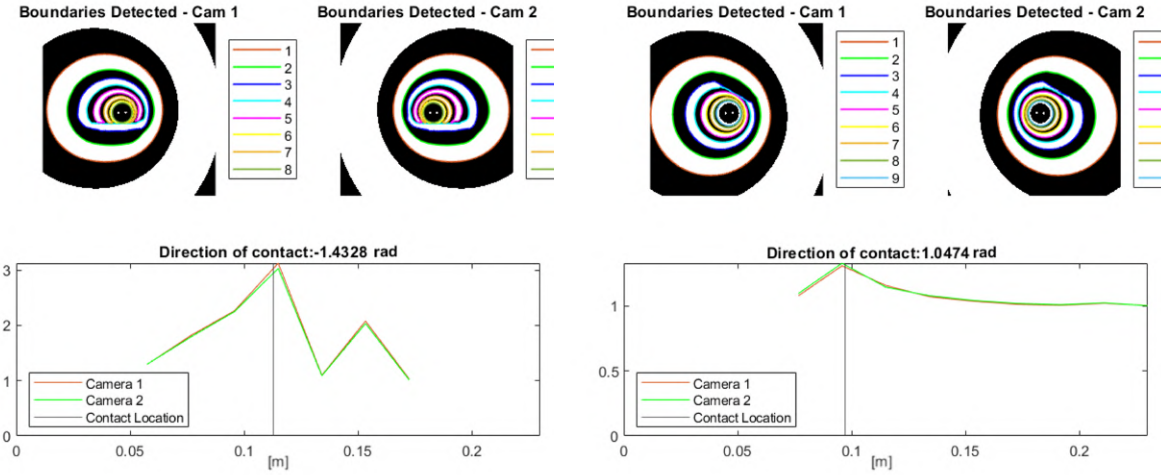
Figure 5.14: Contact Evaluation - Generalization

With these cases in mind, a more appropriate definition is to identify the first peak value or to specify a limit value above which the ratio should be disregarded, as it corresponds to an incomplete marker. This limit should vary for each boundary.

Like for the bending problem, the final step is to apply the same type of method to the stereo vision case.

5.2.2 Stereo Vision

The stereo images do not capture the first marker completely, so it is disregarded, as it is not applicable for the ratio of moments of inertia method. That also means that any contact happening in the first marker or before it will not be recognised.



(a) Frame 85

(b) Frame 163

Figure 5.15: Contact Evaluation - Stereo Vision

There are aspects to improve on the definition of the method since it suffers, as expected, from the same shortcomings of the single-camera one. Additional contact estimation results can be observed in appendix A.3.

This method would be a good candidate for a NN approach, as it would establish rules for the occlusion of markers.

The developed process should not be regarded as a fully finalized method but as the first step of problem-solving and evidence that the second moment of area of the markers provides relevant information for contact characterization.

6

Conclusion

Contents

This chapter presents the conclusions of the developed project, covering its successes and downfalls as well as future work recommendations.

6.1 Conclusions

In this dissertation it was proposed a vision based system as a proprioceptive sensor to estimate the position and contact situation of a inflatable link, through deformations of its internal pattern.

Regarding the created virtual inflatable link and the simulation environment, the animation software Blender was a suitable choice, allowing to retrieve realistic enough images of the the movement of the link as well as the centerline and contact point positions. This was confirmed by comparing the simulation generated images with images obtained from the experimental prototype in preliminary trials.

During the work, it was noted that, instead of performing blob analysis of the ring markers inside the link, there are situations where an assessment of the blob boundaries provides sufficient information. For those cases, boundary analysis is very advantageous, mainly because of its speed, allowing for better results when tracking in real time.

For the bending simulation, it is possible to infer that the markers in the chosen pattern are sufficient to get a smooth and accurate bending curve. It is also perceptible that the process of training a NN is worthwhile since when the network is trained, the deflection it measures, along the curve and at the tip, is much more reliable than the empirical algorithm results - the error is around 20 times smaller along the curve and 7 times at the end point. The NN approach is also promising when evaluating bending in multiple directions, although these results should be confirmed through further simulation work.

It was observed that the stereo camera set does not hinder the bending simulation, with a new NN being able to estimate the deflection curve taking information from both images retrieved. The stereo set also offers an advantage when dealing with occlusion of markers, particularly when the deformation occurs in the plane of the two cameras, perpendicular to the image plane, with one of the images probably being occlusion-free, or, at least, less affected by it.

As for the contact simulation, the ratio of inertia moments is a relevant information, but might not be enough to confidently establish the point of contact, with the average error of the simulation being about $1.7cm$. In the future, results may be enhanced by defining more rules, like limit values to consider (to avoid choosing a marker with occlusion) or using another approach to compute the contact location, for example NN. Another aspect to face is that it might not be possible to get a performance as good as for the bending problem due to the markers width. It can be hard to pick apart contact points on the same marker, even if one is at the beginning of the stripe and the other is at the end, since it is the same marker that is affected. As for the direction of the contact, the method chosen returns accurate results, as long as the resting pose is the same. If the inflatable link is less pressured than what it was trained for, the centroid positions may not match, making the decision mechanism for the direction unreliable.

As the contact problem is not as refined as the bending with a single camera, the simulations with the stereo set were not very deep as well, but some limitations were observed, specially concerning the first marker, that was not fully enclosed in the images and, as such, the ratio of inertia moments was not approximately 1. This problem has already been solved in the new prototype that will be used in the experimental environment, with the new internal pattern using the exponential distribution of the markers width, that guarantees the first marker is fully captured by the camera.

As a final conclusion, it was demonstrated that the designed pattern can in fact provide relevant information for both bending and contact situations, and that a approach merging computer vision and neural networks is a suitable solution for tracking the internal state of the inflatable link.

6.2 Future Work

There are some limitations on the developed system and interesting aspects that could be solved or improved moving forward with this work. The following recommendations are indicated:

- Test algorithm for bending in multiple directions with stereo vision.
- Validate results in experimental context.
- Improve contact algorithm decision mechanism, based on the suggestions given or other that are considered more suited.
- Integrate a NN for contact point estimation in the contact detection algorithm.
- Expand simulation range of experiments, for example with torsion, to evaluate the performance of the markers at the link distal extremity.
- Study feature selection for the inputs of the NN to evaluate the most relevant features (between second moment of area and area).

Bibliography

[1] G. Bao, H. Fang, L. Chen, Y. Wan, F. Xu, Q. Yang, and L. Zhang, “Soft robotics: Academic insights and perspectives through bibliometric analysis,” 2018.

[2] A. Ferreira, “Inflatable structures for robotic applications,” Master’s Thesis, Instituto Superior Técnico, Lisbon, 11 2016. [Online]. Available: https://fenix.tecnico.ulisboa.pt/cursos/memec/dissertacao/1691203502342586

[3] J. Oliveira, “Controlo de um manipulador robótico insuflável,” Master’s thesis, Instituto Superior Técnico, Lisbon, 6 2017. [Online]. Available: https://fenix.tecnico.ulisboa.pt/cursos/memec/dissertacao/1691203502342476

[4] A. Franco, “Estimation of contact force on an inflatable structure using vision sensors,” Master’s thesis, Instituto Superior Técnico, Lisbon, 11 2019. [Online]. Available: https://fenix.tecnico.ulisboa.pt/cursos/memec/dissertacao/846778572212368

[5] S. Kim, C. Laschi, and B. Trimmer, “Soft robotics: a bioinspired evolution in robotics,” *Trends in Biotechnology*, vol. 31, no. 5, pp. 287–294, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167779913000632

[6] D. Trivedi, C. D. Rahn, W. M. Kier, and I. D. Walker, “Soft robotics: Biological inspiration, state of the art, and future research,” *Applied Bionics and Biomechanics*, vol. 5, 2008.

[7] C. Laschi, B. Mazzolai, V. Mattoli, M. Cianchetti, and P. Dario, “Design of a biomimetic robotic octopus arm,” *Bioinspiration and Biomimetics*, vol. 4, 2009.

[8] M. Cianchetti, C. Laschi, A. Menciassi, and P. Dario, “Biomedical applications of soft robotics,” *Nature Reviews Materials*, vol. 3, pp. 143–153, 2018. [Online]. Available: https://doi.org/10.1038/s41578-018-0022-y

[9] S. Shian, K. Bertoldi, and D. R. Clarke, “Dielectric elastomer based ”grippers” for soft robotics,” *Advanced Materials*, vol. 27, 2015.

[10] M. T. Gillespie, C. M. Best, and M. D. Killpack, “Simultaneous position and stiffness control for an inflatable soft robot,” in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 1095–1101.

[11] S. Sanan, “Soft inflatable robots for safe physical human interaction,” Ph.D. dissertation, Robotics Institute Carnegie Mellon University, Pittsburg, Pensilvania, 8 2013.

[12] P. Ohta, L. Valle, J. King, K. Low, J. Yi, C. G. Atkeson, and Y. L. Park, “Design of a lightweight soft robotic arm using pneumatic artificial muscles and inflatable sleeves,” *Soft Robotics*, vol. 5, 2018. [Online]. Available: https://www.liebertpub.com/doi/full/10.1089/soro.2017.0044

[13] S. Sanan, M. H. Ornstein, and C. G. Atkeson, “Physical human interaction for an inflatable manipulator,” in *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 2011, pp. 7401–7404.

[14] A. Stilli, H. A. Wurdemann, and K. Althoefer, “A novel concept for safe, stiffness-controllable robot links,” *Soft Robotics*, vol. 4, 3 2017.

[15] S. Sanan, J. B. Moidel, and C. G. Atkeson, “Robots with inflatable links,” in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, pp. 4331–4336.

[16] J. Tapia, E. Knoop, M. Mutný, M. A. Otaduy, and M. Bächer, “Makesense: Automated sensor design for proprioceptive soft robots,” *Soft Robotics*, vol. 7, 2020.

[17] Z. Kappassov, J. A. Corrales, and V. Perdereau, “Tactile sensing in dexterous robot hands - review,” *Robotics and Autonomous Systems*, vol. 74, 2015.

[18] S. E. Navarro, M. Marufo, Y. Ding, S. Puls, D. Göger, B. Hein, and H. Wörn, “Methods for safe human-robot-interaction using capacitive tactile proximity sensors,” in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 11 2013, pp. 1149–1154.

[19] J. Fishel and G. Loeb, “Bayesian exploration for intelligent identification of textures,” *Frontiers in Neurorobotics*, vol. 6, p. 4, 2012. [Online]. Available: https://www.frontiersin.org/article/10.3389/fnbot.2012.00004

[20] U. Martinez-Hernandez, T. J. Dodd, and T. J. Prescott, “Feeling the shape: Active exploration behaviors for object recognition with a robotic hand,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 48, 2018.

[21] X. Song, H. Liu, K. Althoefer, T. Nanayakkara, and L. D. Seneviratne, “Efficient break-away friction ratio and slip prediction based on haptic surface exploration,” *IEEE Transactions on Robotics*, vol. 30, 2014.

[22] H. Liu, X. Song, J. Bimbo, K. Althoefer, and L. Senerivatne, “Intelligent fingertip sensing for contact information identification,” in *Advances in Reconfigurable Mechanisms and Robots I*, 2012.

[23] M. I. Tiwana, S. J. Redmond, and N. H. Lovell, “A review of tactile sensing technologies with applications in biomedical engineering,” *Sensors and Actuators A: Physical*, vol. 179, pp. 17–31, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0924424712001641

[24] B. Winstone, “Towards tactile sensing active capsule endoscopy,” Ph.D. dissertation, University of West of England, Bristol, 11 2017. [Online]. Available: https://uwe-repository.worktribe.com/output/868181

[25] B. Winstone, G. Griffiths, T. Pipe, C. Melhuish, and J. Rossiter, “Tactip - tactile fingertip device, texture analysis through optical tracking of skin features,” in *Biomimetic and Biohybrid Systems*, N. F. Lepora, A. Mura, H. G. Krapp, P. F. M. J. Verschure, and T. J. Prescott, Eds. Springer Berlin Heidelberg, 2013, pp. 323–334.

[26] B. Ward-Cherrier, N. Pestell, L. Cramphorn, B. Winstone, M. E. Giannaccini, J. Rossiter, and N. F. Lepora, “The tactip family: Soft optical tactile sensors with 3d-printed biomimetic morphologies,” *Soft Robotics*, vol. 5, 2018.

[27] B. Winstone, C. Melhuish, T. Pipe, M. Callaway, and S. Dogramadzi, “Toward bio-inspired tactile sensing capsule endoscopy for detection of submucosal tumors,” *IEEE Sensors Journal*, vol. 17, 2017.

[28] H. Wang, M. Totaro, and L. Beccai, “Toward perceptive soft robots: Progress and challenges,” *Advanced Science*, vol. 5, 2018.

[29] P. Werner, M. Hofer, C. Sferrazza, and R. D’Andrea, “Vision-based proprioceptive sensing: Tip position estimation for a soft inflatable bellow actuator,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 8889–8896.

[30] R. Wang, S. Wang, S. Du, E. Xiao, W. Yuan, and C. Feng, “Real-time soft body 3d proprioception via deep vision-based sensing,” *IEEE Robotics and Automation Letters*, vol. 5, pp. 3382–3389, 4 2020.

[31] I. M. Van Meerbeek, C. M. De Sa, and R. F. Shepherd, “Soft optoelectronic sensory foams with proprioception,” *Science Robotics*, vol. 3, no. 24, 2018. [Online]. Available: https://robotics.sciencemag.org/content/3/24/eaau2489

[32] B. W. McInroe, C. L. Chen, K. Y. Goldberg, K. Y. Goldberg, R. Bajcsy, and R. S. Fearing, “Towards a soft fingertip with integrated sensing and actuation,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 6437–6444.

[33] M. Hofer, C. Sferrazza, and R. D'Andrea, "A vision-based sensing approach for a spherical soft robotic arm," *Frontiers in Robotics and AI*, vol. 8, p. 8, 2021. [Online]. Available: https://www.frontiersin.org/article/10.3389/frobt.2021.630935

[34] J. Oliveira, A. Ferreira, and J. C. Reis, "Design and experiments on an inflatable link robot with a built-in vision sensor," *Mechatronics*, vol. 65, 2020.

[35] N. Otsu, "Threshold selection method from gray-level histograms." *IEEE Trans Syst Man Cybern*, vol. SMC-9, 1979.

[36] K. Zuiderveld, *Contrast Limited Adaptive Histogram Equalization*. USA: Academic Press Professional, Inc., 1994, p. 474–485.

[37] P. R. Reddy, V. Amarnadh, and M. Bhaskar, "Evaluation of stopping criterion in contour tracing algorithms," *International Journal of Computer Science and Information Technologies*, vol. 3, 2012.

[38] M. T. Hagan, H. B. Demuth, M. H. Beale, and O. D. Jesús, *Neural Network Design*, 2nd ed., 2014. [Online]. Available: https://hagan.okstate.edu/NNDesign.pdf

[39] K. Buys, T. D. Laet, R. Smits, and H. Bruyninckx, "Blender for robotics: Integration into the leuven paradigm for robot task specification and human motion estimation," N. Ando, S. Balakirsky, T. Hemker, M. Reggiani, and O. von Stryk, Eds. Springer Berlin Heidelberg, 2010, pp. 15–25.

[40] F. Weichert, D. Bachmann, B. Rudak, and D. Fisseler, "Analysis of the accuracy and robustness of the leap motion controller," *Sensors (Switzerland)*, vol. 13, 2013.

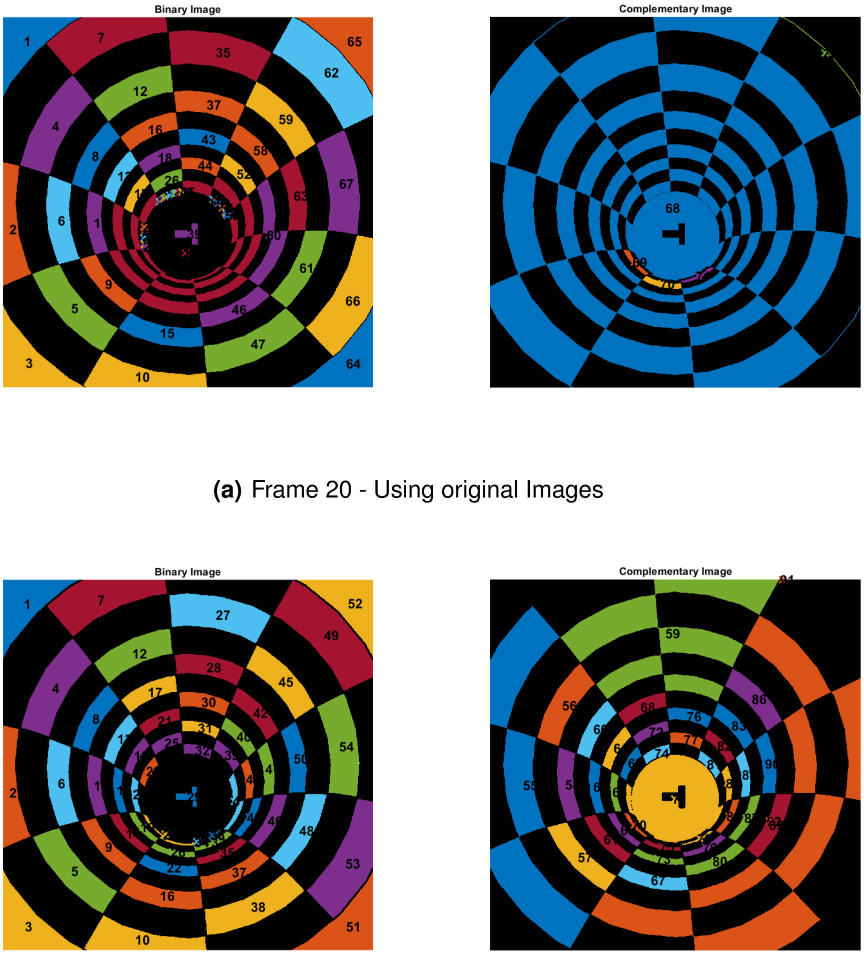
A

Additional Results

This appendix displays additional results of the developed work, covering the processing of patterns and bending and contact estimation.

A.1 Image Processing - Patterns

Figure A.1 illustrates the challenges of analysing the markers individually in the checkered pattern for Blob Segmentation. When no morphological operations are applied to the images (figure A.1(a)), it is difficult to distinguish the different markers, especially for accentuated deformations. If there is an application of morphological operations (figure A.1(b)), for example, erosion, the markers are more defined, but it might completely eliminate thinner markers at the tip of the soft link.

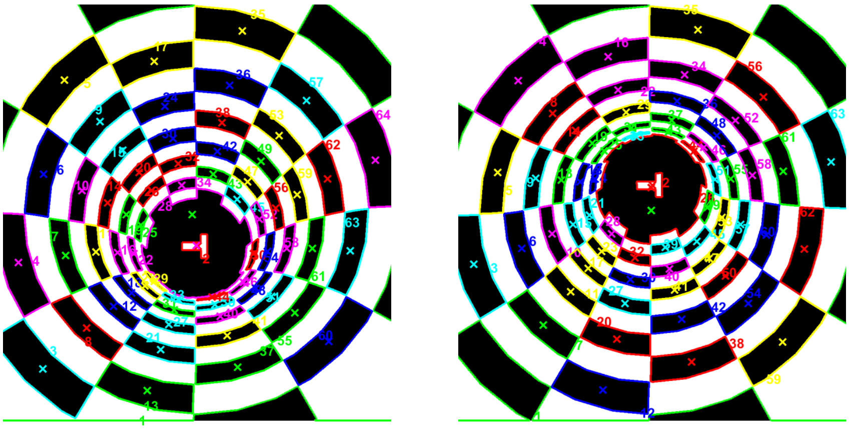


(a) Frame 20 - Using original Images

(b) Frame 20 - Applying morphological operation - Erosion

Figure A.1: Boundary Segmentation

Additional results for the Boundary Segmentation of the checkered pattern are displayed in figure A.2. It successfully identifies the black markers, but not the white.



(a) Frame20 **(b)** Frame 30

Figure A.2: Boundary Segmentation

Figures A.3 and A.4 show a type of image used to train the NN for occlusion of markers, with Blob and Boundary Segmentations, respectively.

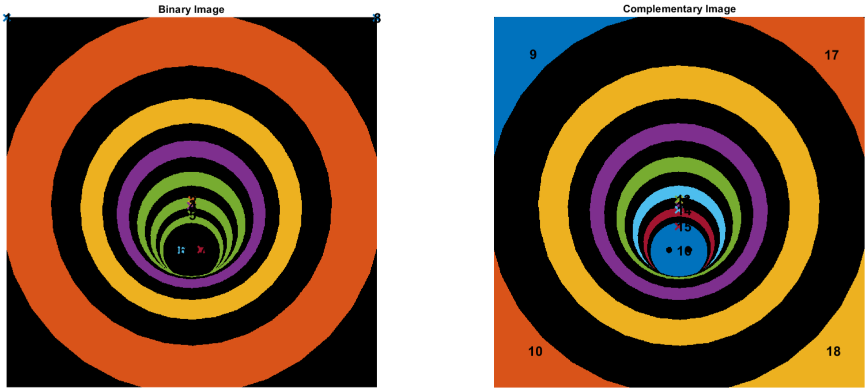


Figure A.3: Occlusion of markers - Blob Segmentation

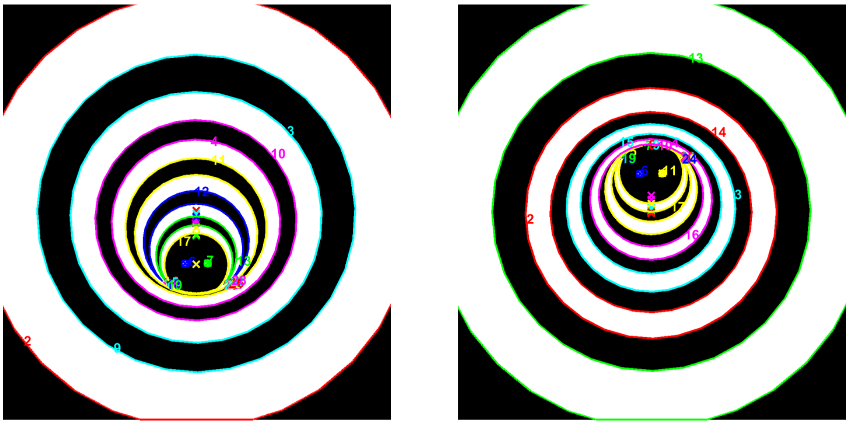
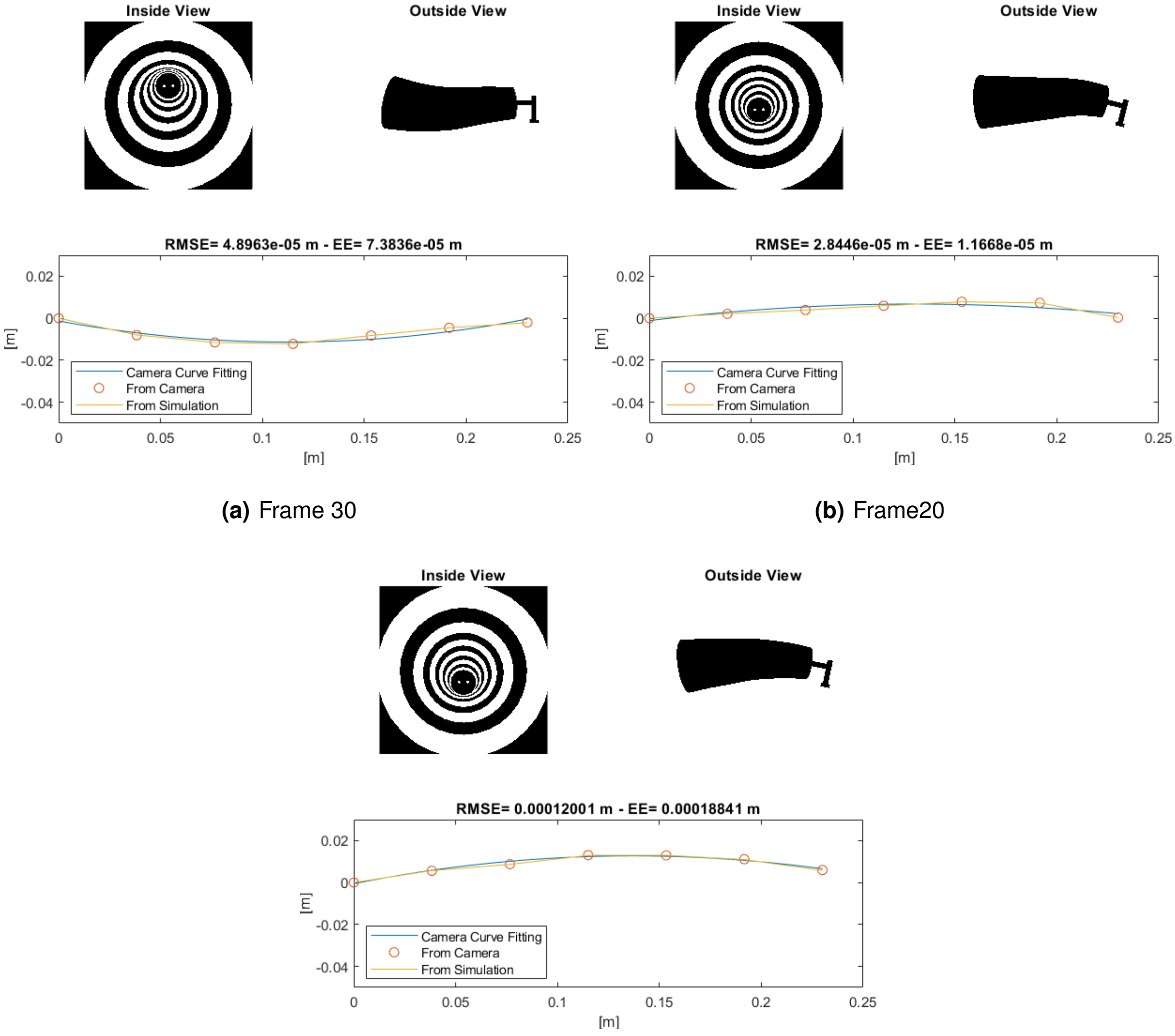


Figure A.4: Occlusion of markers - Boundary Segmentation

A.2 Bending Results

Additional results are portrayed in figures A.5 and A.6, for the NN created for single camera view and stereo camera view, showing more variety of poses of the soft link.

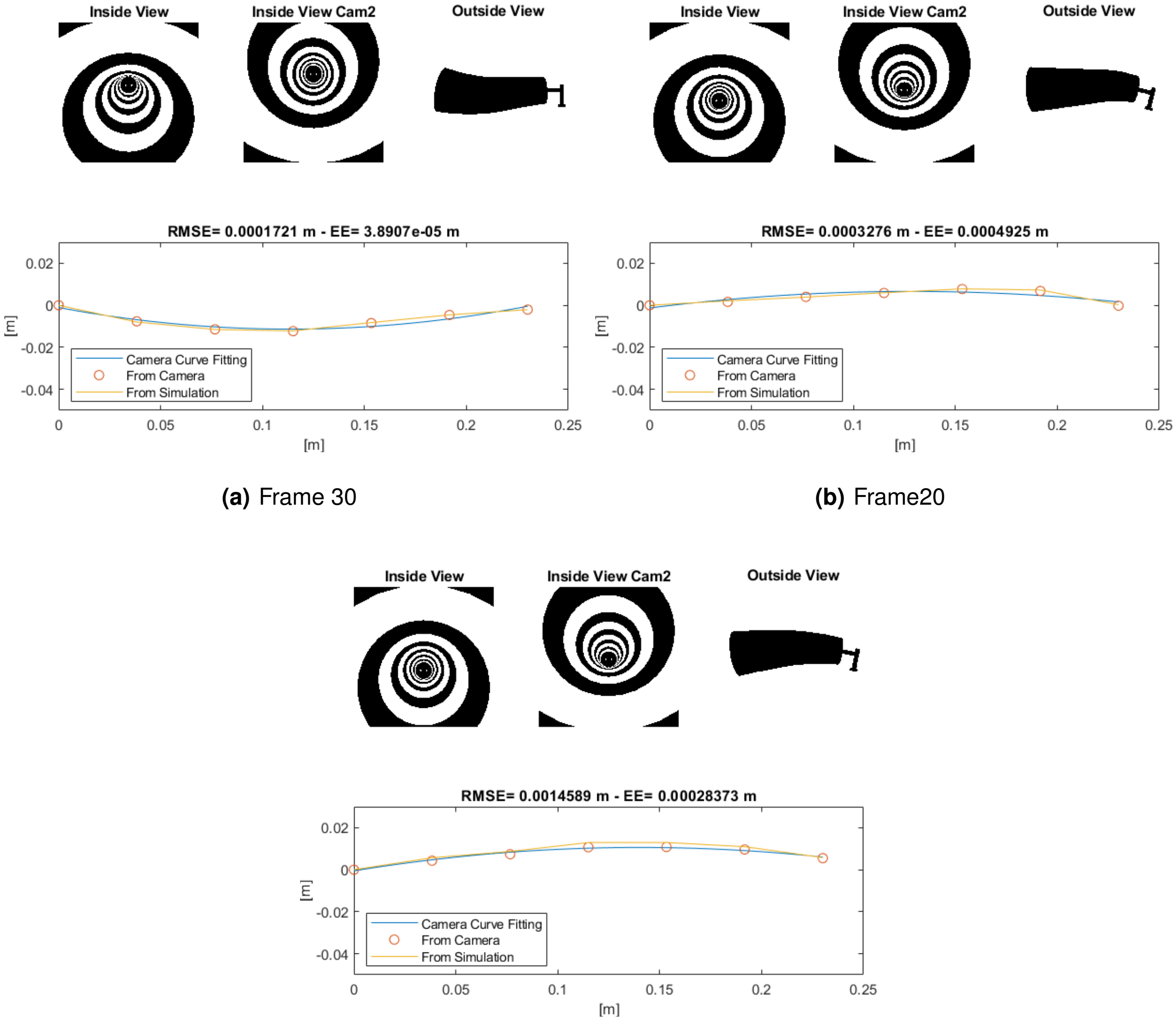


(a) Frame 30

(b) Frame20

(c) Frame20

Figure A.5: Bending Estimation - Additional Results



(a) Frame 30

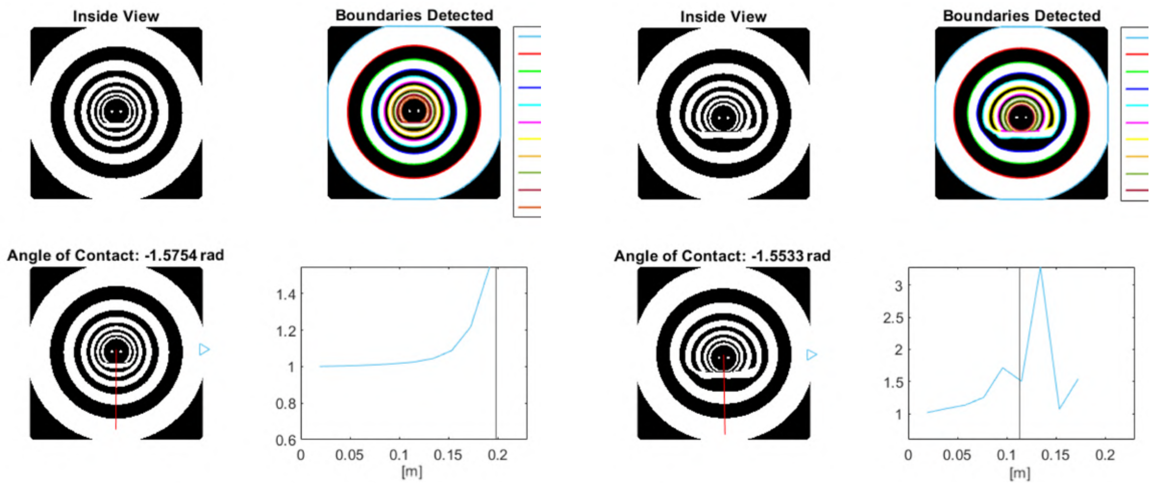
(b) Frame20

(c) Frame20

Figure A.6: Bending Estimation Stereo Set - Additional Results

A.3 Contact Results

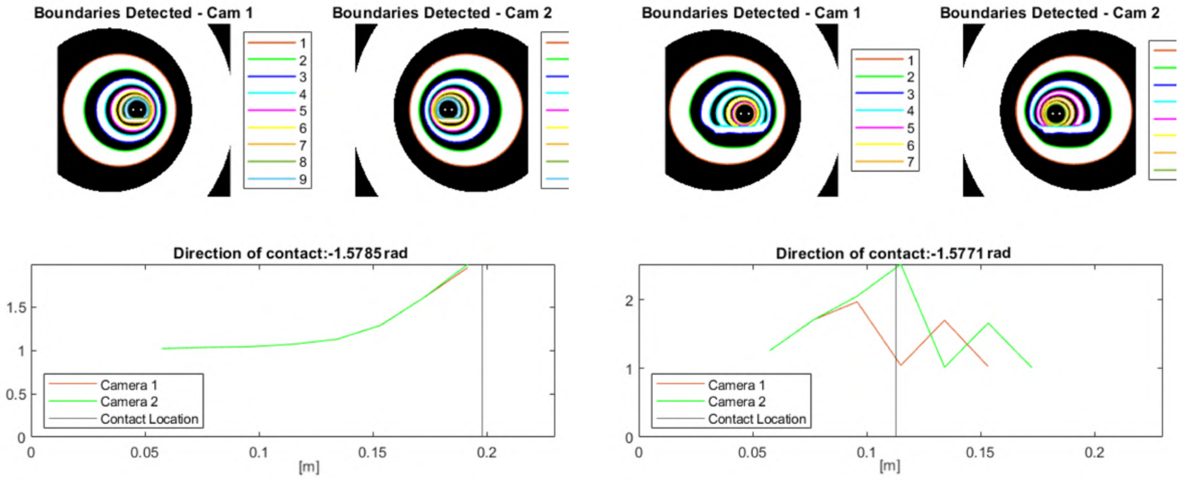
Additional results covering the effects of occlusion of markers for the contact estimation for single camera view and stereo camera view are portrayed in figures A.7 and A.8.



(a) Definition of a closer contact point to the base

(b) Partially hidden marker surpassing real contact point

Figure A.7: Contact Estimation - Single Camera



(a) Definition of a closer contact point to the base

(b) Different Accuracy from both images

Figure A.8: Contact Estimation - Stereo Set