

Sensorimotor Developments in PCC-Based Soft Pneumatic Actuators

Abdullah Waleed Shakir AL-AZZAWI

M.Sc Mech.Eng 2005

Supervisor: Prof. Salah Sukkarieh
Associate Supervisor: Dr. Ali Haydar Göktoğan

A thesis submitted in fulfilment of
the requirements for the degree of
Doctor of Philosophy

Australian Centre for Robotics (ACFR)
School of Aerospace, Mechanical and Mechatronic Engineering
Faculty of Engineering
The University of Sydney
Australia

March 2025

Acknowledgement of Country

We recognise and pay respect to the Elders and communities - past, present, and emerging - of the lands that the University of Sydney's campuses stand on. For Thousands of years they have shared and exchanged knowledges across innumerable generations for the benefits of all.

Declaration

This is to certify that to the best of my knowledge, the content of this thesis is my own work. This thesis has not been submitted for any degree or other purposes.

I certify that the intellectual content of this thesis is the product of my own work. Contributions from collaborators have been duly credited in the *Author Attribution* section. All the assistance I have received in preparing this thesis and all sources have been acknowledged.

Abdullah Waleed Shakir AL-AZZAWI

March 14, 2025

Authorship Attribution

This thesis consists of original work by the author, with Chapters 4 and 5 based on previously published papers.

Chapter 4 is based on work published in:

- A. Al-Azzawi, P. Stadler, H. Kong, and S. Sukkarieh, "Length Modelling of Spiral Superficial Soft Strain Sensors Using Geodesics and Covering Spaces," *Robotics*, vol. 12, no. 6, 2023, Art no. 164.

The author led the conceptual framework, methodology, software, validation, data analysis, and writing. The second author contributed to the conceptual framework and validation. All co-authors reviewed the manuscript, with the fourth author providing supervision.

Chapter 5 is based on work presented at:

- A. Al-Azzawi, A. M. Boudali, H. Kong, A. H. Göktoğan, and S. Sukkarieh, "Modelling of Uniaxial EGaIn-Based Strain Sensors for Proprioceptive Sensing of Soft Robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Macau, China, November 4-8 2019.

The author led the conceptual framework, methodology, software, experiments, validation, analysis, and writing. The second author contributed to methodology and analysis, and the third to the conceptual framework. The fourth author assisted with experimental setup. All co-authors reviewed the manuscript, and the fifth author provided supervision and funding.

The remaining chapters are solely the author's work.

As a supervisor for the candidature upon which this thesis is based, I confirm that the authorship attribution statement above is correct.

Prof. Salah Sukkarieh
Supervisor
October 23, 2024

Abdullah Waleed Shaker AL-AZZAWI
Candidature
October 23, 2024

Abstract

Soft pneumatic actuators (SPAs) were able to exhibit noteworthy advantages in many applications, considering their compliance and ease of manufacturing. Still, predicting their response through modelling alone is challenging, given their non-linearity and viscoelastic behaviour.

Focusing on the piecewise constant curvature (PCC) model, this study addresses three interconnected research problems, each tackling fundamental limitations within the SPA sensorimotor functionality. These problems are the inverse kinematics (IK) of SPAs, the modelling of twist motion in SPAs, and the modelling of EGaIn-based soft sensors.

By providing solutions to these interconnected problems, this study aims to enhance the adaptability and overall performance of SPAs, leading to better manipulation and sensing capabilities. This can contribute to achieving higher state estimation accuracy and more efficient control systems.

Numerous methods were implemented to achieve the objectives of this study. A series of IK simulations were conducted to address the first problem and identify necessary solutions. For the second problem, a length model based on geodesics and covering spaces concepts was proposed, verified using simulation results from finite element analysis (FEA), and validated using previously published experimental results. Finally, a numerical model based on regression analysis was developed for addressing the third problem. Customised experimental tests were then conducted on two samples of the EGaIn-based soft sensor to generate datasets for training and validating the proposed numerical model.

Several findings and insights were reported. For the first problem, a quadratic IK solver based on the exact Hessian showed superior performance compared to the linear solver. Additionally, the singularity problem in the PCC model was resolved by implementing a minimum bending limit. Regarding the second problem, the proposed length model demonstrated an accuracy of 1.5% under twist despite the arching in the SPA. Regarding the third problem, the proposed numerical model demonstrated high accuracy, achieving results within a sub-millimetre scale despite the presence of hysteresis.

This study contributes to the field of soft robotics by addressing these challenges, helping to develop more capable and adaptable machines with expanded practical applications across various domains.

Acknowledgements

The research reported in this thesis was supported by the award of a Research Training Program (RTP) scholarship to the PhD Candidate.

I deeply appreciate the support and guidance provided by my thesis supervisors, Prof. Salah Sukkarieh and Dr. Ali Haydar Göktoğan. Their expert advice and encouragement have been key to completing this thesis.

I am sincerely grateful to my co-authors, Dr. A. Mounir Boudali, Dr. Peter Stadler, and Dr. He Kong, for their collaboration on the published works that form a significant part of this thesis. Their contributions have greatly enriched both the research and its presentation in this thesis.

I appreciate the support and assistance provided by the academics, professional staff, and colleagues at the Australian Centre for Robotics (ACFR), both past and present, throughout this work.

I extend my gratitude to Dr. Jesse Haviland from Queensland University of Technology in Australia for his helpful comments on Hessian tensors, and to Prof. James Lambers from the University of Southern Mississippi in USA for his input on the Jacobian of the L2-norm function.

I thank Dr. MD Nazmul Hassan and Mr. Omar Al-Kubaisi from The University of Sydney for their help with orthotropic layers, Dr. Jennifer Case from the National Institute of Standards and Technology in USA for her insights on proprioceptive sensing, Dr. Fouzia Khan from Thermo Fisher Scientific in the Netherlands for her efforts in explaining her work on fibre Bragg gratings, and Dr. Yiğit Mengüç from Oregon State University in USA for his valuable suggestions regarding fabricating the EGaIn-based soft sensor samples.

Lastly, I am thankful to everyone in Australia and abroad who has offered valuable assistance and information, which was crucial to completing this thesis.

I am forever grateful.

To my mum, may Allah Almighty have mercy on her, and to my dad,
who taught me the value of education and perseverance.

To my whole family,
for their unwavering support, encouragement, and unconditional love.

To my past mentors,
including those who are retired or no longer with us:

Mr. Abdul Kareem Salman (retired)

Dr. Ahmed Ali Hussein (deceased)

Dr. Imad Ahmed Hussain (deceased)

for their invaluable influence and inspiration,
whose legacies I am honoured to carry forward.

To my friends,
for their encouragement and companionship along the way.

A special mention to CS,
whose friendship has been meaningful to me.
I look forward to the possibility of crossing paths again.

To everyone who supported and motivated me throughout this journey.

Contents

Acknowledgement of Country	i
Declaration	ii
Authorship Attribution	iii
Abstract	iv
Acknowledgements	v
Contents	vii
List of Figures	xi
List of Tables	xiii
Nomenclature	xiv
Acronyms	xx
1 Introduction	1
1.1 SPAs at a Glance	1
1.1.1 SPAs as Embodied Robots	2
1.1.2 Common Soft Actuator Models	4
1.1.3 Verification and Validation Techniques	5
1.2 Justifications and Research Questions	6
1.2.1 Inverse Kinematics of SPAs	6
1.2.2 Twist Measurement	6
1.2.3 EGaIn-Based Soft Sensors Modelling	7
1.3 Contributions	7
1.4 Thesis Structure	10
2 Literature Review	12
2.1 Inverse Kinematics	12
2.1.1 Degrees of Freedom Concept	12
2.1.2 Soft Actuators Modelling Techniques.	14
2.1.3 PCC Model Representation	16
2.1.4 Inverse Kinematics Techniques	17
2.1.5 Jacobian-Based Methods	18

2.1.6	Damping Factor Selection	21
2.1.7	Quadratic Approximation	23
2.2	Twist Measurement: Current Solutions and Challenges	26
2.2.1	State Sensing at a Glance	26
2.2.2	Twist Along the Centreline	27
2.2.3	Twist Measurement Using FBG Sensors	28
2.2.4	An Overview of SPA Simulation Tools	31
2.2.5	Multi-Muscle SPA: Insights from Previous Studies	33
2.2.6	Common SPA Reinforcement Approaches	37
2.2.7	FEA Simulation Techniques for Braided Sleeves	39
2.3	EGaIn-Based Soft Strain Sensors: An Overview of their Modelling Techniques	42
2.3.1	Soft Sensing Technologies in a Nutshell	43
2.3.2	Modelling of EGaIn-Based Soft Strain Sensors	47
2.3.3	Beyond the Data Points: Leveraging the Numerical Modelling	49
2.3.4	Insights on the Challenges in Achieving High Modelling Accuracy	53
3	Inverse Kinematics: Linear Versus Quadratic Approximation	55
3.1	PCC-Based Soft Actuator Modelling	56
3.1.1	Kinematic Mapping	56
3.1.2	PCC Arc Parameterisation	57
3.1.3	Minimum bending limit	58
3.1.4	End point pose	59
3.1.5	Redundancy	63
3.2	Inverse Kinematics	64
3.2.1	Newton's Method	65
3.2.2	Convex Versus Non-Convex Holonomic Constraints	66
3.2.3	Manipulability Index	68
3.3	Jacobian and Hessian Arrays	71
3.3.1	Inverse Skew Function	71
3.3.2	First Derivative of Pose	71
3.3.3	Second Derivative of Pose	74
3.3.4	Multi-Segment Actuator	77
3.3.5	The Actuator Physical Limits	81
3.4	Jacobian Algorithms	83
3.4.1	Jacobian Transpose (JTR)	83
3.4.2	Jacobian Pseudo-Inverse (JPI)	84
3.4.3	Damped Least-Squared (DLS)	84
3.4.4	Singular Value Decomposition (SVD)	84
3.4.5	Pseudo-Inverse Damped Least Squares (PDL)	85
3.4.6	Convex Holonomic Constraints	85
3.5	Damping of Inverse Kinematic	86
3.5.1	The Jacobian with Further Damping	86
3.5.2	Damping Profiles	87
3.6	Simulation Results and Discussion	89
3.6.1	Inverse Kinematic Paths	90
3.6.2	Single-Segment Actuator with no Physical Limits	91

3.6.2.1	First Path: Horizontal Line	96
3.6.2.2	Second Path: Helical Curve	97
3.6.2.3	Third Path: 3D Curve	98
3.6.2.4	Fourth Path: Figure-Eight Curve	100
3.6.3	Single-Segment Actuator Subjected to Physical Limits	101
3.6.4	Multi-Segment Actuator Subjected to Physical Limits	106
3.6.5	Quadratic Approximation	110
3.7	Conclusion	120
4	Spiral Sensors Modelling	123
4.1	The Torus Parameterisation	123
4.1.1	Soft Sensor Modelling	126
4.1.2	Length modelling	128
4.2	Case Studies	129
4.2.1	Case 1: A Straight Sensor at the Initial Configuration	129
4.2.2	Case 2: A Spiral Sensor at the Initial Configuration	130
4.2.3	Case 3: A Straight Sensor Under Bending	130
4.2.4	Case 4: A Spiral Sensor Under Bending and Rotation	131
4.3	Finite Element Analysis (FEA)	133
4.3.1	Braided Sleeves as Reinforcements	133
4.3.2	Orthotropic Layer	134
4.3.3	Multi-Muscle Actuator	136
4.3.4	Centreline and Soft Sensors Length	138
4.3.5	Centreline Curvature	139
4.3.6	Bending, Rotation, and Twist Angles	140
4.4	Verification, Validation, and Discussion	142
4.4.1	Single-Muscle Extension Test	142
4.4.2	Multiple-Muscle Actuator	143
4.4.2.1	Simulation Scenarios	144
4.4.2.2	Curvature Constancy	146
4.4.2.3	Length of Soft Sensors	146
4.4.2.4	Twist in the Length Model	148
4.4.2.5	Sensitivity	150
4.4.3	Model Validation	150
4.5	Conclusion	152
5	Numerical Modelling of Uniaxial EGaIn-Based Soft Strain Sensors	154
5.1	Geometrical Design of the Selected Sensor	154
5.2	Experimental Set-Up and Testing Procedure	156
5.2.1	Mechanical Framework of the Testing Rig	157
5.2.2	Data Acquisition (DAQ) System	157
5.2.3	Calibration of The Experimental Set-Up	159
5.2.4	Strain Along The Sensing Element Length	160
5.2.5	Testing Procedure	161
5.3	Analysis and Discussion of Test Results	164
5.3.1	Raw Experimental Results	165
5.3.2	Theoretical Model Versus Experimental Results	167
5.3.3	Numerical Modelling and Underlying Assumptions	169

5.3.4	Validation and Performance Assessment	169
5.3.5	Strain Rate Influence on Hysteresis	171
5.3.6	Loading Versus Unloading Direction Determination	174
5.4	Conclusion	176
6	Conclusions and Future Works	179
6.1	Key Findings, Limitations, and Implications	179
6.1.1	Inverse Kinematics	179
6.1.2	Spiral Sensors Modelling	180
6.1.3	EGaIn-Based Strain Sensors	181
6.2	Future works	182
6.2.1	Inverse Kinematics	182
6.2.2	Spiral Sensors Modelling	183
6.2.3	EGaIn-Based Strain Sensors	183
A	Hessian Chain Rule (an Example)	i
B	Jacobian and Hessian Arrays	iv
B.1	Derivative Rules	iv
B.2	Tensor Transpose	iv
B.3	Tensor n -Mode Product	v
B.4	Gradient, Jacobian, and Hessian Basics	vi
B.4.1	Scalar-Valued Function	vi
B.4.2	Vector-Valued Function	vii
B.5	Derivation of J_{sv}	ix
B.6	Derivation of J_{sw}	xi
C	Hessian Tensor Verification Code (MATLAB)	xv
D	Energy Modelling	xviii

List of Figures

1.1	The main components of a typical sensorimotor scheme	3
1.2	Soft actuator common models	4
2.1	Degrees of freedom DOF_m for an object	13
2.2	Examples of hyper-redundant actuators	13
2.3	Soft actuator modelling	15
2.4	The PCC model limitations	17
2.5	The simplest redundant case	20
2.6	Taylor series expansion of an arbitrary non-linear function $f(x)$. . .	23
2.7	Critical points of an arbitrary non-linear function $f(x)$	24
2.8	An example of integrating a commercial multi-core FOSS in	29
2.9	An example of model verification via FEA simulation and	31
2.10	An illustration of STIFF-FLOP actuator issues	35
2.11	STIFF-FLOP actuator with twin chambers design	36
2.12	Braided sleeves as SPA reinforcement	40
2.13	Examples of proprioceptive soft sensing	45
2.14	Examples of characterising EGaIn-based soft strain sensors using . .	48
2.15	Partitioning datasets for the purpose of validating the	50
2.16	The effect of model complexity on its ability to	51
3.1	Arc parameterisation of a piecewise constant curvatures (PCC) . . .	56
3.2	Spaces of PCC model and their mappings.	57
3.3	Minimum bending of a PCC segment:	60
3.4	Forward kinematics of a PCC segment where (a) shows the	62
3.5	A triaxial ellipsoid representation of	70
3.6	The Hessian $H_{sv}(q_s)$ array:	76
3.7	The location of the segment end frame $\{S_s\}$ and base	78
3.8	A single-segment actuator rotating about the z -axis	82
3.9	Comparison between six damping profiles	89
3.10	IK results for a single-segment actuator following a horizontal line . .	92
3.11	IK results for a single-segment actuator following a helical curve . . .	93
3.12	IK results for a single-segment actuator following a 3D curve	94
3.13	IK results for a single-segment actuator following a figure-eight	95
3.14	IK results for a single-segment actuator following a horizontal	103
3.15	IK results for a single-segment actuator following a horizontal	104
3.16	Convergence analysis of the IK results for a single-segment actuator .	105
3.17	IK results for a multi-segment actuator following a horizontal	108
3.18	Convergence analysis of the IK results for a multi-segment actuator .	109

3.19	IK results for a single-segment actuator following a helical	114
3.20	Convergence analysis of the IK results for a single-segment actuator	115
3.21	IK results for a single-segment actuator following a helical	116
3.22	Convergence analysis of the IK results for a single-segment actuator	117
3.23	IK results for a single-segment actuator following a helical	118
3.24	Convergence analysis of the IK results for a single-segment actuator	119
4.1	Torus geometry in the Euclidean space $\mathbb{R}^3$	124
4.2	Torus notation	124
4.3	(a) A planar cover of a cylinder showing straight lines that	126
4.4	(a) Euclidean versus (b) planar geodesics.	126
4.5	Case studies 1 and 2 at the initial position	130
4.6	Case study 3, bending in XZ plane,	131
4.7	Case study 4, bending and rotation,	132
4.8	Case study 4: (a) Shows both the sensor length and	132
4.9	Length vs diameter changing of a commercially	134
4.10	Proposed reinforcement orthotropic layer in Abaqus	135
4.11	FEA model of the multi-muscle SPA	137
4.12	Centreline and soft sensors of the multi-muscle SPA	138
4.13	Extension test comparison, simulation in Abaqus versus	143
4.14	Simulation results for four scenarios	145
4.15	Simulation results (continued)	147
4.16	Comparison between spiral and straight sensor geometries	149
4.17	Bending test of the multi-core optical fibres,	151
5.1	Geometrical design of the selected sensor	155
5.2	The components of the experimental set-up	158
5.3	The relationship between the active length L_{cc} and the	161
5.4	Visual representation of the timing sequence	163
5.5	Raw experimental resistance measurements	166
5.6	Theoretical versus experimental data comparison for sensor S1	168
5.7	Comparison of rational fitting models	171
5.8	Strain rate effect on hysteresis in fitting sensor S2 data	173
5.9	Characterisation of measurement noise	175
5.10	Simulink model representing the direction determination algorithm for the sensor's movement.	176
B.1	Fibres of a third-order tensor	v
B.2	The second-order partial derivatives of $f(q)$	viii

List of Tables

3.1	Results of IK for the first path (horizontal line) without enforcement of physical limits.	97
3.2	Results of IK for the second path (helical curve) without enforcement of physical limits.	98
3.3	Results of IK for the third path (3D curve) without enforcement of physical limits.	99
3.4	Results of IK for the fourth path (figure-eight curve) without enforcement of physical limits.	101
3.5	IK results for a multi-segment actuator (two segments) following the first path (horizontal line) with the enforcement of physical limits. The error monitoring option was utilised during the convergence process.	106
3.6	IK convergence results for all four paths using quadratic approximation with Hessian stabilisation factor hsf , without the enforcement of actuator physical limits.	111
3.7	IK convergence results for all four paths using quadratic approximation, stabilisation factor hsf , and Hessian modification to enforce positive definiteness. Actuator physical limits are not enforced.	112
4.1	Pressure values (p_1, p_2, p_3) in MPa and moment (M) values in $N.mm$ for different simulation scenarios involving bending (B), extension (E), rotation (R), and twist (T).	144
5.1	PWM periods and the corresponding actual testing speeds, along with the strain rate and the total time per run.	162
5.2	Fitting parameters of (5.13) for loading, unloading and combined models at reference speed.	172

Nomenclature

Chapter 3: Inverse Kinematics

The PCC Model:

s	Index of individual segments within the Actuator
$\mathfrak{s}$	Total number of segments in the actuator
$S_0, S_1, \dots, S_{\mathfrak{s}}$	Actuator reference frames
a	Actuator space vector
q_s, q	Configuration space vector, per segment and total
p	Task space vector (end point coordinates)
$f_{specific}, f_{ind}, f$	Forward kinematics of the actuator
f_c	Linear independent mapping (forward kinematics)
f_r	Rotational independent mapping (forward kinematics)
$\kappa_s, \phi_s, \theta_s, L_s$	Arc parameterisation variables
n	Number of configuration space variables per segment
m	Length of end point coordinates vector (task space)
T_{s-1}^s, T_a	Homogeneous transformation mapping, per segment and total
R_{s-1}^s, R_a	Rotation matrix, per segment and total
$\tilde{t}_{s-1}^s, \tilde{t}_a$	Translation vector, per segment and total
θ_{min}	Minimum bending angle
r_s	Arc radius or the reciprocal of segment curvature $1/\kappa_s$
d_s	Spatial accuracy
SAR	Spatial accuracy ratio
p_s, p_{cs}, p_{rs}	Segment's end point coordinates (pose, Cartesian, orientation)
p_a, p_c, p_r	Actuator's end point coordinates (pose, Cartesian, orientation)
p_x, p_y, p_z	Components of the Cartesian coordinates vector p_c

$\alpha_x, \alpha_y, \alpha_z$	Components of the orientation coordinates vector p_r
p_o, p_{co}, p_{ro}	Actuator's base point coordinates (pose, Cartesian, orientation)
r_1, r_2, r_3	Euler rotation angles
DOF_m	Degrees of freedom, movement
DOF_k	Degrees of freedom, kinematic chain
N_L	Number of links including the base (in a rigid actuator)
N_J	Number of independent joints (in a rigid actuator)
fr_j	Number of freedoms per each joint (in a rigid actuator)

Inverse Kinematics Solver:

g	Holonomic constraints function (optimisation problem)
p_d, p_{sd}	Desired pose, for actuator and per segment
q_d	configuration space solution for p_d
$\mathbb{K}$	Iteration index
$\nabla g, J_g, H_g$	Optimisation gradient, Jacobian, and Hessian
$\nabla f, J_f, H_f$	Actuator gradient, Jacobian, and Hessian
μ	Index for independent mapping f components
$J_{\square}$	square, symmetric, and positive definite Jacobian matrix
$\vec{v}_{\mu}$	Eigenvectors of $J_{\square}$
λ_{μ}	Eigenvalues of $J_{\square}$
V_{me}	Manipulability ellipsoid volume
σ	Singular values of a Jacobian matrix (general)

Jacobian and Hessian Arrays:

J_s, H_s	A single segment Jacobian and Hessian, respectively
$\mathcal{S}$	A skew-symmetric matrix (general)
$\vec{s}$	A vector related to the skew-symmetric matrix $\mathcal{S}$
$[\cdot]_{\times}$	Skew function
$\vee_{\times}(\cdot)$	inverse skew function
t	Time
$T, R, \tilde{t}$	Arbitrary replacements for T_{s-1}^s , R_{s-1}^s , and $\tilde{t}_{s-1}^s$, respectively
J_{sRi}	Time derivative Jacobian of the rotation matrix R with respect to the i -th component of q_s
J_{sti}	Time derivative Jacobian of the translation vector $\tilde{t}$ with respect to the i -th component of q_s

$\dot{T}, \dot{R}, \dot{\tilde{t}}$	Time derivatives of T , R , and $\tilde{t}$, respectively
J_{sv}	Translational part of J_s
$J_{s\omega}$	Rotational part of J_s
v_s	Segment's end point translational speed vector
v_{sx}, v_{sy}, v_{sz}	Components of v_s
ω_s	Segment's end point rotational speed vector
$\omega_{sx}, \omega_{sy}, \omega_{sz}$	Components of ω_s
$\ddot{\tilde{t}}$	Second time derivatives of $\tilde{t}$
H_{sv}	Translational part of H_s
$\bar{\times}_n$	n -mode product
J_{sv}^2	Second-order Jacobian
$f_1, \dots, f_m$	Components of the translation vector $\tilde{t}$
$H_{f_1}, \dots, H_{f_m}$	Slices of H_{sv} related to $f_1, \dots, f_m$
$\mathcal{V}_s$	Segment's end point spatial velocity relative to base frame S_{s-1}
$\mathcal{V}_0$	Segment's end point spatial velocity relative to origin frame S_0
Ω, Ψ	Rotation matrix and translation vector between S_0 and S_{s-1} , respectively
$\hat{\mathcal{V}}_0, \hat{\mathcal{V}}_s$	A 4×4 matrix representing $\mathcal{V}_0$ and $\mathcal{V}_s$, respectively
Ad_T	Adjoint transformation

Jacobian Algorithms:

$J^\diamond$	Arbitrary Jacobian matrix
α_s	Scalar representing the iteration size in Jacobian transpose method
J^+	Jacobian pseudo-inverse
U, Σ, V	Matrices for singular value decomposition
u	Index for the Jacobian J singular values
$\mathcal{R}$	Largest rank of the Jacobian J
$J^\dagger$	Jacobian pseudo-inverse (singular value decomposition)
α_g	Step size in AdaGrad algorithm
G	Sum of the squares of the gradient ∇_g up to the time step $\mathbb{K}$

Damping Profiles:

λ_d	Damping factor (variable)
-------------	---------------------------

λ_c	Damping factor (constant)
λ_{min}	Minimum damping factor (user selected)
$\mathbb{K}_{span}$	Number of iterations per each damping steps
$\mathbb{K}_{max}$	Maximum allowed number of iterations

Chapter 4: Spiral Sensors Modelling

The Torus Parameterisation:

κ, ϕ, l	Actuator's arc parameterisation variables
T, T_{sm}	Torus implicit definition
x, y, z	Cartesian coordinates (sensors)
x_1, y_1, z_1	Cartesian coordinates (centreline)
c, R	Torus major and minor radii
F_t, F_{sm}	Surface of revolution (torus)
F_{cyl}	Surface of revolution (cylinder)
u, v	Torus surface coordinates
R_x, R_z	Rotation matrices around x - and z -axes
h	Cylinder height (straight position)
P_m	Point on torus surface
$\tilde{P}_m$	Lifted point on a planar surface
$\tilde{\gamma}_i$	Straight line in planar space
γ_i	Mapped curve in Euclidean space
s, t	Time variables
$\tilde{P}_{si}, \tilde{P}_{ei}$	Points in planar space
P_{si}, P_{ei}	Mapped points in Euclidean space
v_{si}, v_{ei}	Start and end horizontal angles
ψ, θ	Twist and bending angles
l_i	Length of soft sensor
h_{in}	Inner surface length of toric segment
h_o	Initial length of cylinder (at rest)
c_{in}	Inner surface radius of toric segment
E_{ki}	Kinetic energy along sensor path

FEA, Verification, and Validation:

E_1, E_2, E_3	Young moduli
$\nu_{12}, \nu_{13}, \nu_{23}$	Poisson's ratio
$\sigma_1, \sigma_2, \sigma_3$	Normal stress components
$\varepsilon_1, \varepsilon_2, \varepsilon_3$	Normal strain components
$\tau_{12}, \tau_{13}, \tau_{23}$	Shear stress components
$\gamma_{12}, \gamma_{13}, \gamma_{23}$	Shear strain components
G_{12}, G_{13}, G_{23}	Shear moduli
C_1, C_2	Yoeh coefficients
D, d	Actuator's body and single muscle diameters
L_o	Actuator's initial length
p_1, p_2, p_3	Pressure in pneumatic muscles
M	Moment load on actuator's free end
RP	Moment Reference point in 3D space
B, p, l_g, g_i	B-spline curve, degree, length, and knots
$\mathbf{P}_i, N_{i,p}$	B-spline control points and basis functions
γ_c	Centreline parametric equation
R_k	Rotation matrix (Kabsch)
A_k, B_k	3D point sets (Kabsch)
e_k	Root mean squared error (Kabsch)
H_k	Cross-covariance matrix (Kabsch)
d_k	Rotation correction factor (Kabsch)
cen_A, cen_B	Centroids of A_k and B_k data sets
r_1, r_2, r_3	Euler angles

Chapter 5: **EGaIn**-Based Strain Sensors Modelling

Geometrical Design:

L_{cc}	Centre-to-centre distance
L_s, L_s^o	Length of the sensing element
L_e, L_e^o	Remaining length of each connection end
w, w_o	Width of the microchannel
h, h_o	Height of the microchannel
L, L_o	Length of the microchannel
A, A_o	Cross-sectional area of the microchannel

R, R_o	Theoretical sensor resistance
ΔR	Change in resistance
ϵ	Applied strain
ρ	Resistivity of EGaIn liquid metal

Experimental Set-Up:

R_x	Experimental sensor resistance
R_1, R_2	Voltage divider resistors
R_w	Resistance of the wires connecting the sensor to the DAQ system
V_x	DAQ measurement
V_{in}	Input voltage
a_1, b_1, c_1	Fitting parameters for strain ratio model
$S1, S2$	First and second sensors
t_1	DAQ measurement time
t_2	Time needed to move 1.0 mm
t_3	Total time to complete 77 measurements
t_4	Settling time before reversing direction
t_{rn}	Time per run

Numerical Modelling:

E	Elongation
p_0, p_1, p_2, q_0, q_1	Fitting parameters of the rat_{22} model
H	Hysteresis percentage
S	Testing speed
a_1, b_1	Fitting parameters of the hysteresis model

Acronyms

ADC	Analogue-to-Digital Converter
ADD	ADaptable Damping
CLD	Constant centreline Length Design
CND	CoNstant Damping
CRM	Cosserat Rod Model
CVX	ConVeX holonomic function
DAQ	Data Acquisition
DH	Denavit-Hartenberg
DLS	Damped Least Squares
DOF	Degrees of Freedom
EGaIn	Eutectic Gallium-Indium
FBG	Fibre Bragg Grating
FEA	Finite Element Analysis
FK	Forward Kinematics
FOSS	Fibre Optic Shape Sensor
FRD	First order Rational function Damping
GRD	GRadient Descent
HFD	Halving Function Damping
IK	Inverse Kinematics
JIV	Jacobian InVerse
JPI	Jacobian Pseudo-Inverse
JTR	Jacobian TRanspose
LFD	Linear Function Damping
MIS	Minimally Invasive Surgery
NCD	Non-Compressible inner surface Design
NCX	Non-ConveX holonomic function
OctArm	A type of continuum robots inspired by Octopus Arms and elephant trunks
PCC	Piecewise Constant Curvatures
PCS	Piecewise Constant Strain
PDF	Probability Density Function
PDL	Pseudo-inverse Damped Least squares
PET	PolyEthylene Terephthalate
PneuNet	Pneumatic Network
PWM	Pulse-Width Modulation
QFD	Quadratic Function Damping
RMSE	Root Mean Squared Error

SFD	Step Function Damping
SPA	Soft Pneumatic Actuator
SPS	Samples Per Second
STIFF-FLOP	STIFFness controllable Flexible and Learn-able manipulator for surgical OPerations
SVD	Singular Value Decomposition
VC	Variable Curvatures

Chapter 1

Introduction

The true essence of adaptability lies in its ability to blend the strength of resilience with the art of embracing new challenges and opportunities. In the realm of robotics, this principle resonates profoundly, shaping the trajectory of innovation. Adaptability lies at the heart of robotics research as a driving force propelling the creation of robots capable of thriving in unpredictable and dynamic environments.

One of the best examples within the field of soft robotics research is the soft pneumatic actuators (SPAs), a remarkable category of actuators that exhibit intrinsic compliance and adaptability. These soft actuators, constructed from pliable materials and employing compliant actuation methodologies, find their purpose in various domains, such as mobile robotics, assistive wearable technologies, and rehabilitative devices [1]. SPAs possess the distinctive capability of operating within intricate environments, mitigating concerns related to detrimental impacts on their surroundings or themselves [2].

SPAs inherently embody compliance and exhibit a high degree of manufacturability, thus heralding a promising future for the development of novel robotic systems, characterised by inherent safety, adaptability, and customisability [1]. They can be configured to demonstrate a diverse range of performance attributes, including extension/contraction, bending, twisting/helical motion, bidirectional / omnidirectional motion, and more [3]. These actuators, however, fall behind their rigid actuators counterparts in the ability to sense their state in planar or 3D spaces.

Soft actuators can have infinite degrees of freedom (DOF) and can be deformed by both internal and external loads. Predicting the response of these soft actuators solely through modelling presents a formidable challenge, given the intricate nature of their behaviour characterised by non-linearity and viscoelastic effects. Therefore, it is essential to emphasise the significance of improving their manipulation and sensing capabilities, as they directly correlate with the achievement of higher state estimation accuracy and efficient control systems. [4].

1.1 SPAs at a Glance

The idea of SPAs was first introduced by McKibben in the 1950s when he suggested an artificial muscle made of a rubber tube surrounded by a braided mesh [2]. By

pressurising the tube, a linear contraction movement was achieved, and the mesh maintained the mechanical stability. In many robotic applications today, including wearable exoskeletons, rehabilitation equipment, and prosthetics, this type of actuator is still commonly utilised [5]. A good example of SPAs is the **OctArm** which was developed by researchers at Pennsylvania State University [6]. It is a type of continuum robots that is inspired by the structure and movement of invertebrate organisms such as octopus arms and elephant trunks.

OctArm is constructed from a series of interconnected chambers embedded in an elastomeric matrix. **OctArm** has the ability to bend, twist and extend in different directions. It has a whole arm manipulation capability as it can utilise the entire length of its arm to grasp objects of different shapes and sizes. Such features in the design and control enabled the **OctArm** to be used in various robotic applications, such as prosthetics, rehabilitation devices, wearable exoskeletons, and search-and-rescue operations.

The pneumatic network, known as **PneuNet**, is another type of SPAs that was originally developed by the Whitesides Research Group [7] at Harvard University. The **PneuNet** consists of several chambers contained within an elastomeric matrix. When under pressure, these channels expand and move. By altering the shape of the embedded chambers and the composition of their walls, the nature of this motion can be altered. When a **PneuNet** actuator is pressurised, expansion takes place where it is least rigid and most compliant. **PneuNets** have found applications in various fields, including the development of soft grippers, and quadruped crawlers [8].

1.1.1 SPAs as Embodied Robots

SPAs exhibit characteristics associated with embodied robotics, as their design draws inspiration from the structural and motion mechanics of biological organisms [9]. However, beyond bio-inspiration, the concept of embodied intelligence in soft robotics emphasizes the interplay between an actuator’s physical properties, sensory feedback, and control mechanisms [10, 11].

Embodied intelligence is a well-recognised approach in soft robotics in which a certain intelligence requires a physical body to perform certain tasks. In the context of SPAs, the term *sensorimotor* encompasses the integration of sensory data and motor actions, spanning across various aspects including the SPA’s design along with its kinematics, dynamics, sensing, and control capabilities [11–13].

The SPA’s **design** determines its structure and the materials it is made of, which is pivotal in shaping the actuator flexibility, durability, and motion range. **kinematics** involves analysing the motion of the SPA due to air pressure variations, without accounting for the forces driving the movement.

Dynamics, conversely, involves the study of motion and the forces that cause it, such as how changes in air pressure trigger the SPA’s movement over time. **Sensing** is the capability of the actuator to monitor any changes through the integration of sensors capable of detecting various parameters. This could be changes in the actuator’s own state (like deformation or pressure changes) or changes in its environment.

Control involves the systems used to control the SPA’s movements, which could

involve complex algorithms that take sensor data as input and output instructions to change the air pressure in the actuator. Enhancing sensorimotor functionality in SPAs would involve advancements and innovations in all these areas, making it a truly interdisciplinary field that combines elements of many different areas of study.

Within a typical sensorimotor scheme of an SPA, a sensory system perceives the surrounding environment, controllers process the incoming information and formulate motor action plans, and then a mechanical system executes these plans within the physical environment. *Embodied intelligence* can be understood as the direct mechanical response of the robot's physical body to the environmental stimuli, or mechanical feedback, with no involvement of controllers or processing systems as shown in Figure 1.1.

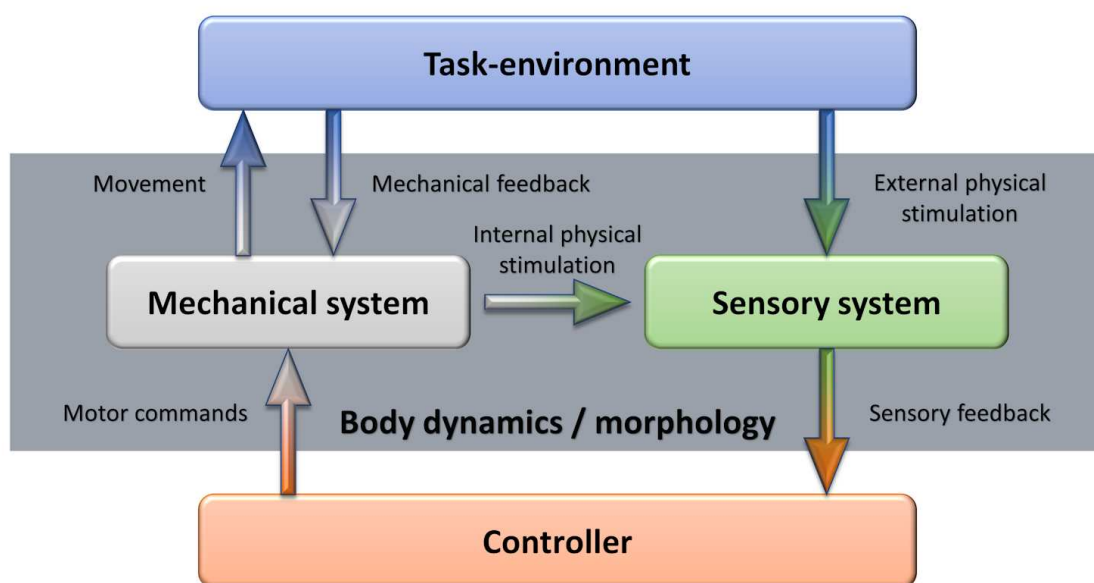


Figure 1.1: The main components of a typical sensorimotor scheme: a mechanical system, a sensory system, and a controller. Adapted from [10] with permission from AAAS.

While SPAs are commonly referred to as actuators in the literature, it is important to acknowledge that they do not fully meet the conventional definition of an actuator. Typically, actuators are characterized by the presence of a transducer, which converts energy from one domain, such as electrical energy, into another, such as mechanical energy. In the case of SPAs, the primary functional component responsible for energy transformation is the external pneumatic system (such as a pump) that supplies compressed air. The SPA itself, as a soft, deformable structure, primarily serves to respond to the applied pressure, creating motion or force. Therefore, while SPAs perform actuation, the transduction process often lies outside the actuator itself, and the performance evaluation of SPAs typically focuses on their response to pneumatic input rather than the energy conversion process. The focus of this thesis, however, is on the SPA itself and its sensorimotor capabilities, without delving into the specifics of the external pumping system.

1.1.2 Common Soft Actuator Models

In general, SPAs can be designed to produce planar, axial, and three-dimensional motions, either individually or in combination [2, 5, 14]. Planar bending actuators are generally made of a single chamber that inflates in a single direction, leading the actuator to bend in a plane. These kinds of actuators are useful for building grippers, crawlers, and other planar motion devices. Axial actuators, on the other hand, are typically designed to extend or contract along their length when pressurized, making them suitable for applications requiring linear motion, such as soft robotic pistons, artificial muscles, and adaptive structures [15].

Conversely, three-dimensional bending actuators are made up of several chambers designed to inflate in various directions, enabling the actuator to bend and twist in three dimensions. These kinds of actuators are important for building tentacles, soft robotic arms, and other devices that require complex motion. However, complex three-dimensional deformations can also be achieved in single-chamber actuators through careful optimisation of geometry and material properties [5].

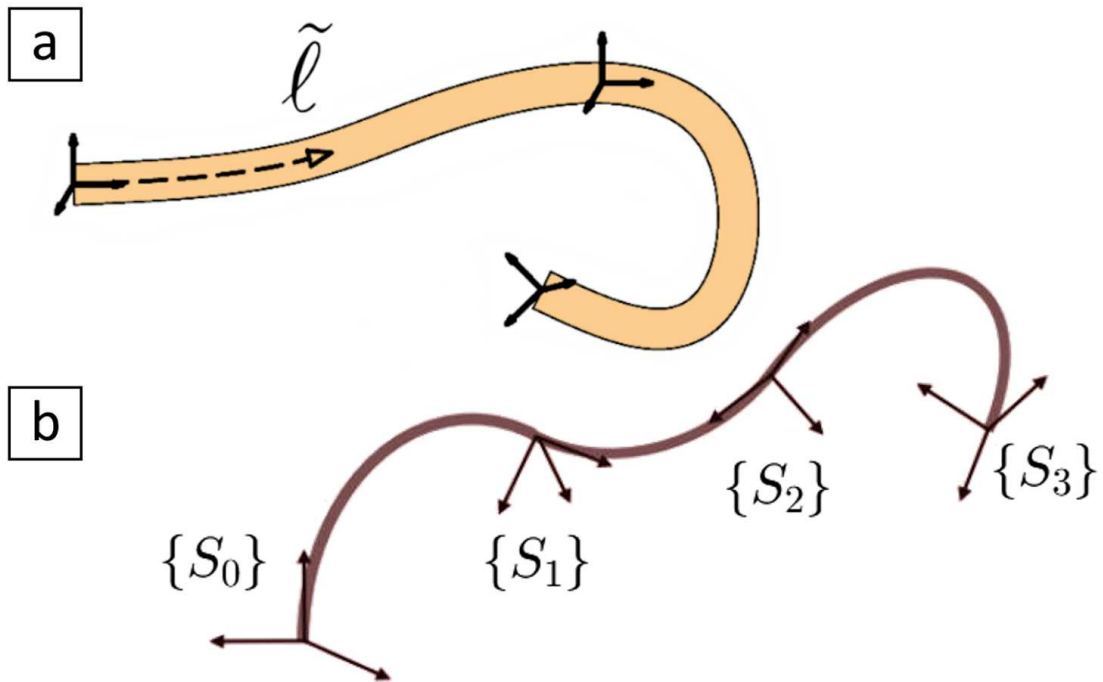


Figure 1.2: Soft actuator common models. (a) represents a VC actuator, and (b) illustrates a multi-segment PCC actuator.

Two prominent SPA modelling approaches were established: variable curvatures (VC) and piecewise constant curvatures (PCC), Figure 1.2(a,b). For VC, the actuator can be considered as a slender rod and modelled via the Cosserat rod model [16]. While for PCC, the actuator can be modelled via several methods [17]. Moreover, a hybrid approach, integrating both VC and PCC, known as piecewise constant strain (PCS), gained recent recognition in SPA modelling [18, 19]. The PCS model extends the constant-curvature assumption to more general rod kinematics and provides a simpler dynamic modelling than the VC model.

The PCC model has several advantages when implementing it in tasks such as the de-

sign, sensing, and control of soft robots. Compared to both the [VC](#) and [PCS](#) models, the [PCC](#) model offers simplicity, generality, and applicability to the aforementioned tasks [20]. But there are also several drawbacks to the [PCC](#) model, particularly in its three-dimensional formulation, where problems with discontinuities and singularities arise, especially with the straight configuration of the actuator [21]. Further details are provided in Chapter 2, Section 2.1.2.

However, modelling [SPAs](#) is a challenging task considering their high non-linearity, material heterogeneity, and contact interactions [22]. Therefore, it is important to assess the accuracy and robustness of these mathematical models against simulated environments in conjunction with experimental data [23]. Researchers commonly employ various verification and validation techniques based on their effectiveness and sufficiency. Verifying and validating the mathematical models is a fundamental requirement in [SPA](#) research, as these models provide quantitative insights into [SPA](#) responses and enable efficient and systematic design, simulation, and control for various applications [22, 24, 25].

1.1.3 Verification and Validation Techniques

Verification via simulation is the process of checking whether a mathematical model of an [SPA](#) behaves as expected in a simulated environment and whether it meets given requirements [22]. This process helps identify and correct inconsistencies in mathematical models and evaluate their performance and robustness under different conditions and scenarios. Furthermore, verification via simulation provides insights into model limitations and assumptions, guiding improvements to model reliability, particularly when using well-established tools that are validated for simulating actuator behaviour.

In the realm of [SPAs](#), software packages like the Robot Operating System (ROS) and Gazebo [26] offer versatile environments for simulating actuator behaviour. Additionally, Finite Element Analysis ([FEA](#)) is commonly used to simulate deformation and stress distribution in [SPA](#) structures [27]. Researchers employ these tools to conduct simulations, comparing model predictions with simulated outcomes under controlled conditions. This process helps identify potential discrepancies or errors inherent in the mathematical framework.

Validation via experimental results represents a further cornerstone in the pursuit of model accuracy, especially in the context of [SPAs](#) [28]. Researchers design experiments tailored to assess model predictions, ensuring their compatibility with real-world scenarios. Moreover, previously published experimental data can be leveraged for validation purposes [24]. By comparing model predictions to empirical data obtained through experiments, researchers determine the model's effectiveness in real applications.

Moreover, the iterative process of Experimental Confirmation and Iteration plays a vital role in soft robotics research, involving continuous refinement and enhancement of mathematical models. This iterative approach often requires physical experimentation on [SPA](#) prototypes, serving as a feedback mechanism. Researchers use these experiments to validate model predictions, guide adjustments, and facilitate improvements. Through this iterative process, model accuracy and reliability progressively

improve, better representing the complex behaviours of SPA systems [29].

These four techniques, collectively encompassed within the broader framework of verification, validation, and iterative confirmation, constitute indispensable components of SPA research. They collectively contribute to ensuring that mathematical models faithfully represent the intricate behaviours of SPA systems, thus fostering advancements in this dynamic and evolving field.

1.2 Justifications and Research Questions

This study embarks on an extensive exploration of several critical challenges in the domain of PCC-based SPAs and soft sensors, with the overarching goal of enhancing the sensorimotor functionality of these actuators. In this section, three interconnected research problems are investigated, each addressing fundamental limitations in the field of soft robotics.

1.2.1 Inverse Kinematics of SPAs

The manipulation and control of SPAs are central challenges in soft robotics, where adaptability and compliance are paramount. However, achieving an optimum control in these inherently compliant systems remains a significant challenge due to the high dimensionality, redundancy, and non-linearity of these systems [2]. This research problem focuses exclusively on the inverse kinematics (IK) of these actuators by investigating some aspects such as convergence rates, singularity avoidance, and physical limits. Then, attempt to answer the following questions:

- How to optimise the convergence rate of IK solutions to ensure efficient responsiveness and control?
- What strategies can be developed to navigate SPA singularities effectively?
- What insights can be gleaned regarding the physical limits of SPA motion, and how can the IK solvers be optimised to achieve effective solutions within these limits?

By addressing these enquiries, I aim to enhance the reliability of SPA control, pushing the boundaries of what is achievable in the realm of soft robotics.

1.2.2 Twist Measurement

Efficient manipulation and control of SPAs necessitate an elaborate understanding of complex deformations, including twist. Twist motion is a complex mode of deformation that can enable soft robots to perform tasks such as grasping, manipulation, locomotion, and shape morphing [30–34]. However, twisting motion introduces challenges in terms of modelling, sensing, and control of SPAs. Therefore, measuring the twist is essential for understanding the SPAs behaviour, improving their performance, and enabling further applications. This research problem focuses on modelling the behaviour of straight and spiral strain sensors attached to the SPA surface or embedded inside it, with an emphasis on measuring twist. Then, attempt to answer the following questions:

- What mathematical concepts can be used to develop a simplified and accessible mathematical model for twist measurement in SPAs? A model that may serve as a valuable tool for researchers in the field of soft robotics.
- What are the relative sensitivities of spiral and straight sensors in capturing twist deformations in SPAs, and how can this information guide sensor configuration choices for specific applications?

By addressing these enquiries, I aim to provide the soft robotics community with a valuable resource for modelling and controlling twist in SPAs, furthering the adaptability and enhancing the capabilities of soft robotic systems.

The following terms are used throughout the text of this thesis: 'segment', 'muscle', and 'sector', which refer to a curved SPA, a single pneumatic tube or muscle within the SPA, and one-third of the SPA body in the FEA analysis, respectively.

1.2.3 EGaIn-Based Soft Sensors Modelling

Closed-loop controllers can benefit from proprioceptive sensing as it enables feedback mechanisms that improve the accuracy of controlled movements. This is especially important for activities requiring fine movements, such as manipulation, locomotion, or balance [35]. For PCC actuators, several solutions were introduced using different technologies such as optical [36, 37], inductive [38, 39], Carbon nanotube-based (CNT) [40], Eutectic Gallium-Indium (EGaIn) [41], or capacitive [42] sensors. All these sensors measure the local change in length or curvature based on the sensor position. EGaIn-based soft strain sensors offer promising advantages but are hampered by inaccuracies in geometry-based models and the omission of calibration parameters. This research problem seeks to address these limitations, drawing on insights from previous studies of EGaIn-based sensors, and attempt to answer the following questions:

- How can a new mathematical model be developed to accurately estimate the strain or elongation of EGaIn-based soft strain resistive sensors, accounting for factors such as strain rate and hysteresis, and incorporating calibration parameters?
- What are the implications of integrating this enhanced modelling approach into the design and application of EGaIn-based soft sensors for soft robotic systems, particularly in terms of improving measurement accuracy and adaptability?

By addressing these enquiries, I aim to contribute to the refinement of EGaIn-based soft sensors, enhancing their accuracy and versatility for soft robotics, wearable technology, and related fields.

1.3 Contributions

This study explores three interrelated research problems that are associated with the sensorimotor functionality of SPAs. Within the scope of this study, I address critical gaps in the existing literature, contributing novel insights by tackling challenges in both the mechanical and sensory systems of the PCC-based SPAs as follow:

Inverse Kinematics: Within the SPA mechanical system, I focus on suggesting solutions for the limitations of existing IK solutions by offering the following contributions:

- Investigate the stability and convergence rate of the current Jacobian-based IK solutions and provide insights into optimising computational efficiency. I aim to accomplish that via testing multiple damping profiles as well as developing effective strategies for navigating singularity condition(s) to enhance the manipulation reliability and stability.
- Formulate an alternative IK solution based on convex optimisation and derive the exact Hessian of the SPA. Then, compare with the current Jacobian-based solutions in terms of stability, singularities, and convergence rate.
- Explore the physical limits of SPA motion for an effective manipulation within these boundaries in order to broaden the applicability of SPAs in real-world scenarios.

Twist Measurement: Within the SPA sensory system, I aim to enhance actuator state sensing capabilities. While the PCC model is capable of representing general deformations, it falls short in representing twist and shear movements as stated earlier in Section 1.1.2. This study excludes shear, caused by gravity or external forces, since it will invalidate the PCC assumption. Even though the PCC model cannot represent twist, the SPA curvature can maintain a constant value when subjected to a pure twist while bending and rotating. For this section of the study, I offer the following contributions:

- Employ the mathematical concepts of *geodesics* and *covering spaces* to derive a mathematical model capable of measuring twist movement in both straight and spiral superficial strain sensors in terms of the SPA state in 3D space. This model will serve as a valuable resource for researchers in the soft robotics field, facilitating efficient manipulation and control of SPAs.
- Build a simplified FEA model in Abaqus using an *orthotropic* layer as a surface reinforcement to verify the twist measurement mathematical model. Such a reinforcement technique can reduce the FEA simulation time significantly since it eliminates the need to use *contact interactions* in Abaqus.
- Investigate the sensitivity of the straight and spiral superficial sensors to various bending, extension, and twist deformations to get further information. These observations will offer guidance for choosing the best sensor arrangement based on particular application needs.

EGaIn-Based Soft Sensors Modelling: This study addresses the accuracy limitations associated with the uniaxial EGaIn-based strain sensors in estimating the strain as a result of the measured resistance. Overcoming such limitations leads to better proprioceptive sensing for SPA deformation and state estimation. I contribute the following to this section of the study:

- Introduce a novel mathematical model that accurately estimates the

strain or elongation as a function of resistance and determines the estimation accuracy. This model incorporates calibration parameters, which significantly improve measurement accuracy compared to geometry-based models.

- Study the effects of strain rate and hysteresis in order to improve measurement accuracy and adaptability. This enhanced modelling approach broadens the utilisation of these sensors in soft robotics, wearable technology, and related fields.
- Offer a comprehensive testing procedure that acts as a practical guide for researchers seeking to replicate or expand upon the experimental methodologies outlined in this study. By providing detailed instructions and insights into overcoming technical obstacles, this work aims to enhance the reproducibility and comparability of experimental findings within the soft robotics community.

Collectively, these interrelated research problems underpin this study’s contributions of advancing our comprehension of soft robotic systems and providing innovative solutions to critical challenges in the field. Each problem addresses a distinct aspect of sensorimotor functionality, contributing uniquely to the adaptability, sensing, and control of SPAs.

The first research problem, concerning IK, establishes a computational framework for motion planning in SPAs by addressing their high dimensionality, redundancy, and non-linearity. By leveraging convex optimization techniques, this work enhances the efficiency and reliability of IK solutions, enabling SPAs to perform tasks with improved responsiveness while maintaining their inherent compliance. This contribution directly supports enhanced motion control for SPAs in dynamic environments.

The second research problem, centered on twist measurement, provides a sensing framework for capturing complex deformation modes essential for advanced functionality. By investigating the relative sensitivities of straight and spiral strain sensors, this work offers a valuable tool for researchers seeking to model or measure twisting in SPAs. While not directly applied to control in this study, the twist measurement framework enhances the broader understanding of SPA deformations and lays the groundwork for potential future applications in adaptive manipulation and locomotion.

The third research problem, focused on EGaIn-based soft sensors, develops a more accurate mathematical framework for interpreting strain measurements. These sensors offer a means of proprioceptive feedback, which is crucial for tasks requiring closed-loop control or self-monitoring. By incorporating calibration parameters and accounting for strain rate and hysteresis, this contribution improves the reliability and adaptability of SPA systems, particularly in tasks requiring fine movements and environmental interaction.

In combination, these contributions advance the sensorimotor functionality of SPAs by addressing significant challenges in motion planning, deformation sensing, and proprioception. Although each research problem is explored independently, their combined insights contribute to the broader domain of soft robotics, fostering the development of more capable and adaptable machines suited to unstructured and

dynamic environments.

1.4 Thesis Structure

This thesis is organised as follows:

- **Chapter 1: Introduction**

In the introductory chapter, an overarching view of the field of soft robotics is presented, along with the specific research problem under investigation. The rationale behind this study is outlined, and the central research aims and questions are introduced, setting the stage for the entire thesis. Additionally, this chapter provides a concise overview of the thesis structure and organisation.

- **Chapter 2: Literature Review**

The second chapter undertakes a thorough review of the pertinent literature within the domains of SPAs, soft sensors, and the broader field of soft robotics. It offers an overview of the historical background, the current state of the art, and the existing research gaps. This thorough review forms the foundation upon which the subsequent research problems are built.

- **Chapter 3: Inverse Kinematics of Soft Pneumatic Actuators**

Chapter Three delves into the Jacobian-based IK of SPAs. It places particular emphasis on essential aspects, including the stability and convergence rates of IK solutions, strategies for singularity avoidance, and the characterisation of physical limits governing SPA motion. The chapter also delves into formulating an alternative IK solution based on convex optimisation, then provide insights gained from comparing it with the Jacobian-based solutions.

- **Chapter 4: Spiral and Straight Superficial Sensors Modelling for Twist Measurement**

In Chapter Four, the focus narrows onto the modelling of sensors tailored for capturing twist deformations in SPAs, with an explicit examination of both spiral and straight superficial sensor configurations. The fundamental objective of this chapter is to develop a mathematical model for accurately measuring twist in SPAs.

In addition, a secondary task is introduced that involves building a simplified FEA model within Abaqus to verify the twist measurement mathematical model, ensuring its accuracy and reliability. The insights garnered in this chapter become instrumental in guiding sensor configuration choices for twist measurement within specific soft robotic applications. Following the FEA task, the mathematical model results are compared to previously published experimental results for validation, further solidifying the reliability of the developed mathematical model.

- **Chapter 5: EGaIn-Based Soft Sensors Modelling**

Chapter Five addresses modelling of uniaxial [EGaIn](#)-based soft strain resistive sensors. It proposes a novel mathematical model that not only accurately estimates strain but also accounts for complex factors, including strain rate, hysteresis, and calibration parameters. This advanced modelling approach is expected to significantly enhance measurement accuracy for the purpose of advancing their utility in the realm of soft robotics.

- **Chapter 6: Conclusions and Future Works**

The concluding chapter provides a synthesis of the key findings and contributions across the research chapters. It underscores the broader implications of this work within the context of the evolving field of soft robotics. Moreover, this chapter sets the stage for future research by suggesting potential avenues for further exploration and ongoing enquiries.

Chapter 2

Literature Review

Building upon the preliminary insights shared in the introduction chapter, this literature review delves into the sensorimotor scheme of the PCC-based SPAs, focusing on three integral aspects. The IK of the SPAs, the twist measurement, and the modelling of EGaIn-based soft strain sensors all occupy central positions in this exploration. These areas collectively highlight the profound implications of sensorimotor schemes in the design, operation, and utilisation of soft robotic systems. Sections to follow provide a detailed exploration of each aspect, dissecting the existing literature to highlight critical gaps while unveiling avenues for in-depth analysis in the subsequent chapters. Furthermore, the forthcoming chapters will highlight the specific contributions of this research within the context of soft robotics.

2.1 Inverse Kinematics

A significant challenge in designing SPA systems is solving the IK problem, which is determining the suitable input pressures to achieve a desired output pose, position and orientation, or trajectory of the SPA. The IK problem remains a challenge to solve considering the large number of DOFs involved as well as the soft materials non-linear behaviour. As a result, development of effective IK models for SPAs is still one of the current research topics in the field of soft robotics.

2.1.1 Degrees of Freedom Concept

Fundamentally, IK is the process of finding the optimum configuration of a given kinematic chain, such as an SPA, to set the end point of the chain at a desired pose. This process helps in determining the required change in the kinematic chain configuration in order to move from an initial pose to a desired pose across a certain path or trajectory [43]. The DOF concept plays an important role in IK. It refers to two different quantities based on the context. For a kinematic chain, DOF refers to the number of movements or independent parameters that define the configuration of the chain [44]. On the other hand, DOF refers to the freedom of movement of an object in the planar or three-dimensional (3D) space [45].

Throughout this thesis, DOF_k is used when referring to kinematic chain configuration, and DOF_m when referring to the freedom of movement in the planar or 3D

space. For a planar space, there are 3 DOF_m which are two translational along x and y axes, and one rotational about the object's centroid, Figure 2.1(a). Similarly, for a 3D space, there are 6 DOF_m which are three translational along three perpendicular axes (x , y , and z), and three rotational about these axes, Figure 2.1(b). The actuator is kinematically redundant when its DOF_k is larger than DOF_m [43]. Furthermore, two distinct types of actuators can be seen in literature when DOF_k is very large compared to DOF_m . The first type is the hyper-redundant actuators, which consists of many short rigid links such as snake-arms [46]. The second type is the continuum actuators, a special category of the hyper-redundant actuators [47], which have elastic structures with continuously bending curvatures and infinite DOF_m [17].

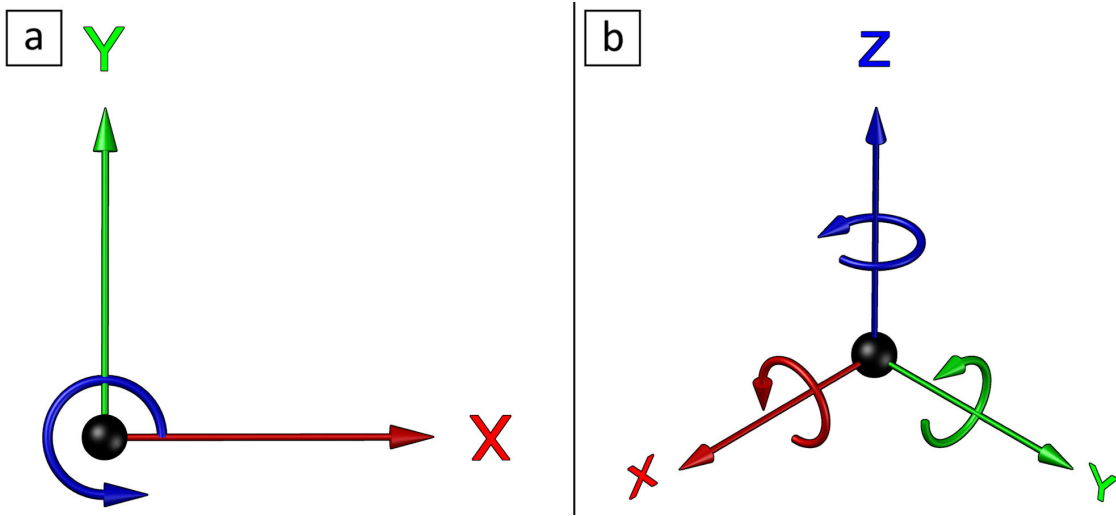


Figure 2.1: Degrees of freedom DOF_m for an object in (a) planar and (b) 3D spaces.

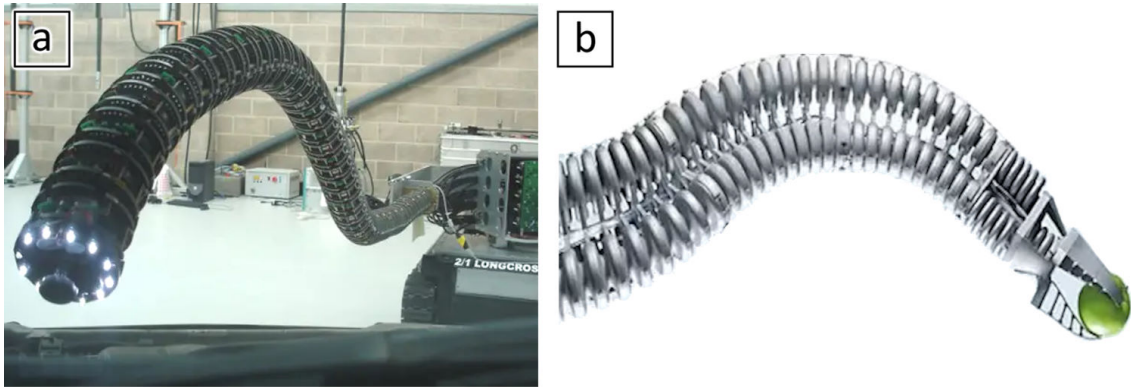


Figure 2.2: Examples of hyper-redundant actuators such as (a) the snake-arm by OC Robotics [48], and continuum actuators such as (b) the Bionic Handling Assistant by FESTO group [49].

Hyper-redundant actuators resemble snakes, elephant trunks, or tentacles in terms of appearance and functionality. These actuators can be built to have better robustness with respect to mechanical failure than actuators with a low degree of redundancy and are ideally suited for operation in severely confined areas because of their highly articulated structures [50]. A good example is the snake-arm by OC Robotics [48], Figure 2.2(a).

Continuum actuators, on the other hand, are highly flexible and can bend and twist in complex patterns. They are inspired by the movement of natural organisms such as snakes and tentacles as well. Continuum actuators consist of several flexible segments that can be actuated and move independently of each other. Several types of continuum actuators were introduced, including cable-driven, pneumatic, and hydraulic, in which they are utilised in several types of applications, such as industrial automation, search and rescue operations, and medical robotics [51]. One example within the field of the industrial automation is the Bionic Handling Assistant by FESTO group [49].

2.1.2 Soft Actuators Modelling Techniques.

Continuum actuators such as SPAs are known for their flexibility, adaptability to the environment, and the ability to undergo large deformation. They can be designed to mimic natural and biological organisms as well as provide safe interaction with people [52]. In general, two prominent SPA modelling approaches were established in the literature which are VC and PCC as discussed earlier in Section 1.1.2.

The soft actuator basic movements can be categorised as bending and rotation, twist, shear, and extension [16] as shown in Figure 2.3(c). The twist is a mechanical rotation about a certain point or axis caused by a torque acting on that point or axis^a. The PCC model can represent extension, bending, and rotation movements only. Both VC and PCS can represent twist and shear as well. The actuator can undergo one or more of these movements, depending on its design and actuation mechanism.

The VC approach, Figure 2.3(a), encompasses a diverse range of modelling techniques. Researchers were able to employ several of them such as Cosserat rod theory [16], Euler curves [53], Pythagorean hodograph curves [54], cyclic coordinate decent (CCD) [46], and absolute nodal coordinate formulation (ANCF) [55]. These techniques allow for a more continuous representation of the SPA curvature. While they offer high fidelity in capturing the complexity of soft actuator behaviour, they may also demand increased computational resources.

Alternatively, the PCC model, Figure 2.3(b), simplifies the modelling process by dividing the actuator into one or more constant curvature segments. Several techniques were employed for modelling SPAs under the PCC model, including mode shape function [56–59], Frenet-Serret frames [17], Denavit-Hartenberg (DH) parameters [18, 19], integral representation [60], and exponential coordinates based on Lie group theory [61, 62]. The PCC model tends to provide a more straightforward kinematic model, particularly when external forces can be considered negligible [55]. It is important to note that PCC modelling techniques differ not only in their level of accuracy but also in their numerical stability [63].

While the VC model excels at capturing fine-grained curvature behaviour, especially when it involves shear and twist movements, the PCC model offers a more straightforward and computationally efficient representation, especially when external forces can be considered negligible. Understanding the trade-offs between these modelling

^aHowever, some textbooks may define the term "twist", denoted by $\mathcal{V}$, as the time derivative of the actuator end point position and orientation [43]

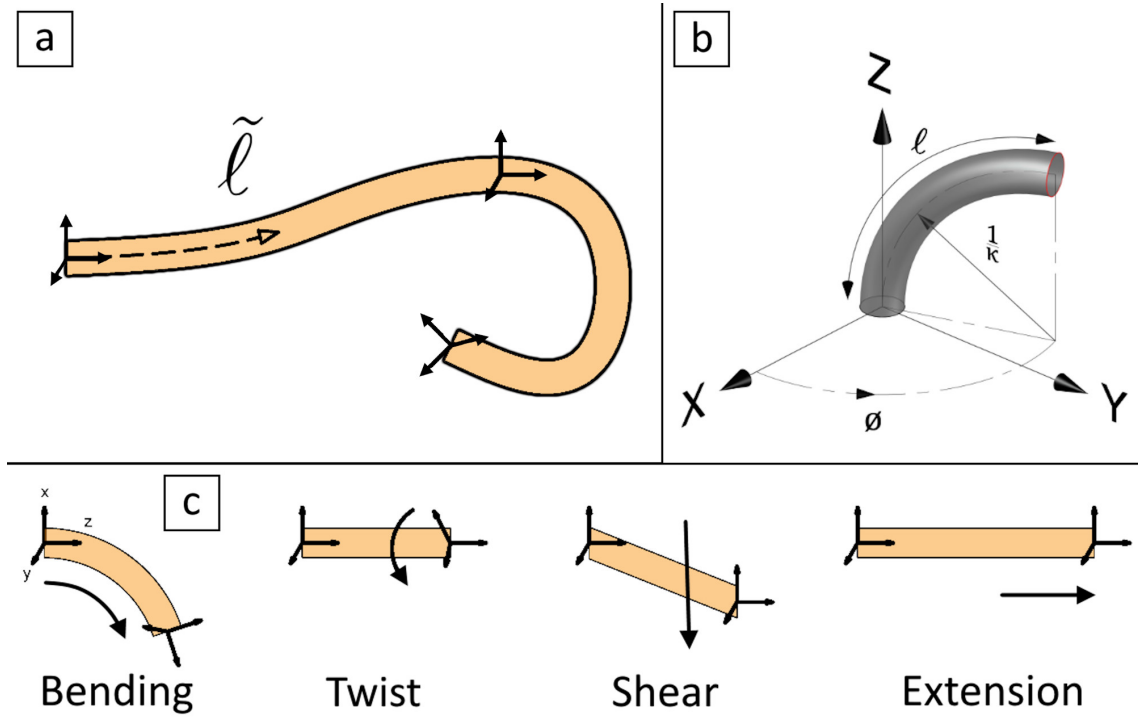


Figure 2.3: Soft actuator modelling where (a) represents a **VC** actuator, (b) represents a single segment in a **PCC** actuator, and (c) are the soft actuator basic movements.

approaches is essential for addressing the unique challenges presented by solving the **IK** problems in **SPAs**.

In addition, it is essential to highlight the Piecewise Constant Strain (**PCS**) model as a distinctive hybrid model that combines elements of the **VC** and **PCC** approaches. While the primary focus of this literature review revolves around the **PCC** modelling approach, the **PCS** model warrants attention due to its innovative synthesis of modelling techniques. **PCS** discretises continuous Cosserat rods, accounting for shear and twist deformations induced by out-of-plane external loads [20, 64, 65]. Various modelling techniques are employed within the **PCS** model, such as screw theory [66], Lagrangian density [65], Galerkin-Ritz method [64], and Lyapunov function alongside torsional Jacobian [20].

Within the **PCS** framework, deformations are approximated as piecewise constant along the actuator length. Each segment of the actuator is discretised, and strain is considered constant within these segments. Such tactic leads to a finite set of strain vectors that describes the soft actuator and performs the same role as the joint vectors in rigid robots. This tactic is particularly valuable in complex soft robotic applications where both precision and computational efficiency are paramount. It underscores the adaptability of the **PCS** model in various scenarios, making it a noteworthy topic within the broader landscape of continuum actuators modelling and control [20, 65].

While both **VC** and **PCS** approaches demonstrate their utility in certain contexts, our primary emphasis in this literature review remains on exploring the **PCC** modelling techniques as they enable the application of rigid robotics kinematic models

and control methods to continuum designs [67]. In addition, the **PCC** model is the most widely used due to its simplicity and rich literature, has a shorter computation time compared to the **VC** and **PCS** models, and requires less complicated control systems [20]. The **PCC** model, however, has limitations regarding external loading such as gravity, as well as other issues such as torsional strain, non-linear material behaviour, and tendon friction if used for actuation [67]. Researchers who prefer to implement the **PCC** model in their soft actuator designs are responsible for justifying its validity and accuracy.

The aforementioned models (**PCC**, **PCS**, and **VC**) are part of a group known as beam theory-based kinematics [68]. Other types can be seen in the literature, such as approximate piecewise constant curvature (APCC) [69], piecewise polynomial curvature (PPC) [70], finite element method (FEM)-based kinematics [71], machine learning-based kinematics [72], among several others [68], and they are outside the scope of this thesis.

2.1.3 PCC Model Representation

The **PCC** model is commonly represented via arc parameterisation [17] which is based on three parameters: arc curvature, arc length, and rotation angle about the vertical axis, Figure 2.3(b). These parameters are explained later in Chapter 3, Section 3.1.2. In theory, each segment in a **PCC**-based actuator possesses three DOF_k , distinguishing it from **VC** and **PCS** models, which exhibit an infinite number of DOF_k . However, solving the kinematics of **VC** and **PCS** models numerically necessitate the discretisation of the actuators into finite segments and finite DOF_k accordingly [65, 73].

Arc parameterisation stands as the most common representation within the literature due to its simplicity as well as facilitating a straightforward modelling of superficial or embedded strain sensors in terms of the actuator state, which are central for proprioceptive sensing applications [35, 37, 74]. Nonetheless, arc parameterisation does have its limitations. Notably, it suffers from a singularity at the straight position (i.e., zero curvature) and discontinuities in the rotation angle [21].

The singularity occur when the curvature of the actuator becomes zero, Figure 2.4(a), resulting in a loss of degrees of freedom or the uniqueness of the solution. This can cause problems for **IK** and control methods that rely on the invertibility and uniqueness of the model. The discontinuities cause two types of shortcomings to the actuator configuration. The first shortcoming is the interpretation of the negative and positive values of the rotation and bending angles in which both cases produce the same physical posture, Figure 2.4(b).

The second shortcoming occur when the rotation angle of the actuator changes suddenly, Figure 2.4(c), resulting in a discontinuous or unrealistic change in the arc parametrisation variables. This can cause problems for numerical methods and optimisation algorithms that rely on the smoothness and continuity of the model.

These issues are not inherent to the **PCC** assumption itself but rather to the commonly used arc parametrisation. To address these challenges, various alternative representations were proposed, including dual quaternion mapping [56], augmented rigid body models [75], closed form solutions [76], opposite virtual arcs [21], rota-

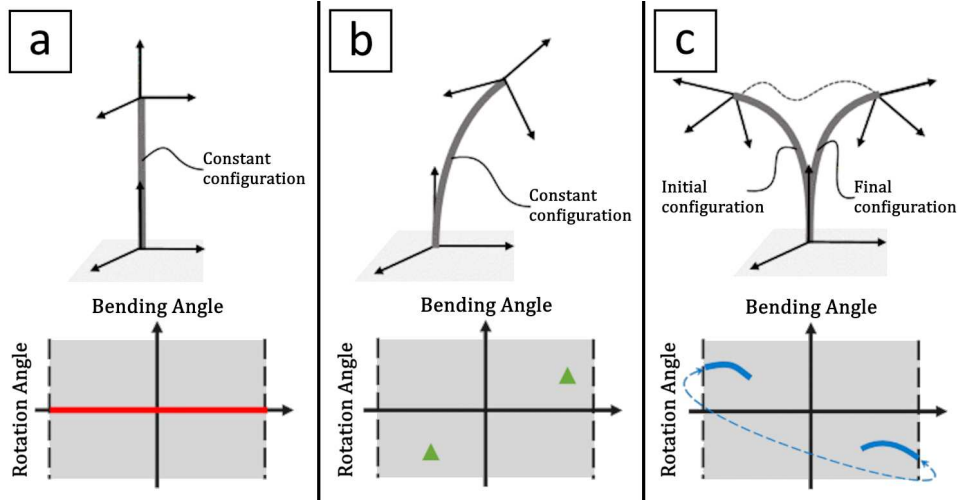


Figure 2.4: The PCC model limitations, adapted from [21] and reproduced under the terms of CC BY 4.0 license. The top row illustrates the actuator physical configurations while the bottom row illustrates a manifold contains the actuator rotation and bending angles at the horizontal and vertical axes respectively. The PCC model shortcomings are (a) the singularity at straight position, (b) same physical posture at positive and negative angles, and (c) discontinuity in the path of the actuator end point.

tion axis mapping [77], and polynomial approximation [78]. These alternatives offer solutions to mitigate the aforementioned singularity and discontinuity, but they often introduce greater complexity, especially in the context of IK and proprioceptive sensing applications.

To tackle this complexity, some studies turned to artificial intelligence methods for designing proprioceptive sensing solutions [28, 41, 79]. There are, however, additional disadvantages to these solutions. They increase the complexity of the mathematical models and demand large volumes of data for validation and training.

It's worth noting that, despite a thorough review of the relevant literature, a comprehensive exploration of the effects of arc parameterisation singularity and discontinuities on IK stability and convergence rate remains a gap in the current literature, highlighting the need for further investigation.

2.1.4 Inverse Kinematics Techniques

Several techniques for solving IK of continuum or soft actuators were proposed in the literature, which can be broadly classified into analytical, numerical, and data-driven techniques. Analytical techniques [76] provide closed-form solutions based on the geometric design of the soft actuator. These techniques are usually fast and precise, but they also rely on the actuator particular design [80]. Therefore, they cannot be generalised to other designs or configurations of the soft actuator. Moreover, analytical solutions may not exist or be unique for some cases, such as when the desired pose is out of the workspace or near singularities.

Numerical techniques use iterative methods to find approximate solutions to the IK problem. These methods can be generalised to more than one design of the soft actu-

ator if the FK model is available. A common class of iterative methods are known as Jacobian-based [81] solvers, which update the actuation configuration with each iteration by using the inverse or pseudo-inverse of the Jacobian matrix. Another class of iterative methods are heuristic solvers, such as forward and backward reaching inverse kinematics (FABRIK) [82] and cyclic coordinate descent (CCD) [46], which use geometric rules to adjust the actuation variables. However, numerical techniques have limitations too, such as convergence speed, stability, and robustness to noise and singularities.

Data-driven techniques use machine learning methods to model the relationship between the end-effector pose of the actuator and its configuration variables. This task can usually be done without the need for explicit mathematical models, opening the door to the potential of simulating the actuator kinematics and/or dynamics in situations where deriving the Jacobians and mathematical models proves challenging. These methods can be artificial intelligence-based [28, 83, 84], adaptive-based [85, 86], or regression-based [87].

Data-driven techniques can handle complex non-linear systems [72, 88]. However, their implementation is complicated. In addition, such methods usually require a substantial amount of data for training and validation. Alternatively, hybrid modeling approaches for inverse kinematics, integrating machine learning with physics-based models, effectively address the inaccuracies of modeling complex SPAs and the material fatigue of soft materials. These methods handle nonlinearity and variability, resulting in accurate predictions of the robot's configurations. By leveraging physics-based models, hybrid approaches reduce the amount of data required for training, making them more efficient. They also enable real-time adaptation to environmental changes, improving performance and robustness. Furthermore, they enhance the control and predictability of movements, reducing errors and increasing precision within the flexible nature of soft robotics [88–92].

In essence, data-driven techniques can be avoided when it is possible to derive the mathematical models and the Jacobians. This thesis focuses on Jacobian-based methods, a class of solvers valued for their simplicity and versatility, making them particularly suitable for various configurations of SPAs.

2.1.5 Jacobian-Based Methods

Jacobian-based methods are a class of numerical solvers that utilise the Jacobian matrix to iteratively solve the IK problem of SPAs. These methods are reasonably easy to implement and can be applied to various SPA designs and configurations, considering that the FK model is available or can be approximated. The fundamental concept behind Jacobian-based methods is their ability to approximate the complex, non-linear FK using Taylor series expansion. This expansion allows for a more tractable and invertible representation of the original system, simplifying the computation of IK [93].

Jacobian-based methods essentially leverage these Taylor series approximations to map the actuator inputs to the desired end-effector position and orientation. The degree of truncation in the Taylor series expansion distinguishes two main approaches: linear and quadratic approximations, each offering unique advantages and trade-offs.

Gradient descent-based algorithms are root-finding methods that can be used with linear approximation. This set of methods is basically trying to find the local or global minimum by starting from an initial seed and then taking repeated steps in the opposite direction of the gradient of the optimisation function at each iteration step [94]. The gradient descent technique, however, is highly sensitive to the initial seed location and can suffer from slow convergence.

Consequently, several variations of the gradient descent algorithm were introduced in the literature, including vanilla, stochastic, momentum, adaptive (such as Adam, AdaGrad, and RMSProp), natural, and conjugate gradient descent, among others. These variations are designed to adjust the convergence rate based on the history of gradients and the specific behaviour of the cost function. They can lead to faster convergence and reduce the sensitivity to the initial seed [95, 96].

Conversely, for both the linear and quadratic approximations, the Newton-Raphson non-linear root-finding method, or simply Newton's method, is commonly used as an iterative algorithm to refine and converge to an approximate solution, enabling the actuator to achieve the desired pose efficiently [97]. The linear approximation is obtained by truncating the Taylor series after the first derivative term. This approach produces a linear representation of the FK that can be solved using the inverse of the Jacobian matrix [43] or via other variations to be discussed below.

The linear approximation is fast and stable, but it may suffer from low accuracy and slow convergence, especially near singularities or out of the workspace [84]. Contrarily, the quadratic approximation is obtained by truncating the Taylor series after the second derivative term. This approach produces a quadratic representation of the FK which can typically be solved using the Jacobian and Hessian matrices [98]. More on that in Section 2.1.7 below.

The Jacobian inverse (JIV) method is a simple and straightforward approach that maps desired end-effector velocities to the rate of change in the actuation variables, i.e., the arc parameterisation variables. This method relies on the inverse of the Jacobian matrix to perform this task. However, the Jacobian matrix has to be square, which may not be true for redundant or under-actuated soft actuators, and non-singular to guarantee a unique solution.

Some improvements, including Jacobian pseudo-inverse, damping factors, null-space projection, or adaptive step size, were suggested to enhance the performance of the linear approximation [99, 100]. Each improvement introduces a different way, compared to JIV, to compute or manipulate the Jacobian matrix as follow:

The Jacobian transpose JTR method is based on the simplifying assumption of the principle of virtual work [101]. This method calculates the rate of change in the actuation variables via the use of the Jacobian matrix transpose instead of the inverse. This method can be effective when the Jacobian matrix is singular, but it may require many iterations to converge and may result in large motions or oscillations. Additionally, the transpose method does not support strict priorities among constrained dimensions and does not consider the relative contribution of actuation variables [100].

The **Jacobian pseudo-inverse JPI**, or *Moore-Penrose inverse* J^+ , method is a generalised inverse that can handle singular matrices. This method has the advantage of being defined for all the Jacobian matrices, including those that are non-square or not of full rank. Such an advantage enables this method to provide an **IK** solution for both under-actuated and over-actuated systems as well. Additionally, the **JPI** method calculates the rate of change in the actuation variables that minimises the norm of the rate of change. This process can help avoid sudden changes in the actuation velocity and improve the stability and smoothness of the motion [102].

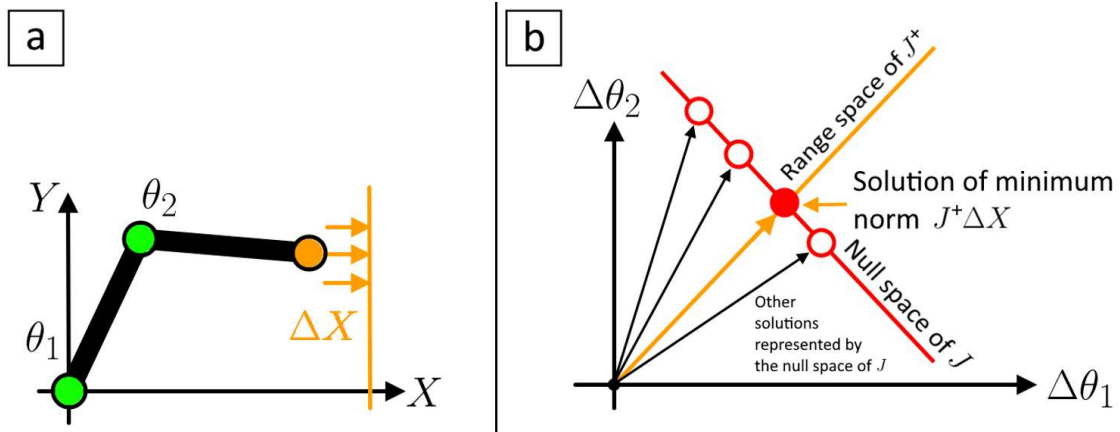


Figure 2.5: The simplest redundant case where (a) is a two DOF_k chain, θ_1 and θ_2 , and a one dimensional constraint along the x -axis for the end-effector. The null space projection (b) shows all the represented solutions as well as the solution at the minimum norm. Adapted from [102] under the permission of CAUL/GO8 Consortium.

The pseudo-inverse performs a projection of the desired motion onto the null space of the Jacobian matrix, Figure 2.5(b), to avoid singularities while achieving the desired end-effector pose [102, 103]. This is because every possible actuation configuration that does not affect the position and orientation of the end-effector is represented by the null space of the Jacobian matrix [98]. Nevertheless, this method has its own limitations as well. For example, the **JPI** method causes significant changes in the actuation variables when the actuator configuration is near a singularity, even though there may not be much movement towards the target, leading to oscillations and discontinuities in motion.

The **damped least squares (DLS)** method, or the *Levenberg-Marquardt* algorithm, stabilises the iteration process by adding a damping factor to the **JPI**. This helps to prevent many of the singularity-related issues that arise with the **JPI** method [104]. The damping factor can prevent large motions or oscillations near singularities or out of the workspace. The damping factor must be chosen carefully in order to achieve numerical stability. **DLS** method works better than the **JPI** and **JTR** methods [105]. However, its superior behaviour is subjected to the damping factor. While a large damping factor produces well-behaved solutions close to singularities, it also slows down convergence, makes tracking targets less accurate, and causes shaking and oscillation.

The singular value decomposition (SVD) method provides a powerful tool for analysing both the **JPI** and **DLS** methods. It allows for the analysis of the singular values of the Jacobian matrix, aiding in the identification of the most critical actuators and redundancy resolution. The **SVD** method is particularly advantageous when optimising performance and avoiding singularities. This method decomposes the Jacobian matrix into three matrices: U , S , and V , where U and V are orthogonal matrices and S is a diagonal matrix of singular values. The pseudo-inverse can be computed from these matrices by inverting the nonzero singular values and transposing U and V . This method can also detect and handle singularities by setting small singular values to zero. Nevertheless, the **SVD** method can be computationally demanding, especially when dealing with large kinematic chains.

The pseudo-inverse damped least squares (PDL) method is a modified version of the **DLS** method that makes use of the **SVD** under the **DLS** method [104]. **PDL** combines the **JPI** and **DLS** methods by adding a damping factor to the diagonal matrix of singular values before inverting them. This method offers enhanced stability over the **JPI** alone, but it still requires careful tuning of damping parameters. **PDL** method is often preferred for redundant systems over the standard **DLS** method due to its ability to handle redundancy resolution more effectively and efficiently [106].

The **PDL** method explicitly addresses redundancy resolution, which involves determining how to best use the additional **DOF** to optimise the task. This is accomplished by using the pseudo-inverse of the Jacobian matrix, which can be tuned to prioritise specific solutions, such as minimising energy consumption, avoiding obstacles, or optimising the end-effector pose [106, 107]. Standard **DLS** may not provide the same level of control and flexibility in redundancy resolution. However, The **PDL** method combines aspects of **SVD** and **DLS**, both of which can introduce computational overhead. The computational cost can be a concern in time sensitive applications, where rapid decision-making and control are crucial. Complex calculations might not be feasible within tight time constraints [108].

There are other Jacobian-based solutions with particular application specifics found in the literature [100]. While they may offer advantages in certain scenarios, their limitations often depend on the specific adaptations and variations used. The applicability of these methods for a particular application may be less clear than that of the more established techniques due to their lack of widespread documentation or validation. In summary, several Jacobian-based methods were introduced to be used for the linear approximation of the Taylor series. These methods have differences in regards to convergence rate, stability, redundancy, and singularity handling. Some of these methods add a damping factor to enhance stability, and the selection of this damping factor affects the behaviour of the iteration process.

2.1.6 Damping Factor Selection

The damping constant serves as a significant parameter that influences the stability and precision of the iteration process in **IK**. It plays a pivotal role in mitigating the occurrence of singularities or ill-conditioned matrices when employing the Jacobian matrix to determine joint velocities. The selection of a suitable value for the damping factor depends on the desired trade-off between stability vs convergence rate.

Various techniques are available in the literature for determining the optimal value of the damping factor.

A fixed damping factor represents the simplest technique, wherein a constant value is assigned to the damping factor, typically determined through trial and error. This damping technique is utilised in both [DLS](#) and [PDL](#) methods. Several studies were conducted to select the optimum constant value. Interested reader may refer to [\[99\]](#) for further details. While it is a straightforward technique, it may not yield optimal results for all configurations and target positions. Using a very small damping factor may lead to instability or oscillations, whereas using a very big value may compromise accuracy and convergence rates.

The adaptive damping technique adjusts the damping factor based on specific criteria, such as proximity to the target or the error norm. The idea is to increase the damping when the actuator configuration is close to a singularity or far from the target. Similarly, the damping factor is decreased when the actuator configuration is well-conditioned and near the target. This approach can enhance the performance and resilience of the [IK](#) solution but may necessitate additional computational resources and parameter tuning. A good example can be seen in [\[99\]](#), where a scaling factor is used in conjunction with the [JTR](#) method to control the step size of the iteration based on the error norm. Another example can be seen in [\[109, 110\]](#), where the damping factors were calculated as functions of the actuator manipulability.

The selective damping technique was introduced as a replacement for the fixed damping in solving the [IK](#) problem of redundant actuators [\[105\]](#). This technique applies distinct damping factors to individual singular vectors of the Jacobian matrix, depending on their corresponding singular values. The aim is to increase damping for singular vectors associated with small singular values, which represent directions of limited mobility or sensitivity. This way, selective damping can reduce the oscillation of the actuator when the target position is unreachable or near the boundary of the workspace, and also improve the numerical stability of the solution. However, it has the highest calculation cost among other techniques since it involves the [SVD](#) of the Jacobian matrix, which can be computationally expensive for larger kinematic chains [\[100\]](#).

The damping profile technique is another alternative in the literature. It is simply varying the damping factor according to specific profile based on certain criteria, such as the number of iterations or the proximity to singularities. The studies of [\[111, 112\]](#) are good examples in which the damping factor is changing according to an exponential profile. The damping factor varies around zero when the actuator is far from any singularity region, then increases exponentially when approaching a singularity. Afterwards, the damping factor decreases exponentially to zero when the actuator passes through the singularity region. These studies provided smooth actuator motion when approaching singularities. However, they used a machine learning method known as the Genetic Algorithm, which requires excessive training to find the optimum variables for their profiles.

Nonetheless, this technique continues to be underrepresented in the existing literature, with limited attention given to its possible advantages and drawbacks. Few studies addressed the specific nuances and applications of damping profiles based on iteration counts. This uncharted territory reflects a gap in our understanding of how this technique can impact the field of robotics. The limited attention given to this technique in the literature points to a compelling avenue for future research and investigation opportunities to delve deeper into the concept, providing valuable insights and contributions to the field of SPA inverse kinematics.

2.1.7 Quadratic Approximation

As mentioned earlier in Section 2.1.5, the quadratic approximation is obtained by truncating the Taylor series after the second derivative term. The linear approximation of the Taylor series is simpler and faster to compute. Though, it may have shortcomings regarding the accuracy requirements for some applications. The linear approximation assumes that the approximated function has a local linear trend at the calculation point, Figure 2.6(a), which means that the curvature of the function is negligible. However, the linear approximation may lead to some errors or slow convergence when working with certain non-linear functions with high curvatures, such as trigonometric functions [113].

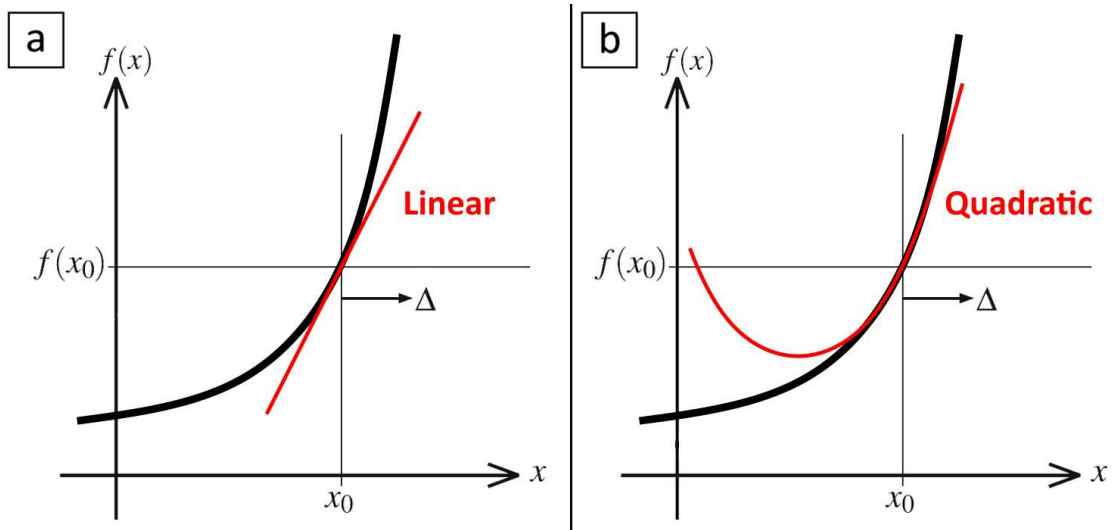


Figure 2.6: Taylor series expansion of an arbitrary non-linear function $f(x)$ where (a) represents a linear approximation at x_0 , and (b) represents a quadratic approximation at x_0 . The expansion is marked with red colour.

Alternatively, in some applications, the robustness and enhanced accuracy of the quadratic Taylor series approximation may provide better results, despite its complexity and higher computation loads. The quadratic approximation takes into consideration the curvature of the function, and it has better capability to capture its non-linear behaviour at the approximation point, Figure 2.6(b). For certain functions, however, that have low curvature or are locally linear, this might not be required. The quadratic approximation can be unnecessary or over-fitting in this situation. The choice of implementing either the linear or quadratic approximation

in any application depends on the trade-off between simplicity vs accuracy, as well as the characteristics of the application and its specific requirements [114].

Although the quadratic approximation has better accuracy and converges with less iterations compared to the linear approximation, it necessitates extra computational resources and can run into numerical issues, such as ill-conditioning or non-positive definiteness of the Hessian matrix [115]. Some methods were proposed to reduce the Hessian computational cost by approximating the Hessian matrix instead of the exact calculation such as Broyden-Fletcher-Goldfarb-Shanno (BFGS) method among other methods [100, 116, 117]. Other methods were introduced as well, such as the *Lazy Hessian* [114], in which the iteration algorithm does not update the Hessian matrix at each step and re-uses the old values for several steps. Moreover, another study implemented the *Halley's* method for solving IK [118], which can be considered an extension of the Newton's method. All these methods are outside the scope of this thesis. Nonetheless, the Newton's method can be highly accurate and may perform better when using the exact Hessian instead of the approximated Hessian [119].

Some studies investigated the advantages of using the Hessian matrix in modelling the kinematics of the continuum actuators. For instance, the studies of [120, 121] provided spatial dynamics models for PCC-based multi-segment continuum actuators. The analyses involved deriving the Jacobian matrices and the Hessian tensors. These studies, however, didn't investigate the IK, and their focus was to model the forward kinematics and dynamics.

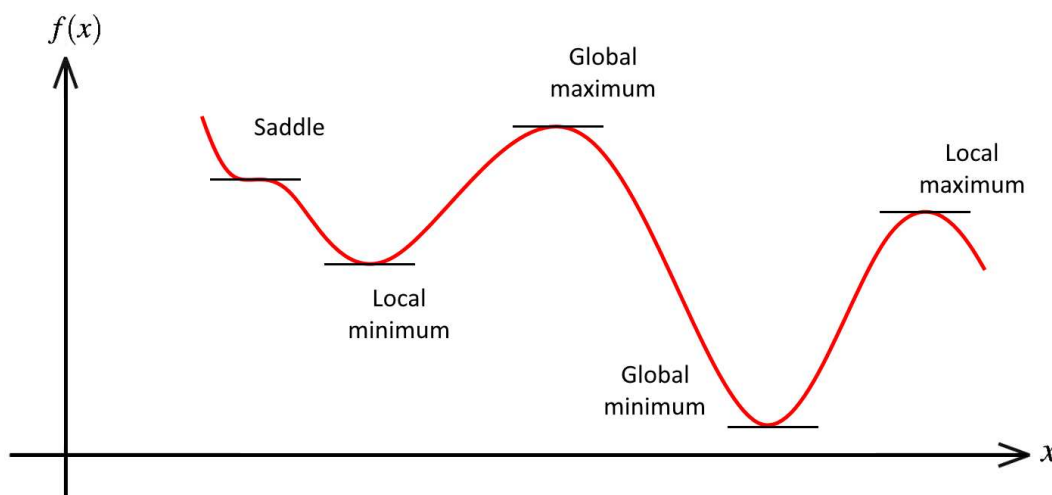


Figure 2.7: Critical points of an arbitrary non-linear function $f(x)$ showing the minimum and maximum points, local and global, as well as the saddle point. The gradient at each critical point is zero.

The quadratic approximation is usually formulated as an optimisation problem [116]. The idea is to select a suitable objective function, the position and/or orientation error minimisation as an example, then iterate till reaching a solution. The Hessian matrix measures the local curvature of the FK model at the approximation point. Therefore, the characteristics of the Hessian matrix play a vital role in determining the solution type [122]. The quadratic approximation convergence requires that the

Hessian matrix can be derived, i.e., the FK model is twice differentiable, and that the Hessian matrix is both invertible and *Lipschitz continuous* [94]. Otherwise, the iteration may enter a cycle of more than one point and not converge at all, or it may converge to a saddle point instead of a local or global minimum.

The solution is normally located at the *critical points* where the gradient of the FK is zero or undefined [98, 113], Figure 2.7. Based on the eigenvalues of the Hessian matrix, the critical point can be either a local minimum, a local maximum, or a saddle point [122, 123]. If the Hessian matrix is positive definite, all the eigenvalues are positive, then the solution is at a local minimum. Alternatively, if the Hessian matrix is negative definite, all the eigenvalues are negative, then the solution is at a local maximum. However, if the Hessian matrix has a mix of positive and negative eigenvalues, then the solution is at a saddle point. Finally, if the Hessian matrix is positive semi-definite, some of the eigenvalues are zero while others are positive, then the critical point type depends on the higher order derivatives of the FK.

In addition, the selected objective function can determine whether the optimisation is a *convex* or *non-convex* problem [97]. The limitation with using Newton's method to minimise a non-convex function is that it does not distinguish between the different types of the critical points. Thus, the applicability of Newton's method is restricted to cases where the Hessian is positive definite, for error minimisation, and Lipschitz continuous. Also, the solution may converge to a locally optimum point instead of a globally optimum point based on the initial seed. A key characteristic of convex functions within certain domains is that every locally minimum point is also a globally minimum point [97]. If the objective function is strongly convex and with Lipschitz Hessian, then the sequence generated by Newton's method will converge to a global minimum [94, 97]. This condition applies for both the linear and quadratic approximations.

As mentioned above, the objective function can be selected to minimise the position and/or orientation errors of the actuator end point. As an example, the position error can be the simple mathematical difference between the current and desired coordinates [43] which is a non-convex function, or the Euclidean norm which is a convex function [97, 124]. Similarly, the orientation error can be the difference between the current and desired orientations. The Euclidean norm could be a suitable choice to use since the orientation can be represented using various methods, such as Euler angles, rotation matrices, or unit quaternions.

In addition, a combination of both the position and orientation errors [58] can be used along with weighting factors to adjust the relative importance of position and orientation control. A desired trade-off between position and orientation accuracy can be achieved by tuning these weights. Anyhow, dealing with orientation representations and their corresponding error metrics involves further complexity compared to dealing with position alone. The specific requirements of each application may determine the necessity for including orientation control.

Generally speaking, the quadratic approximation requires deriving the Hessian twice. The first Hessian is the second partial derivatives of the FK model and it is a third-rank tensor [125]. The other Hessian is for the objective function and it is a second-rank tensor, a matrix, which is derived using the chain rule [126]. More on that in Chapter 3, Section 3.2.2.

The implementation of convex optimisation in IK is a known approach in robotics research [127] as well as other fields such as computer graphics [119, 128]. However, within the field of soft robotics, this approach remains a gap in the current literature, which emphasises the significance for further investigation.

2.2 Twist Measurement: Current Solutions and Challenges

As stated earlier in Chapter 1, SPAs are recognised for their inherent flexibility and capacity to undergo large deformations. However, they fall behind their rigid counterparts in the ability to sense their state in planar or 3D spaces. Given that SPAs deformations can result from both internal and external loads, improving their sensing capabilities holds the potential to refine the accuracy of state estimation. Such enhancements, when integrated, stand to substantially elevate the efficiency of control mechanisms in contrast to the conventional open-loop control paradigm [4].

2.2.1 State Sensing at a Glance

Generally, SPAs cannot accomplish required tasks effectively with just open-loop control, even with a carefully built environment [35]. This limitation is based on two reasons. First, SPAs are usually made from hyper-elastic material that has certain characteristics such as non-linearity, hysteresis, and large deformation. Predicting the SPA behaviour using kinematic and dynamic models alone represents a complex and challenging task considering such characteristics. Second, SPAs deformation is caused by both internal and external loads. While internal loads are usually modelled, external loads, on the other hand, are generally unknown. Thus, state sensing can provide useful feedback for the controller in order to adjust the SPA state based on the sensed errors.

State sensing can be achieved by using external monitoring, such as motion capture cameras [129] or electromagnetic tracking [130]. However, these techniques are designed to work within a lab environment. Alternatively, proprioceptive sensing [35] can sense the actuator state inside or outside the lab environment. Sensors are commonly categorised based on their working principle [131, 132] into exteroceptive, proprioceptive, and external perception.

Exteroceptive sensors are located on the 'robot' body and generate information about the external environment. Common exteroceptive sensors are compass, GPS, laser rangefinder, and haptic sensors. Proprioceptive sensors are located on the 'robot' body and measure its internal state. Common examples are pressure, strain, and temperature sensors. External perception sensors are located outside the 'robot' body to measure its state, such as motion capture cameras, RADARs, and infrared thermometers.

Proprioceptive sensing can provide sensory feedback to a closed-loop controller [35] for enhancing controlled movement accuracy and efficiency. Several proprioceptive sensing solutions for PCC actuators were introduced using different technologies, such as optical [36, 37], inductive [38, 39], Carbon nanotube (CNT) [40], conductive silicone sheets [28], conductive fluids [133], EGaIn [41], piezoresistive [134], and

capacitive [42] sensors. The sensing performance can vary between each solution based on the intended application [24].

All the aforementioned proprioceptive solutions measure the local change in length or curvature between two points on the actuator surface or inside its body based on the sensor position. Numerous criteria can affect the selection of suitable sensing solutions, such as resolution, stretchability, dynamic range (the ratio of sensing range to resolution), response time, robustness, durability, fabrication complexity, cost, design flexibility and scalability, and power consumption [35].

For soft actuators comprised of a singular muscle, a requisite condition involves the incorporation of at least one sensor to detect extension or bending along a single axis [135, 136]. In contrast, when dealing with soft actuators featuring multiple muscles, the demand escalates to the utilisation of two or more sensors. This expanded sensor array becomes essential for capturing changes occurring across multiple axes [28, 137].

2.2.2 Twist Along the Centreline

The twist assessment along the soft actuators' centreline instigated the development of diverse techniques, each exhibiting variations in accuracy and modelling complexity. In an early approach, inertial measurement units (IMUs) were strategically positioned along the centreline of a flexible bar to capture relative movements at specific measurement points [138].

While this study offers a viable method for measuring twist along the centreline, it was originally devised for the purpose of mapping human gestures during the use of prosthetic musical instruments and has nothing to do with soft robotics. The focus of the study was unintended in the context of soft robotics and it involves embedding non-extensible components within the soft actuator core.

Later, a noteworthy investigation conducted by [139] outlined the fabrication and testing procedures for a soft actuator featuring dual pneumatic muscles. This actuator was capable of undergoing deformations that involved both bending and twisting. The study examined the use of [EGaIn](#)-based sensors to assess both bending and twist deformations. Despite its contribution to the field through the experimental implementation of superficial sensors for twist measurement, the study lacked the presentation of a formal mathematical model for quantifying twist. Furthermore, the results were presented solely as raw sensor readings, without a comprehensive analytical framework.

Few research endeavours delved into the integration of machine learning methodologies for the modelling of soft actuators sensing, showcasing diverse applications. Notably, the investigation conducted by [140] demonstrated the application of supervised learning techniques. Their focus was on identifying a strategically optimised subset of [EGaIn](#)-based sensors from a larger redundant set. The primary objective was the development of soft actuators endowed with the ability to execute task-relevant sensing while utilising a minimal number of soft sensors.

Similarly, another significant exploration in this domain was undertaken by [141], wherein coupling field simulation (CFS) was employed to simulate capacitive su-

perificial strain sensors deployed on the surface of an SPA. The core objective here was to characterise the sensors' performance under varying conditions of bending and twist deformations. The researchers were able to create a considerable volume of annotated data by utilising the insights that they acquired from the simulation. This data was then utilised to train a neural network for a spectrum of perception tasks.

While these machine learning-driven methodologies demonstrate proficiency in handling complex designs and deformations, it is crucial to acknowledge their inherent demand for extensive, high-quality datasets for both training and testing phases. Additionally, the computational resources required for the implementation of these techniques can be significant [142].

Expanding the spectrum, another approach involved the integration of a large array of randomly positioned sensors within the soft actuator core [41]. The objective was to select the smallest suitable subset of sensors by quantifying and minimising the potential energy fluctuations. However, the implementation of such a measurement technique invariably demands the incorporation of many sensors within the actuator structure or the integration of non-extensible components within the soft actuator core.

The implementation of such designs introduces inherent challenges and complexities, particularly concerning the accuracy of sensing and the spatial constraints within the soft actuator. Contrarily, fibre Bragg grating (FBG) optical sensors attracted significant research attention due to their softness and other advantageous qualities. The application of FBG sensors on soft robots was extensively studied over the years, as elaborated in the following section.

2.2.3 Twist Measurement Using FBG Sensors

FBG sensors were emerged as versatile tools, showcasing enormous potential in a wide range of applications that necessitate precise deformation sensing. Their popularity in these contexts is attributed to an array of advantageous properties, including remarkable temperature tolerance, immunity to electromagnetic interference, inherent flexibility, impressive tensile strength, and a compact design [143]. While the extensive utilisation of FBG sensors in deformation sensing is well-documented, their application in measuring twist, particularly within the domain of soft actuators, is relatively understudied.

The compliance and adaptability of soft actuators pose unique challenges in the integration of sensing technologies. The difficulties involved in measuring twist in such pliable structures have led to a limited number of studies exploring FBG sensors in this specific context [36, 144, 145]. This limitation is primarily attributed to the inherent complexity associated with soft actuators, where traditional sensing approaches face challenges in providing accurate measurements.

In parallel, FBG sensors have found applications in continuum actuators [146], showcasing their adaptability across diverse fields [143, 147, 148]. The versatility of FBG sensors becomes evident as they transcend the boundaries of soft actuators and spread to various engineering disciplines. However, despite their promising capabilities, certain studies within the realm of soft actuators have highlighted limitations

and drawbacks.

One notable example is found in the theoretical model presented by [144], which utilises the torsion of a spiral curve [149] as opposed to focusing on the twist about the centreline. This departure from the conventional approach raises concerns about the model's ability to accurately represent strain changes in terms of the twist at the free end of the actuator. The deviation from established measurement conventions may impact the broader applicability and reliability of the proposed model.

Similarly, the study conducted by [145] adopts a unique approach by placing FBG sensors along a spiral nitinol wire to safeguard them from excessive strain. However, this design choice, while addressing one challenge, introduces another. According to established mechanical design principles [150], the use of a higher rigidity material such as the nitinol wire may alter the twisting measurement in the soft actuator. The behaviour observed during twisting, where the helix diameter decreases rather than changing its length, as depicted in the results [145, Fig. 5(b)], raises questions about the consistency and reliability of the measurements. Additionally, the absence of a direct comparison between the theoretical model and the experimental twist results introduces uncertainties regarding the effectiveness of this particular design.

Moreover, the study outlined in [36] takes a different approach by not including a theoretical model and primarily focusing on employing a commercial multi-core fibre optic shape sensor (FOSS) sensor to measure the deformation of the soft actuator, Figure 2.8. While this approach may provide practical insights into deformation, the absence of a theoretical framework limits the understanding of the underlying mechanics governing twist in soft actuators.

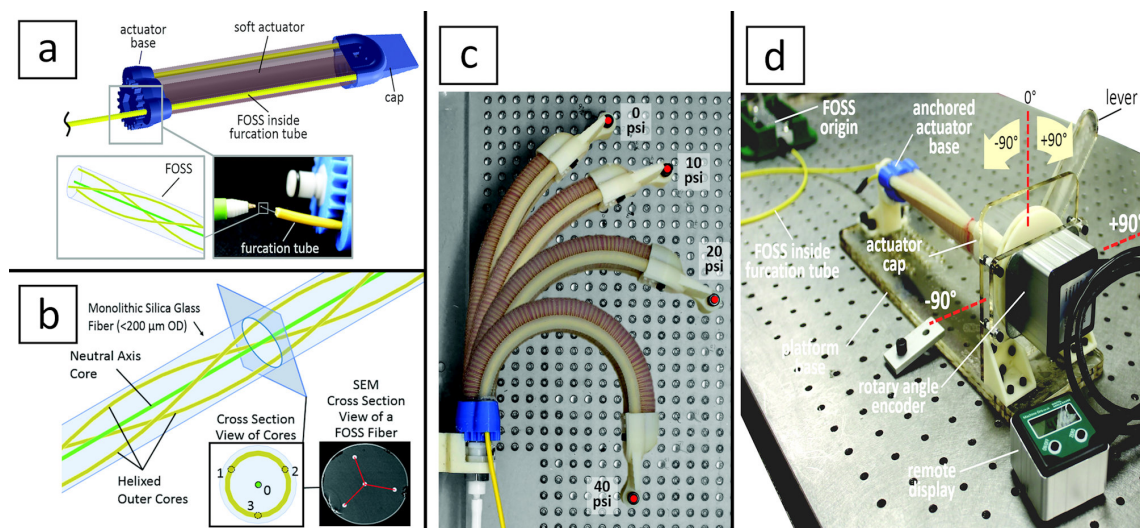


Figure 2.8: An example of integrating a commercial multi-core FOSS in a soft actuator for measuring bending and twist where (a) shows the FOSS's path within the soft actuator, (b) shows the cross-section view of the FOSS using the scanning electron microscope (SEM), (c) shows the bending test, and (d) shows the twist test. Adapted from [36] and reproduced under the terms of CC BY 4.0 license.

Furthermore, in the study conducted by [146], the initial twist test presented a noteworthy concern as the authors did not provide a comparison between their theoretical model and the test results [146, Fig. 5]. This omission raises questions about

the accuracy and reliability of their proposed model, emphasising the importance of verifying the theoretical frameworks through simulations and/or validating through experimental data.

As discussed above, the exploration of twist measurement in soft actuators was marked by a limited number of studies, as evidenced by the examinations of various investigations [36, 144, 145]. While these studies employ FBG sensors as a means of capturing twist, the broader emphasis lies in understanding the complex behaviour of soft actuators during twisting. Challenges abound in this domain, from the formulation of comprehensive theoretical models to ensuring consistent experimental validation and adherence to established mechanical design principles. The findings from these studies underscore the complexities inherent in measuring twist within flexible structures, prompting the need for more extensive research to enhance our understanding of this unique aspect of soft actuator behaviour.

In a study with relevance to the investigation of soft actuator twist, as discussed by [151], electromagnetic sensors were used to measure the twist within a PCC-based cable-driven actuator. The twist observed in this scenario was a consequence of imperfect assembly as a result of a manufacturing error. The study examined the complexities of the actuator's behaviour, with the authors formulating a series of mathematical equations to articulate the length variations of the twisted cables along the actuator's backbone. However, this meticulous work necessitated the determination of multiple calibration factors through experimental measurements. While these equations encapsulate the dynamics of twist in cable-driven actuators, they also open avenues for future research to construct a comprehensive mathematical model that encompasses a broader spectrum of soft sensors subject to diverse movements, including twist.

All things considered, I believe that a viable approach to sensing deformations in soft actuators involves the attachment of either a superficial or a sensory skin to their external surface, as explored in studies such as [28, 152], or even their internal surface, as indicated in [144]. This strategy provides seemingly less intricate solutions, potentially simplifying the measurement of various deformations, including bending, rotation, extension, and twist.

Notably, EGaIn-based sensors hold a significant advantage over FBG sensors, especially in cases involving extension. The remarkable difference in strain measurement range becomes apparent, with FBG sensors are limited to millistrains [153], whereas EGaIn-based sensors can exceed an impressive 700% strain [154], highlighting their superior range and versatility.

Nevertheless, a significant gap persists in the current research despite the progress made in advancing the sensing capabilities of soft actuators. Specifically, there is a lack of superficial solutions that can comprehensively sense and quantify these multifaceted deformations simultaneously in three-dimensional (3D) space. Closely addressing this gap may lead to further developments in soft robotics. Hence, allowing for the development of sophisticated and adaptable capabilities to be included in these systems.

This highlights a critical research requirement for the development of integrated sensing solutions that can offer a comprehensive understanding of the complex de-

formations exhibited by soft actuators in real-world applications. For this reason, the pivotal role of verification via simulation process cannot be overstated. Leveraging simulation capabilities allows researchers to refine and verify mathematical models, providing a better understanding of SPA behaviours. Recognising the crucial interplay between sensing modelling and verification is essential for advancing SPA understanding. This sets the stage for the exploration of recent FEA simulation advancements in soft robotics, as discussed in the following section.

2.2.4 An Overview of SPA Simulation Tools

Analytical modelling of soft actuators presents a formidable challenge due to their complicated geometries, substantial material non-linearities, and the inherent compressibility of air [155, 156]. Several recent investigations exploring the application of FEA in the realm of soft robotics yielded valuable insights [156–160]. These studies collectively establish that FEA possesses the capability to handle extensive deformations associated with both deformation and inflation, predict and optimise the performance of SPA designs under diverse inputs, Figure 2.9, offering a swift and efficient design strategy that minimises costs and development time, enhance our comprehension of stress concentration and strain distribution in SPAs for evaluating fatigue performance, and address contact non-linearities that are caused by environmental interactions.

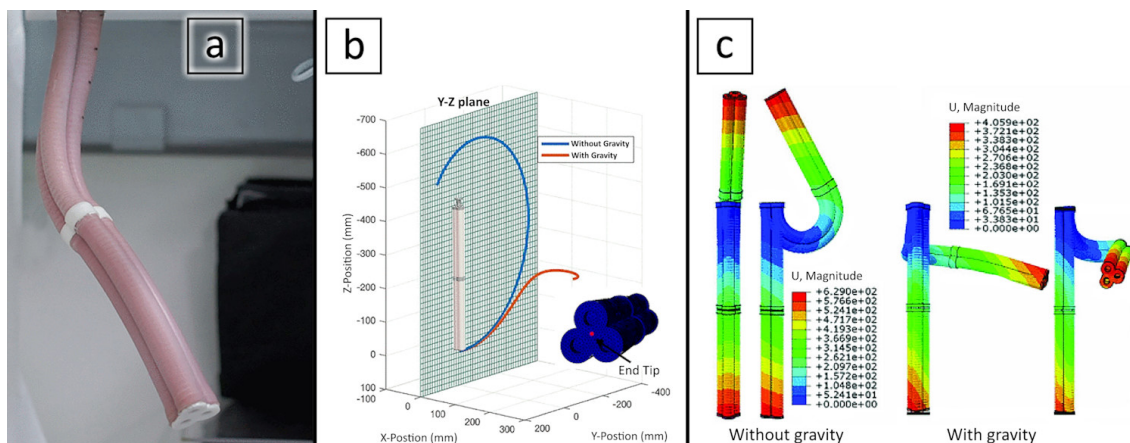


Figure 2.9: An example of model verification via FEA simulation and validation via experimental tests where (a) shows the SPA while being actuated, (b) shows the mathematical model results, and (c) shows the FEA simulation results. Adapted from [160]. Copyright 2019, IEEE.

Prominent commercial FEA software tools were employed in the simulation of soft robotics, including Abaqus, ANSYS, and COMSOL. These tools provide various features such as force measurements, design optimisation, modelling of interactions with other objects, and analysis of multi-physics phenomena [161], including fluid-structure interaction [161] and thermo-structural analysis [162, 163]. Meanwhile, open-source alternatives like MOOSE and VegaFEM cater to the simulation needs of soft actuators [164]. VegaFEM, for instance, were utilised in modelling soft pneumatic fingers with a fibre-reinforced design [164]. However, it's noteworthy that MOOSE is confined to the Neo-Hookean hyperelastic model [165], and VegaFEM lacks collision detection or contact handling capabilities [166].

Although these FEA packages are practical, their computational speed frequently becomes a bottleneck, restricting their use in scenarios involving real-time control. Addressing this concern, the SOFA toolkit, tailored for interactive medical simulation [167], incorporates a soft robotics plugin [168] to facilitate fast, real-time simulation and control [166, 169–171].

Notably, many FEA studies predominantly employ quasi-static simulations, wherein pressure loads are gradually increased at small time steps. However, dynamic effects may necessitate consideration in scenarios involving high pressures or rapid actuation, where the quasi-static assumption becomes unfeasible, leading to the potential observation of vibrations [172, 173].

In contrast, differential simulators, a subset of physics-based simulators, integrate gradient-based optimisation algorithms. These algorithms enable the direct input of calculated gradients into numerical optimisation algorithms, forming a mathematical framework to identify and bridge simulation-reality gaps specific to applications, optimally control embedded soft actuators for tasks such as grasping and locomotion, and estimate the mechanical state of the soft systems based on optimally embedded sensors [174].

A significant progress in this domain is the creation of simulators that can train and evaluate controllers, namely those derived from reinforcement learning. For instance, ChainQueen stands out as a real-time, differentiable hybrid Lagrangian-Eulerian physical simulator designed for deformable objects, enabling physical inference, soft robot control, and co-design of robotic arms [175] but with limitations in handling visco-elastoplastic deformation as well as the number of applications so far considering the short time since this simulator was introduced [176].

Moreover, Differential Projective Dynamics (DiffPD) serves as a rapid differentiable simulator inspired by projective dynamics, specifically tailored for the optimisation of soft-body learning and control applications [177]. This simulator is integrated with a differentiable, analytical hydrodynamic model, further enhancing its utility in the comprehensive modelling and control of underwater soft robots [178]. Despite its advantages, DiffPD has limited performance in regards to simulations involving hyperelastic material or contact interactions that need to be addressed for better optimisation strategies [177].

In the same fashion, traditional rigid body simulators were adapted to accommodate soft robotics applications. Gazebo and ROS, as evidenced in [179, 180], were utilised to simulate robotic manipulation, incorporating a rigid robotic arm equipped with a soft pneumatic gripper. SoRoSim, a MATLAB toolbox employing the geometric variable strain approach, offers a unified framework with various capabilities for modelling, analysis, and simulation of different types of actuators, including soft, rigid, and hybrid [181]. Another notable example was Evosoro [182], a soft robot simulator based on Voxelyze, a versatile voxel-based soft-matter physics engine supporting static and dynamic analyses [183]. Additional robotic simulators can be seen in the literature, each of which is equipped with capabilities for modelling soft robotic components [5, 184].

In general, FEA stands out as a widely employed tool for simulating the deformation of SPAs. Leveraging its versatility, FEA effectively captures the behaviour of these

actuators under various types of loading, accommodating scenarios involving complex geometries, varied materials, and different boundary conditions [2]. Drawing from my familiarity with FEA, having utilised it in prior research, I find it to be a powerful and effective tool for simulating SPA behaviour, particularly in quasi-static analyses where dynamic considerations are not in play.

However, it is essential to acknowledge certain considerations associated with the use of FEA, including the demand for substantial computational resources and the inherent limitations in simulating dynamic behaviours [5, 155]. Despite these considerations, my preference for FEA originates from its proven capability to meet the requirements of my mathematical analysis, which primarily focuses on quasi-static behaviours. Additionally, the wealth of resources within the soft robotics community provides valuable guidelines and insights, offering assistance when encountering challenges in building sophisticated FEA models.

While alternative simulation tools may offer solutions to specific limitations of FEA, it is crucial to recognise that each tool comes with its own set of constraints. Researchers engaging in the study and analysis of SPA behaviours should be mindful of computational demands and potential constraints, making informed decisions about the most suitable tool for their specific research objectives.

2.2.5 Multi-Muscle SPA: Insights from Previous Studies

A standard pneumatic actuator typically comprises one or more pneumatic muscles, alongside additional supporting layers tailored to facilitate specific movements [185]. Pneumatic muscles, such as McKibben muscles, exhibit contraction under the influence of air pressure [186]. Nevertheless, achieving diverse movements, such as extension, twist, or bending, necessitates further design adaptations.

To enable a broader range of motions within a single actuator, the incorporation of two or more pneumatic muscles is imperative, allowing for the generation of bending, extension, and twist movements in three-dimensional space [17]. An effective strategy involves integrating fibre-reinforcements along the surface of the external soft actuator or individual pneumatic muscles to restrict radial expansion. Consequently, longitudinal expansion occurs, which can be harnessed to actuate simple joints [187] and generate twisting [188] or complex motions [189].

Moreover, research indicates that by strategically adjusting the orientation of the fibre-reinforcements [190] or employing an inverse actuation mechanism [191], researchers successfully generated a spectrum of motions, emphasising the adaptability and versatility of multi-muscle SPAs. This adaptability is particularly advantageous in scenarios where complex movements are essential for the intended application.

A noticeable soft actuator in the realm of robotic solutions for minimally invasive surgery (MIS) is the STIFF-FLOP (STIFFness controllable Flexible and Learn-able manipulator for surgical OPERations), which underwent iterative design refinements, yielding valuable insights and discoveries that significantly advance the field of MIS soft actuators [192].

The initial design, presented in [193], featured segments comprising three pneumatic chambers with semi-circular cross-sections serving as actuation muscles. Addition-

ally, each segment included a central chamber with a circular cross-section for granular jamming and an external braided sheath around the segment body to restrict radial expansion. Notably, the concept of jamming was pivotal in modulating stiffness within individual segments, thereby inducing specific behaviours in the SPA. The authors utilised coffee beans as the jamming material and manipulated stiffness by controlling the vacuum level in the central chamber. The utilisation of granular jamming for stiffness control offered a dynamic mechanism to adapt the actuator's properties during operation.

Although the authors introduced a conceptual design for a modular MIS actuator, their study lacked a comprehensive mathematical model. Experimental tests primarily focused on characterising the design under various loading conditions, encompassing extension, bending, rotation, and stiffening. Despite the absence of a formal mathematical framework, the experimental analyses provided crucial insights into the actuator's mechanical response under diverse loads, laying the groundwork for subsequent developments in the mathematical modelling of similar SPAs [194].

Subsequent to the original design of the STIFF-FLOP, a series of refinements were introduced by [137] to optimise its performance. The authors identified challenges associated with the integration of optical length sensors, drawing inspiration from the work presented by [195], and proposed effective solutions. These challenges resulted from issues related to the inflation of individual chambers within the actuator. While the external braiding effectively restrained outward inflation of the pressure chamber, redirecting expansion primarily inward, it unintentionally altered the internal structure of the segment. This modification led to undesirable effects, including non-linear actuation, dependence of length sensor readings on chamber pressures, and the intricacies of mathematical modelling.

A primary concern addressed by [137] was related to the individual chambers' cross-sectional area. The introduction of pressure into the actuation chamber prompted changes in its cross-sectional area, a phenomenon further complicated by dependencies on pressures in other chambers. This non-linear relationship between actuation pressure and internal force resulted in heightened sensitivity to pressure variations, presenting a significant challenge in achieving consistent and predictable actuation.

A second critical issue addressed by [137] involved the displacement of the cross-sectional centre of individual chambers. The chamber actuation caused a deviation from the original semi-circular shape of its cross-section, since it was limited by the adjacent chambers and the external braiding. Consequently, the displacement of the chamber's cross-sectional centre became dependent on the actuation pressure value. This behaviour manifested as a reduction in the achieved bending angle when actuating two chambers compared to the bending angle achieved by actuating a single chamber at the same pressure value, as illustrated in Figure 2.10(c).

A third concern addressed by [137] relates to the friction between the silicone material and the braided sheath. The interaction between the outer silicone surface and the braided sheath introduces friction, leading to the dissipation of actuation energy. This frictional effect restricts the extension of the affected chamber when two chambers are actuated sequentially, illustrated in Figure 2.10(b). Consequently, this limitation impedes the segment's ability to rotate while bending, significantly influencing the attainable shape of the workspace.

The fourth issue revolves around the interaction with optical sensors. These sensors are embedded in channels positioned along a concentric circle with the segment's cross-section, as depicted in Figure 2.10(a). The irregular inflation of chambers, as evident in Figure 2.10(c), affects the sensor measurements, leading to discrepancies in the recorded lengths. This discrepancy complicates the process of accurately reconstructing the segment's deformation and understanding its dynamic behaviour.

To mitigate or eliminate these issues, [137] proposes several modifications. First, they advocate for changing the cross-section of pneumatic chambers to a circular shape. Second, they suggest the use of braiding around each individual chamber, rather than encompassing the entire segment. Additionally, the authors recommend substituting the sheath with thread for the braiding. In this modified configuration, chambers are braided using a thin thread applied as a tight helix around the chamber's surface, allowing for expansion in longitudinal directions while impeding radial inflation, as illustrated in Figure 2.10(d).

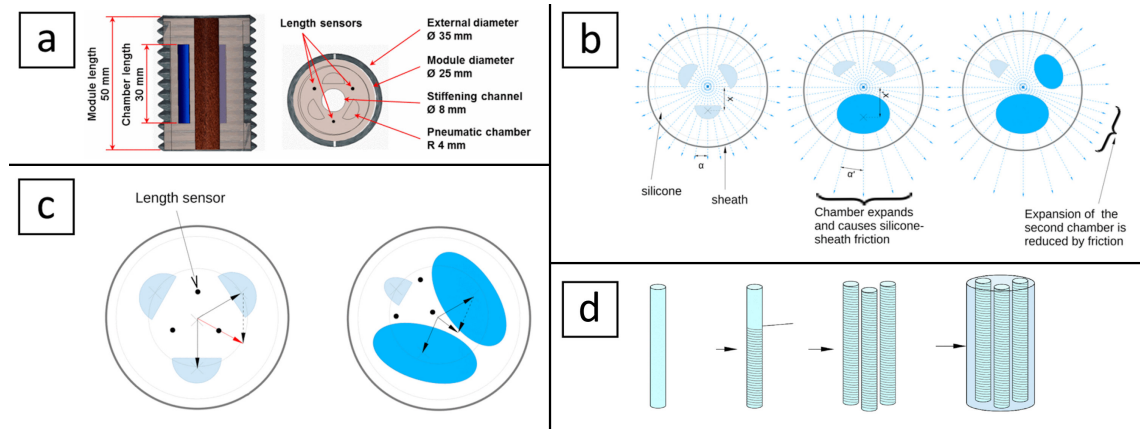


Figure 2.10: An illustration of **STIFF-FLOP** actuator issues as identified by [137] where (a) shows the initial design as suggested by [193] and the location of the embedded optical sensors, (b) shows the segment cross-section at initial state (left) with no friction between the sheath and silicone, the contact between the sheath and the segment body due to a single chamber actuation (centre), the effect of the sheath friction in restricting the expansion of the second chamber (right), (c) shows the displacement of the cross-sectional centre of individual chambers when actuated, and (d) shows the braiding of the individual chambers instead of just the segment external surface. Adapted from [137]. Copyright 2015, IEEE.

Analogous to the approach taken in [193], the study by [137] does not present a formal mathematical model for the **STIFF-FLOP** actuator. However, the proposed modifications hold promise for addressing the identified issues and improving the overall performance of the **STIFF-FLOP** actuator, highlighting the continuous refinement and adaptation of **SPAs** in terms of both design and implementation.

Subsequently, a comprehensive analytical modelling approach for the **STIFF-FLOP** actuator was presented in another study, employing the geometry deformation approach as detailed in [194]. The focus of this investigation was on modelling the deformation during extension and bending movements, particularly under high external forces, Figure 2.11(a,c). The study utilised both the **PCC** and **VC** models to

analyse the actuator's behaviour. Notably, special attention was given to the deformation of the circular cross-section of the actuator under the [PCC](#) assumption.

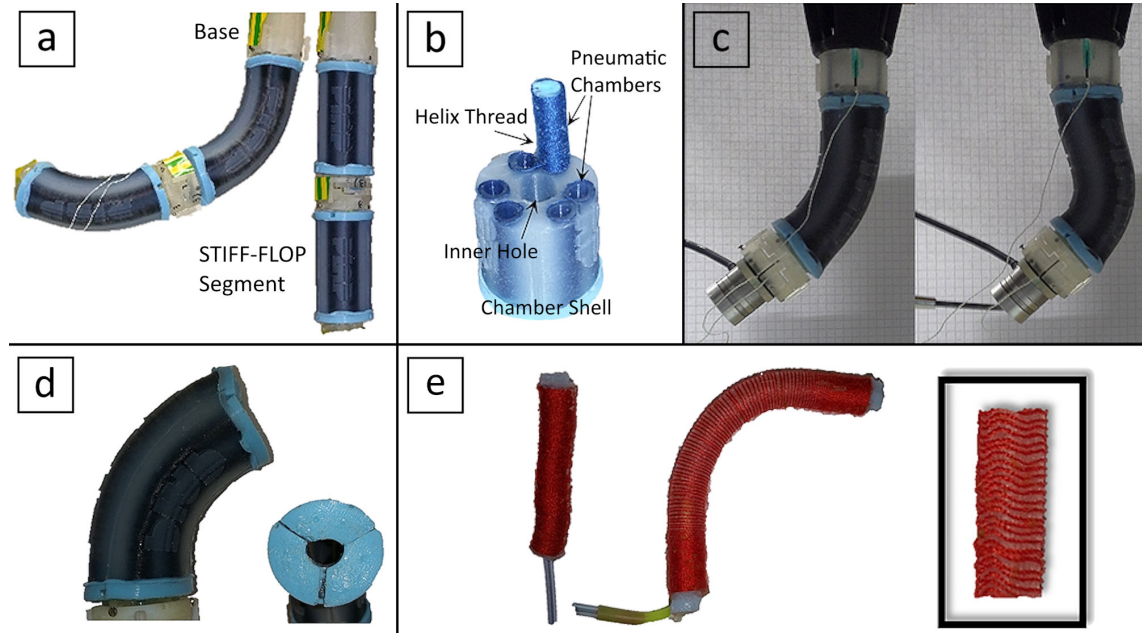


Figure 2.11: [STIFF-FLOP](#) actuator with twin chambers design where (a) shows an assembly of two segments under testing, (b) shows the segment cross-section with the twin chambers design, (c) shows the testing of the actuator under external loads, (d) shows the deformation of the actuator's cross-section while bending, and (e) shows the effect of braiding thread in causing bending instead of pure extension. Adapted from [194] and reproduced under the terms of CC BY 4.0 license.

In the pursuit of enhanced performance, this study introduced additional design modifications. Notably, the shift from single chambers to twin chambers, resulting in a total of six chambers per segment as shown in Figure 2.11(b), aimed to augment the actuator's capabilities. Furthermore, a hollow core was incorporated to facilitate the passage of pressure pipes to the subsequent segment. To address potential unwanted deformations, a body shell was added to the external surface of the actuator, as depicted in Figure 2.11(a,b). Nevertheless, there were challenges with this design.

The actuator's cross-section exhibited deformation from a perfect circular shape during bending as seen in Figure 2.11(d), introducing complexities in the expected geometrical behaviour. Also, undesirable bending of pneumatic chambers was observed due to inhomogeneity in the braiding and tube moulding processes. Moreover, the braiding threads, while serving as a structural element, partially constrained the extension of chambers, potentially leading to undesired bending instead of pure extension, as demonstrated in Figure 2.11(e).

These design complexities highlight the inherent challenges in achieving a balance between structural integrity, flexibility, and desired actuation characteristics. Despite the identified issues, this study contributes valuable insights into the analytical modelling of the [STIFF-FLOP](#) actuator and highlights the ongoing quest for innovative design solutions in the realm of soft robotics.

Subsequent research endeavours aimed at refining the design and mathematical mod-

elling of the **STIFF-FLOP** actuator were conducted in several studies [66, 129, 196–199]. Despite the previously discussed limitations associated with using braided threads to constrain radial expansion, this concept continued to be prevalent in these studies. Nevertheless, it is important to point out that alternative reinforcement approaches were investigated in the literature beside this one. An in-depth review for these alternative approaches is presented in the next section.

An essential consideration in understanding the behaviour of **SPAs** is the correlation between their behaviour and the number of pneumatic muscles (or chambers) they encompass. Typically, **SPAs** feature three pneumatic muscles positioned 120 degrees apart to facilitate three-dimensional movements [81, 200–204]. Some variations opt for twin chambers per muscle, resulting in six chambers instead of three, aimed at augmenting the generated force without increasing the actuator diameter [194, 196–198]. Other designs adopt four pneumatic muscles spaced at 90-degree intervals to achieve smoother movements [3, 28, 32, 34, 75, 205, 206]. Additionally, certain configurations incorporate one extra pneumatic muscle at the core of the actuator to enhance axial force [207].

In summary, the developmental stages of the **STIFF-FLOP** actuator underscore the efficacy of utilising braided threads around individual muscles, as opposed to braiding solely the external surface of the actuator, for improved actuation performance. Another crucial observation is that increasing the number of pneumatic muscles allows for greater force generation without enlarging the actuator diameter. This not only enhances controllability and motion smoothness but also provides better redundancy in case of muscle failure. However, this increased capability comes with the trade-off of added complexity for the control system as well as the associated control algorithm [3, 198].

2.2.6 Common SPA Reinforcement Approaches

In the preceding section, it was explained that a typical pneumatic muscle exhibits contraction when subjected to air pressure. However, achieving a diverse array of movements, such as extension, twist, or bending, necessitates further design refinements that incorporate a synergistic combination of two or more pneumatic muscles within the soft actuator.

An extensively adopted strategy involves the incorporation of reinforcement layers or components into the actuator to enhance stiffness in regions where deformation is undesired for the intended task [185]. The literature reflects a plethora of reinforcement techniques, including in-extensible layers, fibre-reinforcement, meshes, shells, and braided sleeves, each presenting distinctive advantages and drawbacks.

For instance, the utilisation of an in-extensible layer or fabric positioned on one facet of the actuator [208] facilitates the generation of bending and twisting movements. This configuration typically permits unrestricted expansion on the opposing side upon pressurisation, thereby inducing the required bending motion. A modified approach, as showcased in [209], entails the insertion of a pneumatic muscle between two distinct layers—knit and woven—each possessing different strain properties.

Upon pressurisation of the pneumatic muscle, the upper layer (i.e., high-strain knit) selectively experiences longitudinal strain, while the lower layer (i.e., low-strain wo-

ven) restricts strain in multiple directions, thereby inducing the bending motion. Despite conferring enhanced constraint for the pneumatic muscle, this dual-layer approach presents a formidable challenge in modelling the complicated non-linear behaviours exhibited by the textiles under varying loads.

On the alternative front, fibre-reinforcement emerges as a prominent approach employed across various studies, entailing the encasement of a moulded hollow soft core with in-extensible materials such as cotton or Kevlar. Then, a second soft layer is applied for effective sealing. The in-extensible fibres play a pivotal role in constraining the radial expansion of the internal pneumatic muscle, thereby influencing the resultant strain profile of the actuator. A subtle variation of winding geometries empowers researchers to exert precise control over the behaviour of the actuator, as evidenced in studies such as [210].

The strategic combination of actuators with diverse fibre geometries in series opens avenues for configurable trajectory matching, as demonstrated in a body of literature [157, 190, 211–214]. Despite its effectiveness, the fabrication of these actuators is complex, demanding meticulous control over factors such as fibre path, tension, and adhesion [215]. Consequently, devices resulting from this approach typically exhibit diminished extensibility and flexibility when compared to structures composed solely of elastomeric materials [216].

In a bid to substitute fibre structures with a more robust and manufacturable reinforcement strategy, wrapped soft pneumatic artificial muscles (WSPAMs) were emerged [217]. These WSPAMs leverage a thin silk mesh structure to constrain bloating, characterised by excessive radial expansion of the actuator core. However, the mesh structure is glued onto the actuator on all sides, significantly over-constraining the entire motion range of the actuator and preventing the delivery of large bending angles.

Although there is a degree of tunability in the bending angle by modifying the mesh pattern, a notable drawback surfaces: the issue of bloating intensifies as the wrap pattern incorporates thinner reinforcements. This complicated interplay between mesh patterns and bloating highlights the challenges in achieving both precise control and manufacturability in the design of SPA reinforcements.

Moreover, [215] introduced a moulding technique featuring pre-formed integrated reinforcement shells crafted from an in-extensible polyethylene terephthalate (PET) layer. The objective was to streamline the fabrication process by implementing a single-step moulding approach for both bending and linear actuators, thereby enhancing repeatability and design flexibility. In a parallel vein, [218] presented a bending SPA incorporating an origami shell reinforcement, enabling the sustainability of higher actuation pressure and the delivery of larger blocking torque. Adjusting the design parameters of the origami shell results in noticeable variations to the performance of the soft actuator.

In a complementary approach, [219] introduced an additive manufacturing method for composite material soft pneumatic actuators, designed to execute a spectrum of programmable motions. This innovative approach employed an open-source 3D printer to produce the soft actuator by extruding both the soft (silicone) muscle and the reinforcement (thermoplastic elastomer, known as TPE) shell. The flexibility

inherent in this methodology allows for the creation of multiple designs by varying the shell pattern, thereby enabling the manufacture of soft actuators capable of executing bending, contraction, or twisting movements. This manufacturing method facilitates rapid customisation of soft actuators through intuitive input parameters.

It is important to note that all the previously mentioned designs were specifically tailored for single-muscle actuators, aimed at generating individual movements—be it bending, extension, or twist—in planar space. These designs do not readily extend to multi-muscle actuators. The FEA simulation of each reinforcement is discussed within related references, shedding light on the complicated interplay between the chosen reinforcement techniques and the unique challenges posed by FEA, thus contributing to a comprehensive understanding of the complexities associated with soft actuator design and optimisation.

In the context of soft pneumatic actuators, as discussed in Section 2.2.5, the combination of two or more pneumatic muscles within a single actuator becomes imperative for the generation of diverse motions in three-dimensional space [17]. One conventional strategy involves incorporating fibre-reinforcements around the pneumatic muscles within the actuator, a technique showcased in designs like the STIFF-FLOP soft actuator [137], Figure 2.10(d), and other similar innovations [160]. The fibre-reinforcements restrict radial expansion while permitting longitudinal expansion, offering a means to generate various movements. This approach, while effective, has inherent limitations, as elaborated earlier in this section.

Alternatively, a less conventional approach entails applying reinforcement around the actuator itself rather than around individual pneumatic muscles. Instances of this approach can be found in various literature examples, such as the use of glass sponge skeleton [202], or the bellow sheath as in earlier STIFF-FLOP [193, 220] designs as well as other studies [221, 222]. However, the resulting motion from this approach is constrained to bending in three-dimensional space, lacking the capability for extension or twist.

A combination of these two approaches yields a robust design by employing a nylon braided sleeve around each pneumatic muscle, limiting radial expansion without relying on fibre-reinforcements. Unlike traditional fibre-reinforcement, braided sleeves permit unrestricted longitudinal expansion, extending freely up to a certain length. By precisely controlling the pressure in each pneumatic muscle within the actuator, sophisticated movements can be achieved [129, 221, 223].

It is noteworthy that the diameter of the braided sleeve exhibits an inverse relationship with its length [27, 224, 225]. Nonetheless, this change in diameter can be neglected when the braided sleeve operates near its maximum diameter, characterised by a dense braid [189, 193]. While the use of braided sleeves around pneumatic muscles enhances actuator robustness, the FEA simulation of these sleeves introduces considerable complexity into the model design, substantially inflating computation time, as discussed in the next section.

2.2.7 FEA Simulation Techniques for Braided Sleeves

Commercially available braided sleeves are commonly crafted from materials such as nylon or PET, predominantly employed in the fabrication of McKibben-type pneu-

matic muscles. These sleeves, resembling tubular coverings, are intricately formed by interweaving strands of material in a pattern analogous to braiding hair [226] as shown in Figure 2.12(a). The interwoven strands follow both clockwise and anti-clockwise directions, forming an angle with the sleeve's longitudinal axis referred to as the braid angle [27], Figure 2.12(b).

The dynamic nature of braided sleeves is evident in their ability to contract or extend, influencing the braid angle. At a critical value of 54 degrees, the cylindrical volume of the sleeve reaches its maximum [27], as illustrated in Figure 2.12(b). Some studies alternatively consider the complementary angle, approximately 35 degrees, denoting the angle between the strands and the sleeve radius [225]; however, the underlying concept remains consistent.

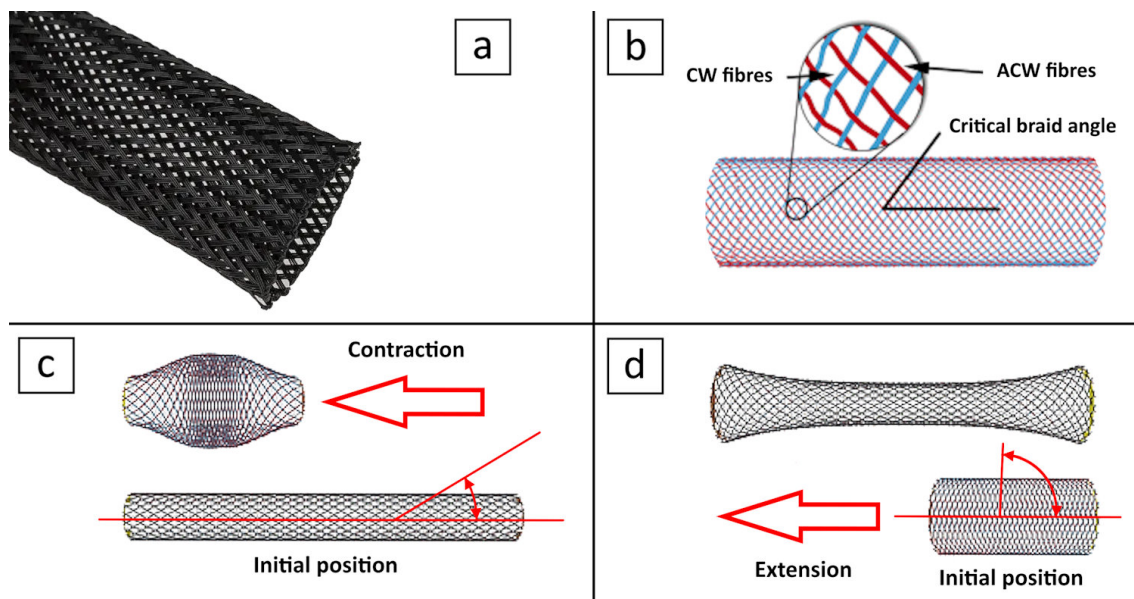


Figure 2.12: Braided sleeves as SPA reinforcement where (a) shows a commercial braided sleeve and its interweaving strands, (b) illustrates the critical braid angle as well as the clockwise and anti-clockwise fibres, (c) illustrates a contractile sleeve in which the initial braid angle is less than the critical value, and (d) illustrates an extensile sleeve in which the initial braid angle is greater than the critical value. Adapted from [27]. Copyright 2019, IEEE.

Crucially, the braid angle at the actuator's initial position determines whether the individual pneumatic muscle is contractile or extensile [224]. Starting from a braid angle less than the critical value leads to a contractile pneumatic muscle, and vice versa, as illustrated in Figure 2.12(c,d) respectively. In the context of a multi-muscle SPA, the braided sleeve must be extensile to facilitate the required movements in three-dimensional space. Selecting the maximum diameter of the braided sleeve to match the pneumatic muscle's diameter at rest, ensuring a dense braid, is pivotal. This choice guarantees a maximum braid angle at the initial position, compelling the reinforced muscle to extend upon pressurisation, Figure 2.12(d).

Various sophisticated techniques were applied for the FEA simulation of braided sleeves, each tailored to the unique objectives of the respective studies. In the work presented by [227], the authors selected linear beam elements BEAM4 in ANSYS to model the sophisticated structure of braided sleeve strands. Characterised by its

three-dimensional linear elastic properties, BEAM4 features 2 nodes and 6 DOFs at each node. The FEA model, focused on a singular pneumatic muscle integrated with a braided sleeve reinforcement, took a streamlined approach by excluding end caps, thus reducing the model's element count. It's noteworthy that the FEA model assumed negligible friction between the reinforcement and the pneumatic muscle, as well as among the strands themselves.

In a similar style, [228] adopted quadratic beam elements B32 within Abaqus to simulate the behaviour of braided sleeve strands. With each of its 3 nodes boasting 4 DOFs, the B32 element type facilitated a more sophisticated representation. The authors introduced an innovative design featuring a pneumatic bending actuator, where the reinforcement was composed of two halves of braided sleeves, each possessing a distinct initial braid angle to induce specific bending movements. The FEA model consisted of three distinct sets corresponding to the pneumatic muscle, braided sleeve, and end caps.

To construct threads of strands, an array of points was geometrically defined by a helical curve along the outer surface of the pneumatic muscle. Similar to the approach in [229], tie constraints were then strategically implemented to ensure a robust connection between the strand braiding, pneumatic muscle, and caps, ensuring a seamless and cohesive assembly with no sliding or separation between these interconnected components. A final layer of rubber tubing was affixed to the outer surface of the braided sleeve's reinforcement layer, adding an extra complexity to the FEA model while mirroring practical aspects of soft actuator designs.

An alternative technique was employed in [229], where the modelling of braided sleeve strands utilised LINK180 truss elements within ANSYS. These truss elements were precisely coincident with the nodes aligned along each edge of the external surface of the pneumatic muscle elements. The study focused on analysing the performance of a single-muscle contractile actuator, leading to an FEA model consisting of three distinct sets corresponding to the pneumatic muscle, braided sleeve, and the end caps.

In the formulation of this FEA model, it was presupposed that the braided sleeve and the pneumatic muscle maintained continuous connection without experiencing any friction between them. The three-dimensional simulation intentionally represented only one-eighth of the physical prototype by assuming symmetry on the plane perpendicular to the muscle axis, leveraging the axis-symmetric geometry of the muscle. This simplification not only adheres to the principles of symmetry but also significantly reduces the overall simulation time.

While the aforementioned studies often simplify their FEA models by presupposing a bond or tie between the braided sleeve and the pneumatic muscle, neglecting contact and the subsequent friction effects, [221] takes a more detailed approach. This study meticulously considers the sophisticated contact effect between the individual braided sleeve strands, specifically addressing thread-to-thread interactions. The modelling employed linear beam elements B31 in Abaqus, strategically placing nodes in the transverse planes where contact occurs. Additionally, the nuanced model accounts for the unique configuration of the braided sleeve, acknowledging its compressed and axially folded state, resulting in the formation of a bellow sheath.

Likewise, [27] delves into the complexities of contact and friction effects in modelling braided sleeve strands, with a primary focus on studying axial/radial deformation and changes in the braid angle. The study commenced by generating a detailed geometrical model for a braided sleeve, featuring a length of 85mm and a diameter of 25mm, using a Python script. Subsequently, the authors simulated this geometrical model in FEA employing 8400 beam elements (specifically, element type-1 Hughes–Liu beam) within LS-Dyna.

The braided sleeve was conceptually divided into two parts in the FEA model, distinguishing between clockwise and anti-clockwise strands. Both parts were modelled with an isotropic elastic material. Notably, the FEA mesh assigned no shared nodes between the two parts, and an automatic general beam-to-beam contact was applied to the model, introducing a static friction coefficient of 0.15. Despite demonstrating a favourable correspondence between simulation and experimental results, the simulation time was intentionally kept shorter than the actual physical experiments to reduce total computation time. Subsequently, the number of strands was later reduced as an additional measure to further minimise the overall computation time.

In summary, various FEA techniques were employed to model braided sleeves, with beam and truss elements emerging as the preferred choices for simulating strands, whether accounting for contact effects or not. However, this preference introduces a considerable challenge: the need to incorporate thousands of elements into the FEA model, elevating its complexity. The complexity arises from the required high mesh density, important for minimising discretisation errors in representing the curved strands, and the management of numerous contact points, if applicable, within the FEA simulation [27, 221]. These factors collectively contribute to a significant increase in the simulation time.

This challenge underscores a notable gap in the current literature. The absence of a simplified FEA model with a minimal number of elements for braided sleeves, and without contact interactions, presents an opportunity for advancement. Addressing this gap is a key contribution of this thesis, as it offers a streamlined simulation process and has the potential to establish a more practicable and efficient methodology for the design and optimisation of SPAs.

2.3 EGaIn-Based Soft Strain Sensors: An Overview of their Modelling Techniques

In the realm of sensing technologies, a sensor functions as a transducer, adept at converting various physical variables, such as mass, length, temperature, and others, into distinct signal variables such as mechanical or electrical outputs [230]. Within mechanical systems, force or displacement commonly serve as signal variables, often observed on instruments like dial scales, while in electrical systems, voltage emerges as the prevailing signal variable [230].

The classification of sensors into passive or active categories hinges upon the requisite energy source for the measurement process. Passive sensors, represented by devices like pressure gauges or thermocouples, operate independently, without any reliance

on external power or energy sources for the execution of measurements. Conversely, active sensors, exemplified by instruments like laser range finders or RADAR, depend on an external power or energy source for their operational capabilities [231]. This categorisation highlights the diverse nature of sensors and their reliance on energy considerations, offering a subtle perspective on their design and functionality within the field of measurement technologies.

To illustrate, the strain gauge serves as a fundamental example of a passive sensor, distinguished by its direct correlation between resistance and changes in measured strain. This unique characteristic highlights that the gauge's resistance is solely influenced by variations in strain, making it a singular source of energy variation. Despite this dependence, an auxiliary power supply is necessary to measure the resulting changes in resistance through a Wheatstone bridge [232].

When exploring the field of robotic systems, characterised as electrical systems, the outputs of measurements often manifest as minute changes in voltage. These voltage variations, typically requiring amplification due to their small magnitude, are then subjected to digitisation through analogue-to-digital converters (ADC). The transformed digital data is subsequently processed by microprocessors, forming a crucial part of the sophisticated feedback systems integral to the functionality of robotic platforms [131].

In the specialised field of soft robotics, sensors take on a unique role, necessitating designs that are soft, compliant, and sometimes stretchable. These characteristics are vital to ensure the seamless integration of sensors without impeding the inherent flexibility and movement of soft robotic systems. Achieving this balance often involves the incorporation of materials possessing intrinsic conductivity or producing sensors that exhibit flexibility and stretchability [233, 234]. Further exploration of soft sensors is elaborated upon in the following section.

2.3.1 Soft Sensing Technologies in a Nutshell

The conventional design paradigm for soft robots typically involves segregating actuation and sensing components. Soft sensors play a crucial role in this framework, serving to measure a spectrum of physical quantities, including but not limited to strain, force, and temperature [4]. While this modular approach is widely adopted, there is a growing body of literature introducing self-sensing actuators. These innovative components, as documented in various source [235–237], offer potential advantages by streamlining the manufacturing process. Nevertheless, it is critical to acknowledge that this integration could potentially compromise repair capabilities, especially in the event that the actuator or any of the sensors malfunction in the intended manner.

Soft sensors, designed specifically for integration into soft robotic systems, must possess certain characteristics to ensure optimal performance. They need to exhibit sufficient compliance to avoid impeding the movement of the soft actuator, resilience and stretchability to endure repeated motion cycles, and should lack any features that might induce damage or failure due to stress concentration [4]. As previously mentioned in Chapter 1, Section 1.2.3, soft sensors assume a pivotal role in proprioceptive sensing. This capability provides valuable feedback to closed-loop con-

trollers, enhancing the manipulation, locomotion, and balance of soft actuators [35]. The intricate interplay between soft sensors and the closed-loop control systems underscores their significance in advancing the capabilities of soft robotic platforms.

Soft sensors, built upon diverse sensing principles, offer a versatile range of output signal variables that respond to changes in resistance [40, 41], capacitance [42, 238], light reflectance [36, 37], or inductance [38, 39]. Each sensing type introduces specific challenges in fabrication, integration into soft actuators, as well as the subsequent amplification and measurement processes [234]. Moreover, each type boasts its own strengths and weaknesses, rendering it suitable for distinct scenarios [35].

Resistance-based sensors are widely recognised as the most common among the different types due to their simplicity, cost-effectiveness, and ease of fabrication [5]. These sensors operate by altering their resistance when subjected to deformation, enabling straightforward measurement. Diverse conductive materials, including EGaIn [41], ionic liquids [239], carbon nanotubes (CNT) [40], and others [240], were employed in the fabrication of these resistance-based sensors.

These sensors find application in a variety of scenarios, such as measuring curvature [40, 241], strain [242, 243], twist [139, 140], and force [244, 245]. However, resistance-based sensors may exhibit noticeable hysteresis due to the viscoelastic behaviour of the elastomeric material that they are usually constructed from. This hysteresis, influenced by the material's capacity to store and release energy over time, can limit their utilisation in dynamic scenarios where real-time feedback is essential [246]. Understanding these aspects assists in selecting the optimal soft sensor type for various applications, taking into account both its benefits and drawbacks.

Moreover, **piezoresistive sensors** stand out as a noteworthy subcategory within resistance-based soft sensing technology. Comprising elastomeric composite materials filled with conductive elements like nanoparticles, wires, or flakes, these sensors exhibit changes in their resistivity corresponding to any variations in the applied strain or pressure [247]. Researchers are drawn to using these sensors in applications such as soft skin [248] and superficial strain sensors [28] as shown in Figure 2.13(a), due to their simplicity of fabrication and the ability to tune mechanical and electrical characteristics. Despite these advantages, piezoresistive sensors face challenges such as significant hysteresis and non-linearity, necessitate extended recovery times, and exhibit relatively slow response rates [249].

While piezoresistive sensors fall under the broader category of resistance-based sensing, their working principle differs from other resistance-based sensors such as liquid metal or CNT-based sensors. Specifically, piezoresistive sensors exhibit resistance changes due to strain-induced variations in resistivity at the material level (i.e., the piezoresistive effect), whereas other resistance-based sensors, like liquid metal sensors, rely predominantly on geometrical changes in conductive pathways. This distinction is crucial in understanding their respective advantages, limitations, and applications in soft sensing [250, 251].

Capacitance-based sensors are gaining increasing popularity owing to their mechanical flexibility, high sensitivity, and ease of miniaturisation [253, 254]. These sensors can be tailored to diverse structural designs, enabling the conversion of physical inputs, such as strain, pressure, and proximity, into changes in capacitance [246].

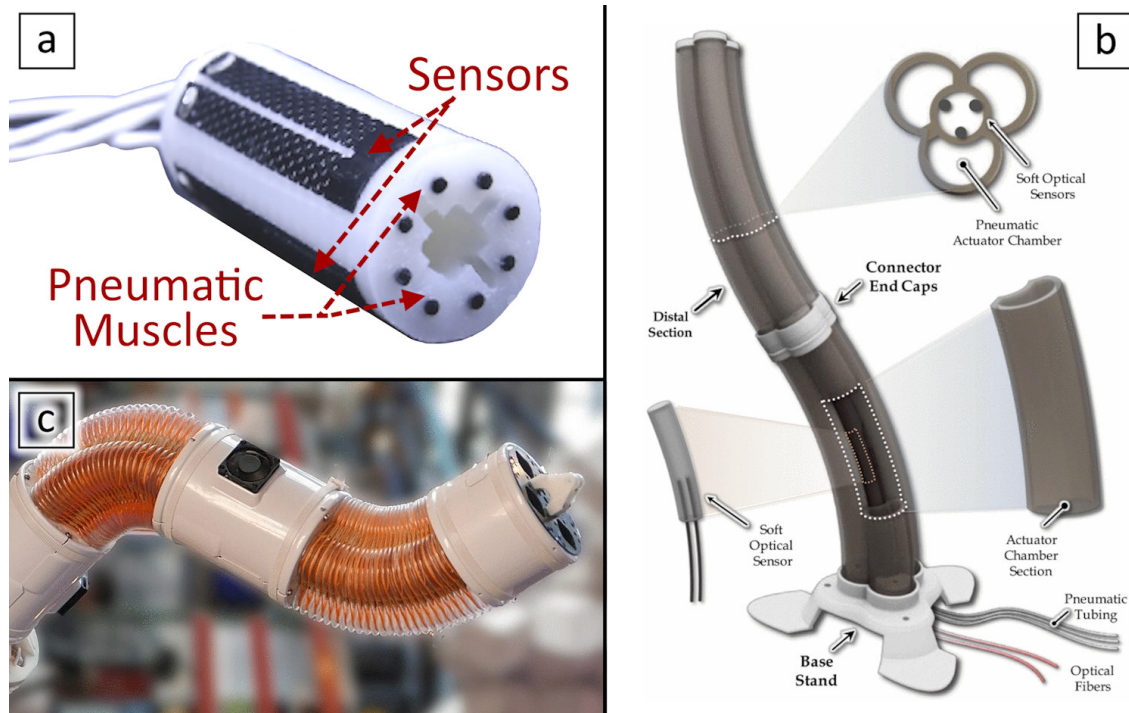


Figure 2.13: Examples of proprioceptive soft sensing where (a) shows a multi-muscle soft segment fitted with four superficial piezoresistive strain sensors with kirigami cut patterns. Adapted from [28] and reproduced under the terms of CC BY 4.0 license, (b) shows an SPA fitted with multiple optical diffraction sensors at the centreline. Adapted from [252]. Copyright 2018, IEEE, and (c) shows an SPA fitted with inductance-based sensors around its pneumatic muscles. Adapted from [39]. Copyright 2018, Springer Science Business Media, LLC, part of Springer Nature.

The fundamental design concept involves creating a sandwich structure comprising two parallel electrodes separated by a central core layer [255].

A wide spectrum of materials was employed in the fabrication of these sensors, including conductive fabric [256], nanocomposites [257], and EGaIn [258], among other materials [35]. Researchers applied this type of sensors across a multitude of applications, involving force, pressure, temperature, proximity, curvature, and strain measurements [35, 255].

Although capacitance-based sensors include significant benefits such as a wide dynamic range, rapid response, exceptional linear range, and sensitivity, they do have certain drawbacks. These drawbacks encompass susceptibility to contaminants, sensitivity to mechanical perturbations, and potential proximity effects [234]. Understanding these trade-offs is essential for informed decision-making when selecting capacitance-based sensors for specific applications, ensuring that the benefits align with the requirements of the intended use while considering and mitigating potential challenges.

Light reflectance-based, or optical, sensors are rapidly advancing and providing high-resolution sensing capabilities [259]. These sensors detect variations in the intensity, frequency, or phase of light based on physical inputs such as strain or pressure [35]. Notably, they exhibit insensitivity to electromagnetic interference, can be

miniaturised, and demonstrate low hysteresis [246]. Various types of optical sensors were explored in the literature, including stretchable waveguides [259, 260], FBG (as discussed earlier in Section 2.2.3), optical tactile sensors [261], optical diffraction sensors [252] as illustrated in Figure 2.13(b), and polymer optical fibres [262], among other types [35, 246].

While optical sensors offer distinct advantages, they are not without their limitations, presenting challenges in the context of soft robotics. These challenges encompass high cost, complexity in signal interpretation, considerations related to reliability, repeatability, flexibility, and strain range [234, 263]. These restrictions highlight the need to carefully evaluate trade-offs and choose the best optical sensor type for soft robotics applications depending on their requirements and constraints.

Inductance-based sensors are drawing attention for their commendable stretchability, low hysteresis, and notable signal repeatability [264]. The fundamental design incorporates a conductive coil, where the inductance varies based on changes in its diameter and turns spacing [5]. These sensors find applications in diverse fields, ranging from wearable devices [265] to self-sensing pneumatic actuators [38] and bending measurement [39] as shown in Figure 2.13(c), among other applications [266].

While these sensors can boast high accuracy and miniaturisation capability, they do come with certain limitations. These include factors such as limited strain range, sensitivity to electromagnetic interference, and limited compliance [266, 267]. It's crucial to recognise that the limited compliance limitation can have a direct impact on the deformation of soft actuators [35]. Understanding and addressing these challenges are essential for effective integration and application of inductance-based sensors, particularly in contexts where compliance plays a pivotal role in ensuring optimal performance.

In addition to the previously mentioned sensing technologies, **magnetic strain and tactile sensors** represent alternative methodologies. These sensors entail the incorporation of a magnetic source within the soft actuator, alongside magnetic field sensors designed to detect variations in the magnetic field caused by the movement of the soft actuator [268]. Magnetic sensors offer numerous benefits, such as having a compact form factor, highly sensitive, affordable, and can be seamlessly integration into various systems. However, they are susceptible to external magnetic interference from the environment, as well as potential interactions with ferromagnetic objects [35].

Surveying the wide spectrum of soft sensing technologies reveals a landscape rich with diversity, each type characterised by its unique strengths and limitations. The critical task of selecting the optimal sensor type for a specific application necessitates a nuanced evaluation of these variations. Within the framework of this thesis, particular attention is directed towards **EGaIn**-based soft strain sensors. This deliberate choice is motivated by their widespread acclaim, inherent design simplicity, and cost-effectiveness in fabrication, as well as their high stretchability, remarkable sensitivity, and stable electrical properties, which make them well-suited for soft robotic applications.

2.3.2 Modelling of EGaIn-Based Soft Strain Sensors

EGaIn-based soft strain sensors represent a notable advancement in the field of soft sensing. These sensors are characterised by elastomeric substrates featuring sophisticated patterns of microchannels filled with a liquid metal, typically the EGaIn alloy. This innovative design enables the measurement of strain or pressure by monitoring the substrate's stretching behaviour [269, 270]. The fundamental principle underlying their functionality lies in the alteration of microchannel dimensions within the elastomeric substrate consequent to stretching induced by external forces.

As the substrate undergoes deformation, the length and cross-sectional area of these microchannels dynamically change, which in turn generate a corresponding variation in electrical resistance. This resistance modulation adheres to the principles established by Ohm's law, forming the basis for the relationship between strain and electrical response known as electrical resistance formula [271].

In the domain of soft robotics, EGaIn-based sensors acquired considerable attention owing to their inherent characteristics. Notably, these sensors exhibit remarkable sensitivity, stretchability, and stability, all of which contribute to their reliability and adaptability in diverse applications [272–274]. These sensors exhibit an exceptional capacity to endure strains exceeding 700%, based on the specific elastomeric material employed during fabrication [154, 269]. Such attributes render them highly versatile and adaptable for deployment across a spectrum of applications spanning biomedical, agricultural, and underwater domains [275]. Their robustness and flexibility make them indispensable tools in diverse scenarios necessitating precise strain monitoring and tactile feedback mechanisms.

An essential prerequisite in the application of EGaIn-based soft strain sensors is the accurate estimation of strain in relation to the measured resistance. This estimation hinges not only on the geometric characteristics of the microchannels but also on the physical properties and behaviour of the elastomeric material utilised in the sensor's fabrication [88, 269, 270].

Various studies endeavoured to establish analytical models by employing the electrical resistance formula, focusing on the microchannel geometry within each sensor design [241, 276–281]. These analytical models aim to delineate the resistance-strain relationship, encompassing both linear and non-linear forms. However, it is imperative to acknowledge that these analytical models might not comprehensively account for certain factors that can impact measurement accuracy. These factors include, but are not limited to, strain rate, hysteresis, manufacturing variability, and non-uniform strain distribution within the microchannels [88, 249, 280].

An alternative approach to analytical modelling involves characterising these EGaIn-based sensors through the generation of datasets correlating strain with resistance. This characterisation can be achieved typically through FEA [282, 283] as shown in Figure 2.14(a), or experimental testing [284, 285] as shown in Figure 2.14(b). Subsequently, mathematical models are trained using numerical methods, such as machine learning [285–287] and regression analysis [245, 284, 288, 289], leveraging the generated datasets correlating strain with resistance. These numerical methods are employed with the primary objective of enhancing the performance and reliability of EGaIn-based soft strain sensors. They seek to reduce error and uncertainty in

strain estimation, thereby augmenting the proprioceptive sensing capabilities and state control of soft robots.

FEA-based modelling employs numerical simulations to solve the governing equations describing the sensor deformation across varying strain rates and conditions. This computational approach offers comprehensive insights into stress distribution and changes in channel geometry within the sensors. Moreover, **FEA** facilitates the integration of crucial factors such as material properties, hysteresis effects, and boundary conditions [290]. By simulating diverse designs and behaviours of soft sensors under different testing scenarios, **FEA** holds the potential to streamline prototyping processes, thereby conserving valuable time and resources [5, 156].

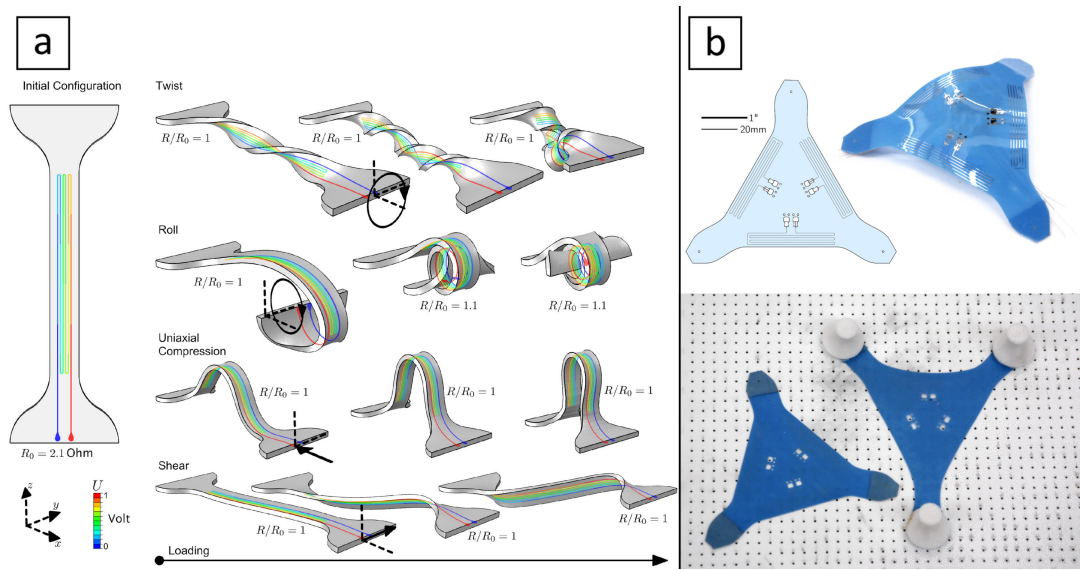


Figure 2.14: Examples of characterising **EGaIn**-based soft strain sensors using (a) **FEA** under various testing scenarios. Adapted from [282]. Copyright 2014 Elsevier Ltd. All rights reserved, and (b) experimental testing for a sensing module that has a triangular design (top) with three microchannel patterns at each side. The test bench (bottom) is a polystyrene pegboard with 3D printed pegs to hold each corner of the module at the required holes. Adapted from [284]. Copyright 2016, IEEE.

However, it is imperative to acknowledge that **FEA** may impose significant computational demands and necessitate specialised expertise in constructing 3D models and executing analyses [156, 234]. Furthermore, accommodating manufacturing variability within **FEA** simulations entails the adoption of specialised techniques, thereby introducing an additional layer of complexity [291]. Additionally, the fidelity of **FEA** models may depend on accurately defining the viscoelastic behaviour of elastomeric materials, which often necessitates experimental testing to derive material properties [292].

Conversely, experimental testing entails conducting physical tests on actual sensors, thereby yielding real-world data that can serve to validate theoretical models. This approach offers the advantage of capturing sophisticated phenomena that may pose challenges in theoretical modelling. Nonetheless, it is essential to acknowledge that experimental testing can entail significant time and expense, particularly when multiple iterations are necessary [154, 293, 294].

In essence, experimental testing not only furnishes empirical data but also serves a crucial role in model validation, complementing the theoretical predictions facilitated by [FEA](#). Together, these approaches play pivotal roles in characterising soft sensors by generating requisite datasets for training numerical models. Moreover, they can synergistically contribute to the design process, with [FEA](#) informing initial designs and subsequent validation through experimental testing [\[156\]](#).

2.3.3 Beyond the Data Points: Leveraging the Numerical Modelling

Numerical modelling plays a fundamental role in simulating and understanding the behaviour of complex systems, requiring the selection of appropriate equations and numerical methods to emulate the system under investigation [\[234\]](#). Integral to this process are the steps of verification, validation, and calibration of the modelling performance.

Verification serves to ensure the accurate implementation of the model within the computational framework [\[22\]](#), thereby confirming the reliability of the numerical solution. Validation, on the other hand, entails the rigorous comparison of the model's predictions against empirical data [\[28\]](#) or previously published findings [\[24\]](#), thereby assessing its fidelity in representing real-world phenomena.

Moreover, calibration emerges as a vital step aimed at refining the model's parameters to strengthen its alignment with empirical data. This fine-tuning process accounts for various factors such as manufacturing variability and measurement drifting attributed to polymer ageing [\[295\]](#), thereby enhancing the model's predictive accuracy and reliability.

As discussed previously, the datasets correlating strain with resistance obtained from either [FEA](#) or experimental testing serve as resources for training mathematical models using numerical techniques such as machine learning and regression analysis. Subsequently, the trained model must undergo validation to assess its accuracy and guard against overfitting [\[296\]](#), ensuring the reliability of predictions derived from the model.

The validation process typically involves dividing the available dataset into two subsets. one is designated for training the model, and the other is intended for validation. In the realm of machine learning, an additional step is often undertaken, wherein researchers further test the validated model using an unseen portion of the data after fine-tuning during the validation phase [\[297\]](#) as illustrated in Figure [2.15\(a\)](#). This step serves to ascertain the robustness and generalisability of the model beyond the training dataset, thus enhancing confidence in its predictive capabilities.

Furthermore, the concept of model complexity plays an essential role, defining the extent of independent parameters necessitated by a model to accurately predict the outcomes. A model characterised by a high degree of complexity possesses the capability to encapsulate intricate variations within the dataset. However, this heightened complexity introduces challenges throughout the training process and increases the likelihood of overfitting. The model gets heavily focused on the distinctive characteristics of the training data, potentially compromising its generalisability. Con-

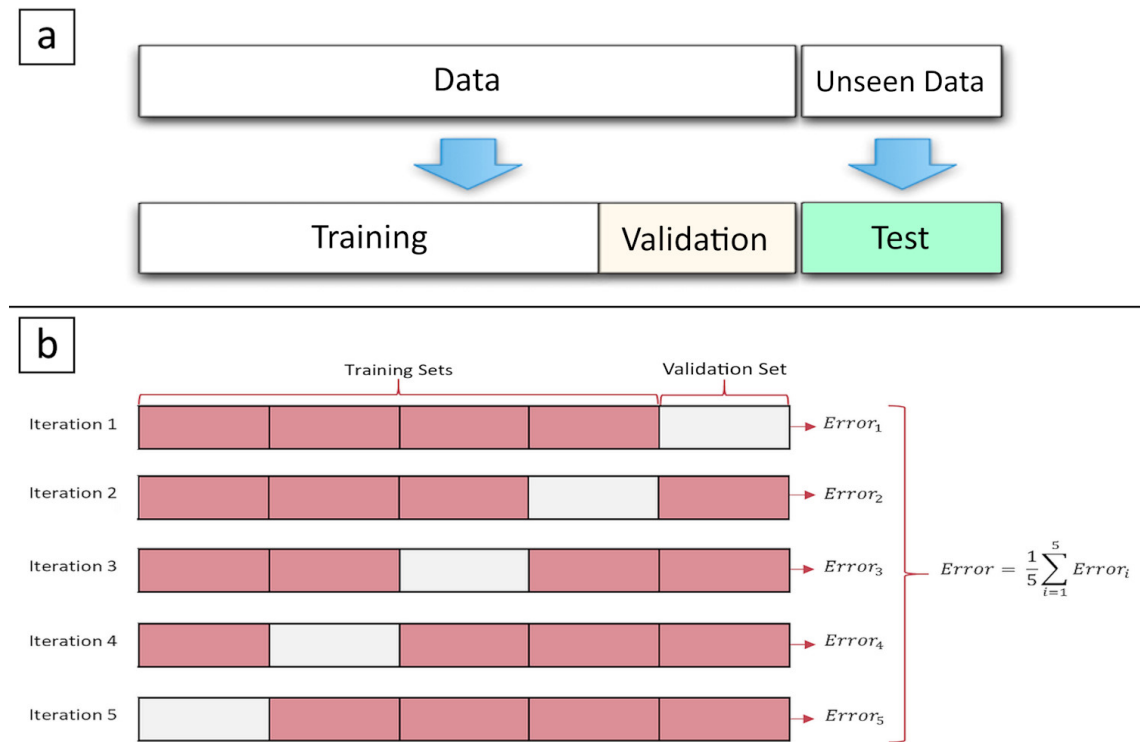


Figure 2.15: Partitioning datasets for the purpose of validating the numerical model accuracy where (a) illustrates the common strategy of partitioning the dataset into training, validation, and test subsets while (b) illustrates an example of the k-fold method with k=5.

versely, a model with a lower degree of complexity may offer simplicity in training, yet it runs the risk of inadequately capturing the full breadth of information present in the dataset, potentially leading to suboptimal predictive performance [296, 297] as illustrated in Figure 2.16.

Attaining the delicate balance between model complexity and predictive power stands as a cornerstone in the pursuit of successful numerical modelling. Several factors contribute to the heightened susceptibility of a model to overfitting, including a high number of tuneable parameters, a broad value range of these parameters, and inadequate training datasets [296, 297].

Various strategies exist for partitioning datasets, determined by factors such as dataset size and characteristics [296]. As an example, a common strategy involves assigning two-thirds of the datasets for training the model and reserving the remaining third for validation. Another common strategy entails utilising 80% of the datasets for training and reserving the remaining 20% for validation.

Additionally, researchers may lean towards cross-validation techniques, such as the k-fold method, which offers a robust approach to model validation [296]. In the k-fold method, the available dataset is subdivided into a total of 'k' subsets, using a value for 'k' that is chosen appropriately. Then, the model iteratively trained using 'k-1' subsets and validated on the remaining subset. This process is repeated 'k' times, ensuring that each subset serves as the validation data exactly once [296] as illustrated in Figure 2.15(b). Incorporating these diverse dataset splitting practices

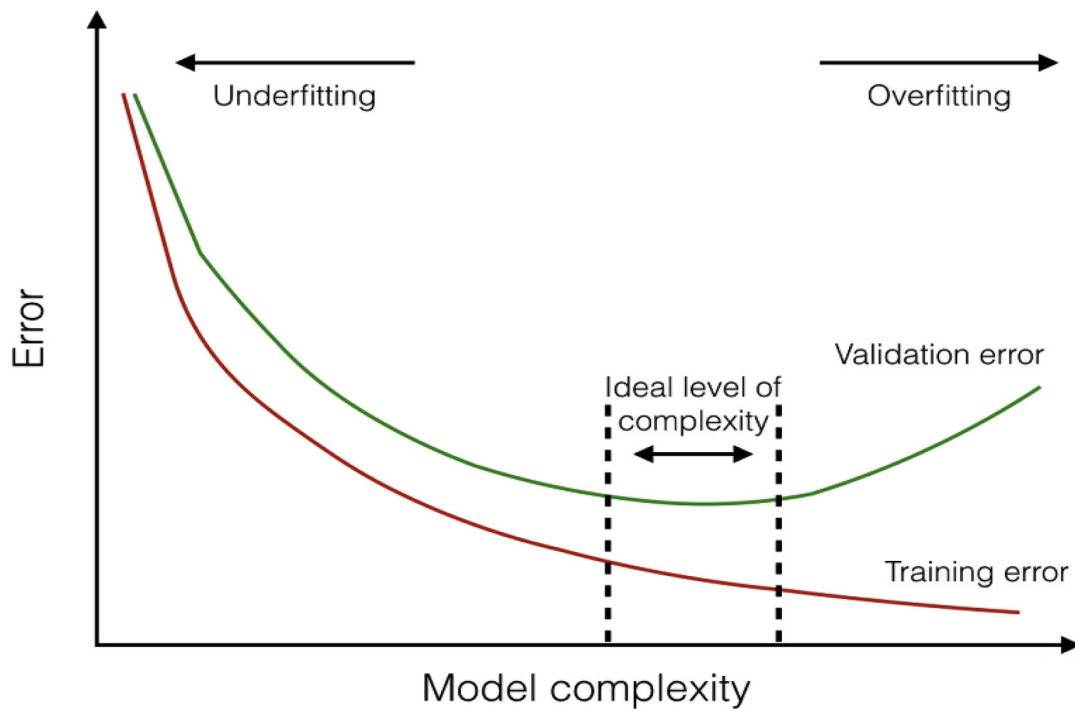


Figure 2.16: The effect of model complexity on its ability to capture the full breadth of information present in the dataset.

ensures robust model validation, laying the foundation for effective machine learning and regression analysis in data-driven research endeavours.

Machine learning models, such as artificial neural networks, support vector machines, or random forests, are rooted in data-driven algorithms aimed at uncovering underlying patterns by iteratively optimising algorithm parameters [142, 286]. Machine learning techniques harness the power of these algorithms to discern complex relationships between sensor resistance and strain, without relying on explicit mathematical models. By processing large volumes of data, machine learning algorithms excel at capturing complex, nonlinear patterns inherent in the data, while also accommodating additional variables such as temperature, pressure, hysteresis, and manufacturing variations.

Nevertheless, despite their inherent flexibility and robustness in comparison to regression analysis, the execution of machine learning algorithms presents notable difficulties. These algorithms demand substantial computational resources and extensive data to effectively train the models. Acquiring and curating such datasets can prove to be a costly and time-consuming endeavour, further highlighting the practical challenges associated with employing machine learning techniques [142, 286].

Given this, hybrid proprioceptive sensing methods that integrate machine learning with physics-based models provide numerous advantages. These methods improve sensing capabilities by delivering more precise and consistent measurements, even in dynamic and complex settings. By leveraging physics-based models, hybrid approaches reduce the amount of data required for training while enhancing accuracy

and robustness. They also optimise sensor distribution by strategically placing sensors to maximise coverage and efficiency. This blend of techniques enables real-time adaptation to changing conditions, boosting the overall performance and resilience of the robotic system. Moreover, the incorporation of machine learning allows the system to learn and evolve over time, resulting in continuous improvements in accuracy and responsiveness. These hybrid methods pave the way for more advanced and versatile soft robots that can be utilised in various applications [41, 88, 92, 286, 298, 299].

Conversely, regression analysis involves a range of statistical techniques tailored to model the relationships between a dependent variable and one or more independent variables [300]. Various regression models, including linear, logistic, and non-linear regression, among others, are characterised by predefined mathematical functions that aim to best fit the data by adjusting model parameters (or coefficients) [300]. This adjustment process is guided by the objective of minimising fitting errors, which can be quantified using metrics such as root mean squared error (RMSE) or other appropriate techniques [301].

Regression-based modelling offers several advantages, rooted in its simplicity and adherence to fundamental statistical principles such as correlation and least-square error [301]. Compared to machine learning approaches, regression models boast enhanced interpretability, leveraging predefined mathematical functions and requiring less data for training and validation. Moreover, they afford researchers the flexibility to manually select independent variables, further enhancing interpretability and control over the modelling process [300].

However, regression models do possess certain drawbacks. The manual selection of an appropriate model is imperative, as regression techniques do not automatically address non-linearity in the data. Moreover, the reliability of regression models diminishes with increasing complexity, increasing the risk of overfitting, and in the presence of lower data quality stemming from redundancy, outliers, or imbalanced data distributions [296, 302].

When it comes to modelling EGaIn-based soft strain sensors, both regression analysis and machine learning offer valuable approaches. Regression analysis serves as a solid starting point, appreciated for its simplicity and ease of interpretation. However, machine learning presents a compelling alternative, leveraging its capability to handle complex relationships and adapt to new data. This can result in more accurate models, particularly in scenarios where large datasets are available [286].

A good illustrative example would be the application of machine learning to proprioceptive sensing. Here, instead of individually modelling each strain sensor, machine learning enables the collective modelling of resistance measurements from multiple sensors against the deformations of the soft actuator [28, 41]. This approach allows for an overall understanding of the interaction between the sensors and the actuator, potentially enhancing the accuracy and depth of predictive models.

2.3.4 Insights on the Challenges in Achieving High Modelling Accuracy

As previously outlined in Section 2.3.2, **EGaIn**-based soft strain sensors exhibit remarkable sensitivity, making them valuable for various applications. However, achieving high modelling accuracy, particularly at sub-millimetre levels, necessitates careful consideration of several crucial factors.

Firstly, sensor design plays a pivotal role in minimising non-uniform stretching within the microchannels, which can significantly impact the accuracy of strain measurements [242]. Moreover, meticulous characterisation processes are essential to ensure precise and reliable data acquisition, contributing to the fidelity of the modelling outcomes [284]. Additionally, accurate modelling of viscoelastic behaviour is imperative, as **EGaIn**-based soft sensors often exhibit time-dependent mechanical responses that must be appropriately accounted for in the modelling framework [285].

Various experimental techniques were employed to characterise **EGaIn**-based soft strain sensors in recent studies. For instance, in the investigation conducted by [242], an optical motion capture system with a spatial accuracy of 0.44 mm was utilised in order to quantify hysteresis in measurements. Additionally, [303] employed a motorised testing stand with a remarkable linear accuracy of 0.05 mm to analyse resistance versus strain data. Their analysis of the data unveiled a coefficient of determination (R^2) exceeding 0.99 which indicates an excellent goodness of fit.

Furthermore, [304] demonstrated a successful integration of **EGaIn**-based soft strain sensors into a soft gripper. Their characterisation process involved calculating strain from a series of images captured during gripper actuation. Resistance measurements were plotted within 90% confidence intervals to account for noticeable error margins. Moreover, [305] employed a recurrent neural network to characterise the pressure response of microfluidic soft sensors known for their high hysteresis.

Other studies were conducted to estimate strain based on resistance measurements and to analyse the measurement uncertainties associated with such sensors. For example, [284] introduced a novel multi-element strain gauge module, Figure 2.14(b), which not only provided enhanced strain measurements but also addressed the inherent uncertainties in the process. Their study reported a modest 6.08% error in strain measurement, which they meticulously accounted for. To mitigate uncertainties, they employed a sophisticated Monte Carlo approach to reconstruct the module's triangular shape, enabling a more precise estimation of strain. This innovative methodology allowed for a comprehensive understanding of the uncertainties involved in the strain measurement process.

Moreover, the same research team presented another groundbreaking development: a multi-mode strain and curvature sensors module [241]. This module offered a versatile approach to measuring both strain and curvature simultaneously, providing valuable insights into the deformation behaviour of soft materials. Their findings indicated errors of 7.00% and 8.75% in strain and curvature, respectively, when comparing experimental and reconstructed data. To further evaluate the reliability of their measurements, the team meticulously assessed the uncertainties associated with the experimental setup. Surprisingly, they found that electrical noise, often considered a significant factor in sensor performance, played an insignificant role

compared to the measured system noise. This finding highlighted the significance of accounting for different sources of uncertainty in evaluating the performance of soft sensors.

Despite the multitude of studies and the widespread use of these sensors [154, 234], mathematical models with higher accuracy for estimating strain have yet to be developed. The ideal model should be capable of achieving estimation errors at sub-millimetre scales. Such precision is crucial if proprioceptive sensing is to replace external perception in detecting SPA deformation. However, the required level of accuracy may vary depending on the specific application [242]. This gap in the current literature necessitates dedicated research efforts.

A comprehensive study is needed to advance current research by introducing a new, simplified model. This study will focus on exploring the model's accuracy limits and examining the effects of strain rate (or speed) on hysteresis. Such an endeavour could pave the way for understanding the feasibility of reconstructing soft actuator deformation using a network of sensors, without the reliance on machine learning models.

It is important to consider that the selection between regression analysis and machine learning depends on particular factors within the modelling task. These factors are mainly the data availability, computational resources, and the complexity of relationships within the datasets. In the case of this thesis, regression-based modelling is chosen as the preferred method. This decision originates from a number of different reasons.

Firstly, the task at hand involves modelling individual strain sensors, which inherently entails a simpler relationship between variables. In this scenario, there is a single independent variable—resistance measurement—and a single dependent variable—the corresponding strain. Furthermore, the simple nature of the relationship makes regression analysis ideal for the task, since it offers a straightforward framework for estimating the connection between these variables. Additionally, regression-based modelling is a suitable choice for this thesis since it has lesser computational requirement and performs effectively with smaller datasets. This makes it a more feasible option considering the available resources and data limitations.

In summary, the decision to implement regression-based modelling in this thesis is informed by the simplicity of the modelling task, the availability of data, and considerations of computational efficiency. By leveraging regression analysis, this thesis aims to provide accurate and interpretable models for understanding the relationship between resistance measurements and corresponding strain in the EGaIn-based soft strain sensors.

Chapter 3

Inverse Kinematics: Linear Versus Quadratic Approximation

In this chapter, various aspects related to the [IK](#) of [SPAs](#) are carefully examined, specifically focusing on convergence rates, singularity avoidance, and physical limits. As introduced in Chapter 2, Section 2.1, the [IK](#) optimisation problem can be addressed using a linear or quadratic Taylor series expansion, among other techniques. This chapter focuses on the differences between linear and quadratic approximations in exploring these aspects, aiming to identify any ambiguous outcomes that may arise.

Throughout this chapter, the [PCC](#) model is revisited to provide detailed illustrations of certain attributes such as the singularity problem, discontinuity in the rotation angle, and redundancy. Solutions to these issues are detailed and examined within this chapter.

Furthermore, an in-depth formulation of the [IK](#) optimisation problem is presented for both linear and quadratic approximations. The impact of employing convex versus non-convex holonomic constraints is examined in relation to the optimisation Jacobian and Hessian matrices. A detailed mathematical background on actuator manipulability is also provided, which will be used later for evaluating the performance of the [IK](#) solver.

Subsequently, a thorough derivation of both the actuator Jacobian matrix and Hessian array is presented, accompanied by a mathematical overview of various Jacobian algorithms reported in the literature. Additionally, several damping profiles are proposed to enhance the performance of the [IK](#) solver by mitigating large motions or oscillations near singularities.

Subsequently, for the linear approximation, the [IK](#) solver performance is thoroughly analysed through various combinations of Jacobian algorithms and damping profiles, applied to four different paths with varying complexities, ranging from simple to highly complex. The comparison is then expanded to include the quadratic approximation, focusing on differences in convergence rates, performance near singularities, and the influence of physical limits on convergence between these two approximations.

3.1 PCC-Based Soft Actuator Modelling

Generally, the **PCC** soft actuator can be considered as a series of curved s -th segments, as illustrated in Figure 3.1(a), where $s = 1, \dots, \mathfrak{s}$. Each s -th segment has a set of arc parameters that define its state at any given time, as shown in Figure 3.1(b). The actuator origin is located on a fixed frame $\{S_0\}$, and has $\mathfrak{s}$ reference frames $\{S_1\}, \dots, \{S_s\}$ positioned at the end of each segment.

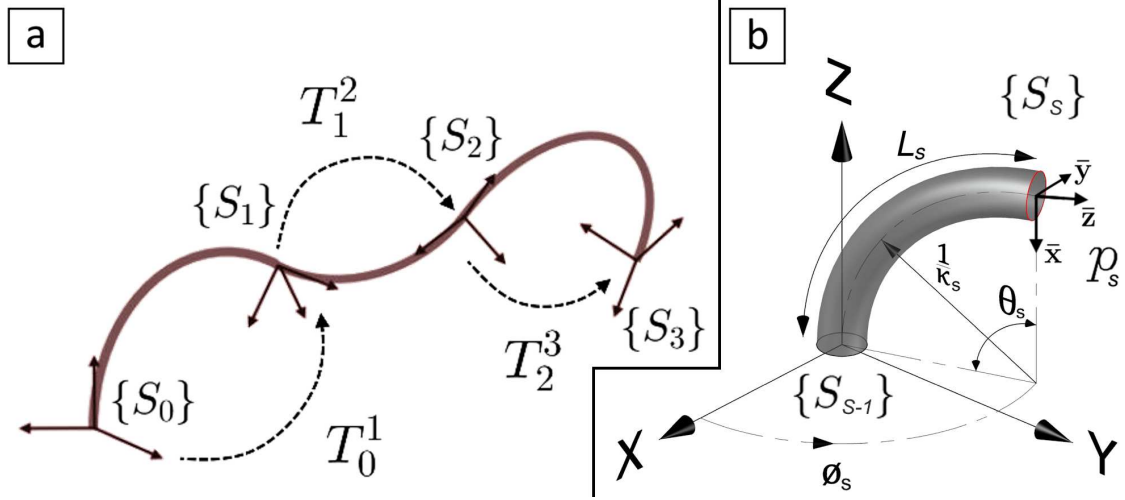


Figure 3.1: Arc parameterisation of a piecewise constant curvatures (**PCC**) soft actuator where (a) illustrates the transformation between the frames of a multi-segment soft actuator consisting of three segments, and (b) shows the arc parameterisation of a single s -th segment along with the orientation of its end point pose p_s . The s -th segment has a base frame $\{S_{s-1}\}$, and an end frame $\{S_s\}$.

3.1.1 Kinematic Mapping

The **PCC** approach breaks down the kinematics into two mappings [19], specific and independent, between three spaces that define the kinematics of the soft actuator, as shown in Figure 3.2. The state of the actuator at any given time is defined by specific variables within each space. The actuator space a encompasses the variables that describe the physical measurements of the actuators, such as the length of the pneumatic tubes. The configuration space q includes the variables that describe the arc parameterisation of the actuator. Finally, the task space p consists of the variables that describe the position and orientation of any point along the length of the actuator centreline.

The specific mapping f_{sp} transforms the actuation space a into the configuration space q , while the independent mapping f_{ind} , or f for simplicity, transforms the configuration space q into the task space p . This process is known as forward kinematics (**FK**). Conversely, **IK** involves the inverse transformation from the task space p to the configuration space q and then to the actuation space a . The derivation of the specific mapping depends on the unique design of each actuator, as each design influences the configuration space q differently. In contrast, the derivation of the independent mapping is applicable to all actuators that can be approximated as **PCC** arcs [17]. The specific mapping is beyond the scope of this thesis; therefore, the remaining sections will focus solely on the independent mapping f .

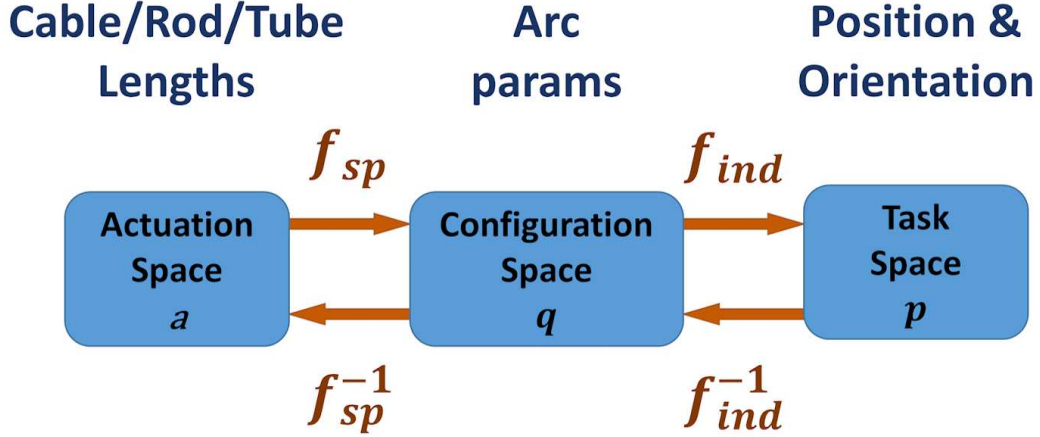


Figure 3.2: Spaces of PCC model and their mappings.

3.1.2 PCC Arc Parameterisation

The configuration space q comprises three variables ϕ , κ , and L , which represent the rotation angle of the plane containing the arc, the arc curvature, and the arc length, respectively [17], as illustrated in Figure 3.1(b). Alternatively, the arc can be parameterised using the bending angle $\theta = \kappa L$ to better visualise the bending limits. The configuration space q_s for each segment is defined as:

$$q_s = [\phi_s \quad \theta_s \quad L_s]^T \in \mathbb{R}^n, \quad s = 1, 2, \dots, \mathfrak{s}, \quad (3.1)$$

where $n = 3$ is the number of configuration space variables per segment, and $\mathfrak{s}$ denotes the total number of segments in the actuator. Consequently, $q \in \mathbb{R}^{n\mathfrak{s}}$ represents the configuration space of the soft actuator, which combines q_s for all segments. Each segment has a degree of freedom $DOF_k = n$, and the entire actuator has $DOF_k = n\mathfrak{s}$, as discussed further in Section 3.1.5 below.

The independent mapping $f(q) : \mathbb{R}^{n\mathfrak{s}} \rightarrow \mathbb{R}^m$ provides the required FK from the $n\mathfrak{s}$ configuration space coordinates to the m end point coordinates. This mapping is expressed in terms of $T_{s-1}^s \in SE(3)$, as shown in Figure 3.1(a), which represents the homogeneous transformation from $\{S_{s-1}\}$ to $\{S_s\}$ as follows:

$$T_{s-1}^s = \begin{bmatrix} R_{s-1}^s & \tilde{t}_{s-1}^s \\ [0 \quad 0 \quad 0] & 1 \end{bmatrix}, \quad (3.2)$$

where $R_{s-1}^s \in SO(3)$ represents the rotation matrix and $\tilde{t}_{s-1}^s \in \mathbb{R}^3$ denotes the translation vector. The rotation matrix R_{s-1}^s and the translation vector $\tilde{t}_{s-1}^s$ can be derived using various techniques, such as DH parameters or exponential coordinates, as mentioned earlier in Chapter 2, Section 2.1.2. The derivation can be expressed in terms of the arc parameters ϕ_s , θ_s , and L_s as follows [17]:

$$R_{s-1}^s = \begin{bmatrix} c_{\phi_s} c_{\theta_s} & -s_{\phi_s} & c_{\phi_s} s_{\theta_s} \\ s_{\phi_s} c_{\theta_s} & c_{\phi_s} & s_{\phi_s} s_{\theta_s} \\ -s_{\theta_s} & 0 & c_{\theta_s} \end{bmatrix}, \quad (3.3)$$

$$\tilde{t}_{s-1}^s = \frac{L_s}{\theta_s} \begin{bmatrix} c_{\phi_s}(1 - c_{\theta_s}) & s_{\phi_s}(1 - c_{\theta_s}) & s_{\theta_s} \end{bmatrix}^T,$$

in which c_{ϕ_s} , s_{ϕ_s} , c_{θ_s} , and s_{θ_s} represent $\cos(\phi_s)$, $\sin(\phi_s)$, $\cos(\theta_s)$, and $\sin(\theta_s)$, respectively. The end point frame p_s in (3.3) is oriented such that its x -axis points towards the PCC arc centre, as shown in Figure 3.1(b).

Alternatively, the end point frame can be aligned with the base frame without rotating around the z -axis by multiplying (3.3) by $R_z(-\phi_s)$. The resulting rotation matrix is:

$$R_{s-1}^s = \begin{bmatrix} c_{\phi_s}^2(c_{\theta_s} - 1) + 1 & s_{\phi_s}c_{\phi_s}(c_{\theta_s} - 1) & c_{\phi_s}s_{\theta_s} \\ s_{\phi_s}c_{\phi_s}(c_{\theta_s} - 1) & s_{\phi_s}^2(c_{\theta_s} - 1) + 1 & s_{\phi_s}s_{\theta_s} \\ -c_{\phi_s}s_{\theta_s} & -s_{\phi_s}s_{\theta_s} & c_{\theta_s} \end{bmatrix}. \quad (3.4)$$

Finally, the overall homogeneous transformation that maps a multi-segment soft actuator from $\{S_0\}$ to $\{S_s\}$ is:

$$T_a = \begin{bmatrix} R_a & \tilde{t}_a \\ \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} & 1 \end{bmatrix} = \prod_{s=1}^s T_{s-1}^s. \quad (3.5)$$

where $R_a \in SO(3)$ denotes the total rotation matrix and $\tilde{t}_a \in \mathbb{R}^3$ represents the total translation vector. Eq. (3.2) has two main issues [21]. The first issue is the singularity that occurs when the bending angle is zero, $\theta_s = 0$, which corresponds to the actuator's straight configuration. The second issue involves rotation discontinuity at $\phi_s = \pm\pi$. Various alternative solutions were proposed to address these problems, though they increase the complexity of the kinematic modelling, as mentioned earlier in Chapter 2, Section 2.1.3.

Alternatively, a simpler approach is proposed by suggesting a minimum bending limit to prevent singularity in the straight configuration, as detailed in the next section. Moreover, a recommended practice is to apply a range of $[0, 2\pi]$ for both the rotation angle ϕ_s and the bending angle θ_s to help mitigate the discontinuity.

3.1.3 Minimum bending limit

Since the singularity occurs when the actuator is at the straight position, determining a suitable minimum bending angle θ_s that is slightly greater than zero can achieve the required IK solver accuracy without introducing singularity. Given that

each segment can bend independently of the others within the actuator, each segment can have its own minimum bending angle. For simplicity, all segments in the actuator are assumed to share the same minimum bending angle θ_{min} .

Let d_s be a spatial accuracy limit for the PCC segment end point, as illustrated in Figure 3.3(a). This limit represents the allowable error in the horizontal distance between the end point of the PCC segment and the z -axis. This accuracy limit remains constant for all values of the rotation angle ϕ_s . The trigonometric relationship between the bending angle θ_s , the arc radius r_s , and the spatial accuracy limit d_s is illustrated in Figure 3.3(b) and is defined as:

$$\tan\left(\frac{\pi - \theta_s}{2}\right) = \frac{r_s \sin \theta_s}{d_s}, \quad (3.6)$$

then substituting $r_s = L_s/\theta_s$ into (3.6) and simplifying yields:

$$\frac{1 - \cos \theta_s}{\theta_s} = SAR, \quad (3.7)$$

where $SAR = d_s/L_s$ denotes the spatial accuracy per unit length, referred to as the spatial accuracy ratio. Solving (3.7) numerically for any given SAR provides the minimum bending angle θ_{min} necessary to meet the required spatial accuracy. Numerical solutions for (3.7) were obtained over the SAR range of $[10^{-10}, 10^{-2}]$, as shown in Figure 3.3(c).

The results reveal a linear relationship where $\theta_{min} = 2 * SAR$ within the solution range. Through trial and error, it was determined that $SAR = 10^{-7}$ is the smallest value that does not lead to convergence issues in the IK solver. However, an SAR value of 0.01%, i.e., $SAR = 10^{-4}$, is recommended, corresponding to a spatial error of 0.1 mm per 1 m length [306]. Smaller SAR values necessitate additional iterations for minimal gains in spatial accuracy.

3.1.4 End point pose

The end point pose $p_s = f(q_s)$ of each segment, illustrated in Figure 3.1(b), comprises its position and orientation in the 3D space. Generalising this for a multi-segment actuator, the end point pose $p_a = f(q)$ of the actuator consists of its position and orientation in the 3D space as follows:

$$p_a = \begin{bmatrix} p_c \\ p_r \end{bmatrix} = f(q) = \begin{bmatrix} f_c(q) \\ f_r(q) \end{bmatrix} \in \mathbb{R}^6, \quad (3.8)$$

where $p_c = [p_x \ p_y \ p_z]^T \in \mathbb{R}^3$ is the position vector that represents the Cartesian coordinates of the end point pose p_a , and $p_r = [\alpha_x \ \alpha_y \ \alpha_z]^T \in \mathbb{R}^3$ is the orientation vector that represents the rotation angles of the end point pose p_a about the x , y , and z axes. Similarly, $f_c(q) \in \mathbb{R}^3$ represents the linear independent mapping, and $f_r(q) \in \mathbb{R}^3$ represents the rotational independent mapping.

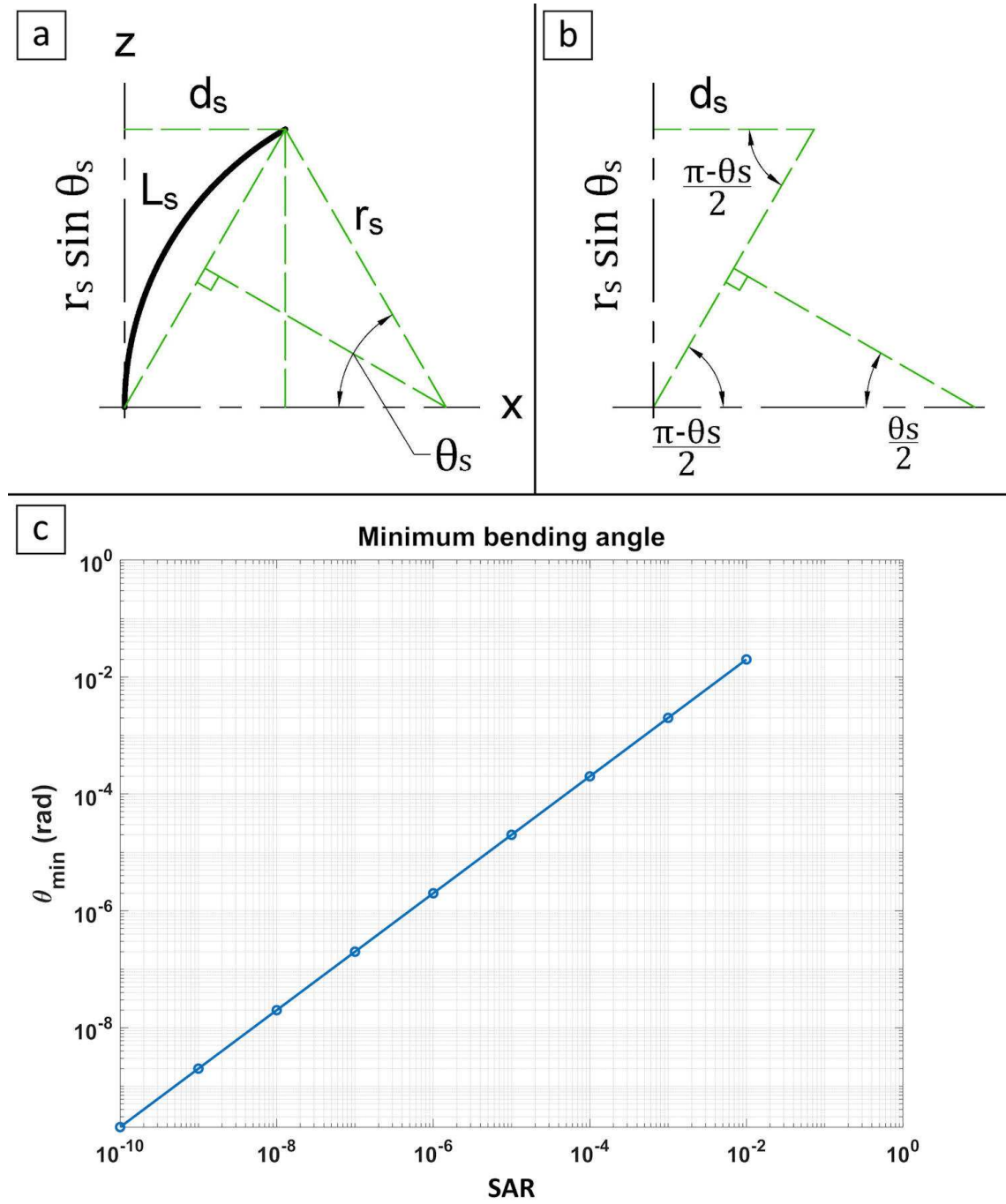


Figure 3.3: Minimum bending of a PCC segment: (a) displays the spatial accuracy limit d_s at the segment's end point, (b) depicts the trigonometric relationship between the bending angle θ_s , the arc radius r_s , and the spatial accuracy d_s , and (c) shows the numerical results for the minimum bending angle $\theta_{\min}$ versus the spatial accuracy ratio (SAR).

The base point pose is fixed and denoted by $p_o = [p_{co} \ p_{ro}]^T$ where $p_{co} = p_{ro} = 0_{3 \times 1}$, i.e., zero vectors. The end point pose p_a can be determined by calculating its two vectors p_c and p_r individually via the independent mappings $f_c(q)$ and $f_r(q)$, respectively.

Using the linear independent mapping $f_c(q)$, the position vector p_c is calculated as:

$$p_c = f_c(q) = \tilde{t}_a, \quad (3.9)$$

which is essentially the result derived from the translation vector $\tilde{t}_a$ in the total homogeneous transformation mapping T_a . Thus, the position vector p_c is determined as follows:

$$\begin{bmatrix} p_c \\ 1 \end{bmatrix} = T_a \begin{bmatrix} p_{co} \\ 1 \end{bmatrix}. \quad (3.10)$$

Conversely, the orientation vector p_r is obtained by converting the rotation matrix R_a into Euler rotation angles r_1 , r_2 , and r_3 corresponding to rotations about the Cartesian axes. This conversion necessitates selecting the appropriate rotation order to ensure accurate transformation of Euler angles into the end point orientation. It is assumed that the **PCC** segment is not exposed to external loads that induce twisting. An analysis of twisting effects is provided in Chapter 4.

Starting from the initial configuration of the **PCC** segment, which is straight, the rotation angle ϕ occurs about the z -axis, while the bending angle θ occurs about the y -axis. Any twisting would also occur around the z -axis. This rotation order is denoted as ZYZ . However, for cases involving only twisting, this order results in a singularity when $\theta_s < \theta_{min}$. Therefore, an alternative rotation order, XYZ , is used, which eliminates rotation around the z -axis, i.e., $\phi = 0$, since it is irrelevant in the initial configuration.

For a **PCC** segment with a bending angle $\theta_s \geq \theta_{min}$, i.e., using the ZYZ rotation order, the Euler angles can be computed using R_a as follows [307]:

$$\begin{aligned} r_1 &= \arctan \left(\frac{R_{a23}}{R_{a13}} \right), \\ r_2 &= \arctan \left(\frac{\sqrt{1 - R_{a33}^2}}{R_{a33}} \right), \\ r_3 &= \arctan \left(\frac{R_{a32}}{-R_{a31}} \right). \end{aligned} \quad (3.11)$$

Therefore, the rotational independent mapping is:

$$f_r(q) = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} - p_{ro} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix}. \quad (3.12)$$

Eq.(3.11) can be directly implemented in MATLAB using the `atan2` function. Alternatively, the `dcm2angle` function can be used, after inverting the rotation matrix R_a to convert the rotation from extrinsic to intrinsic. The 'robust' option should be added to address the issue of *gimbal lock* when $\theta = \pi$.

To examine the discontinuity problem, an arbitrary soft actuator consisting of a single PCC segment, $\mathfrak{s} = 1$, was selected. A rotation range of $\phi_s = [0, 2\frac{1}{2}\pi]$ and a bending range of $\theta_s = [\theta_{min}, 1\frac{1}{2}\pi]$ were applied simultaneously in the simulation. The end point pose p_a was calculated for the entire path. Figure 3.4(a) shows the end point position p_c in 3D space and its projections onto the XY , XZ , and YZ planes. For clarity, only the first and last PCC arcs are shown and highlighted in purple.

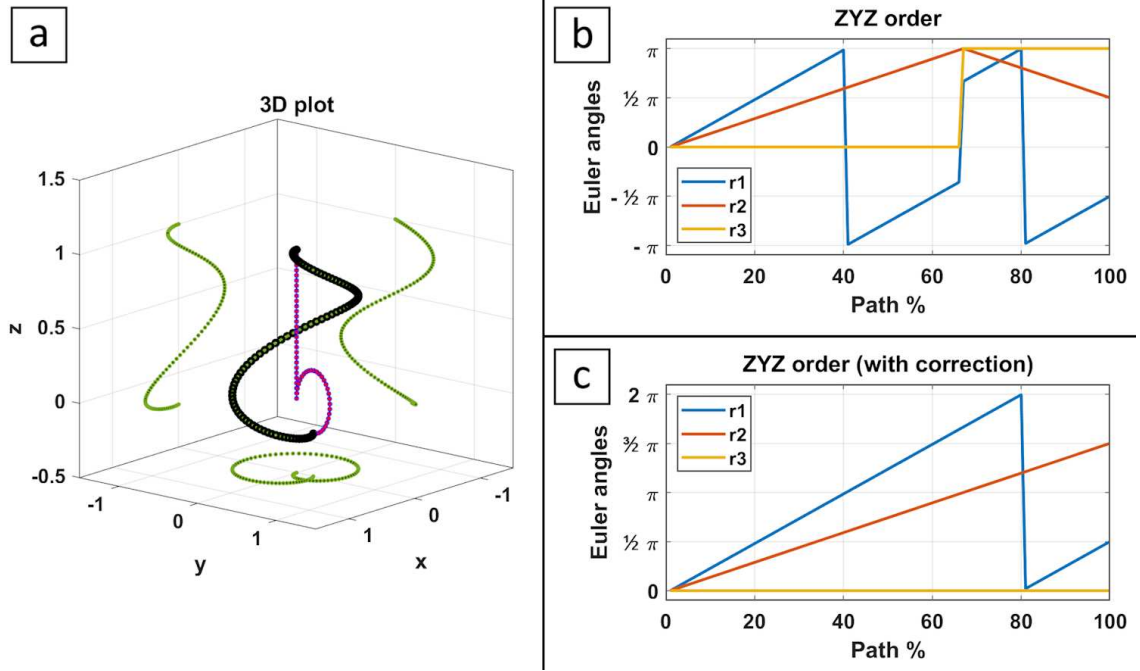


Figure 3.4: Forward kinematics of a PCC segment: (a) displays the 3D path of the end point and its projections on the XY , XZ , and YZ planes, (b) shows the Euler angles results for the ZYZ order before correction, and (c) shows the results after correction.

The results of (3.11) exhibited two issues: First, the Euler angles are in the range $[-\pi, \pi]$, which is inconsistent with the selected work range of $[0, 2\pi]$ introduced earlier in Section 3.1.2. Second, r_3 is supposed to be zero for the entire path since there is no twisting. However, $r_3 = \pi$ in part of the path, indicating a mirrored or flipped rotation in both r_1 and r_2 . This flip in r_3 occurs when the bending angle θ reaches π . Such discontinuity in Euler angles can lead to challenges in optimisation algorithms, as mentioned earlier in Chapter 2, Section 2.1.3.

A suggested solution for this discontinuity is to correct the angles based on certain ranges or values [308]. Based on the results in Figure 3.4(b), a two-step correction can be used to address these two issues as follows:

- Step 1) Correcting the flipping in r_3 :

$$r_1, r_2, r_3 = \begin{cases} r_1 = r_1 + \pi \\ r_2 = 2\pi - r_2 \\ r_3 = 0 \end{cases} \quad \text{if } r_3 = \pi, \quad (3.13)$$

$$\text{no correction,} \quad \text{if } r_3 = 0.$$

- Step 2) Correcting the negative sign in r_1 :

$$r_1 = \begin{cases} r_1 = r_1 + 2\pi & \text{if } \text{sign}(r_1) = -1, \\ \text{no correction,} & \text{if } \text{sign}(r_1) = +1. \end{cases} \quad (3.14)$$

A negative sign correction is only required for r_1 since r_2 depends solely on the applied bending angle θ which is always positive. Finally, the orientation vector p_r is calculated as follows:

$$p_r = \begin{bmatrix} \alpha_x \\ \alpha_y \\ \alpha_z \end{bmatrix} = f_r(q)_{corrected} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix}_{corrected}, \quad (3.15)$$

in which $\alpha_z = r_3 = 0$ at all times since there is no twisting. This is because rotating the actuator around the z -axis causes the top end face to rotate in the opposite direction, as the base is fixed [21]. Therefore, $\alpha_z = 0$ if there is no external or additional twisting.

3.1.5 Redundancy

The 3D space has 6 DOF_m , which includes three translational and three rotational DOF_m , as stated earlier in Chapter 2, Section 2.1.1. The number of DOF_k in the robotic actuator determines its mobility in the 3D space. The actuator is considered non-holonomic, holonomic, or redundant if it has DOF_k fewer than, equal to, or more than DOF_m , respectively [45]. Let us consider a rigid actuator consisting of several links and joints. The mobility formula, known as the Chebychev-Grübler-Kutzbach criterion [309], determines the total number of DOF_k in any kinematic chain as follows [43, 45]:

$$DOF_k = DOF_m(N_L - 1 - N_J) + \sum_{j=1}^{N_J} fr_j, \quad (3.16)$$

where N_L is the number of links including the base, N_J is the number of independent joints, and fr_j is the number of degrees of freedom per joint. For a serial mechanism

or an open kinematic chain, there is always one more link than the number of joints, and (3.16) simplifies to:

$$DOF_k = \sum_{j=1}^{N_J} f r_j . \quad (3.17)$$

The soft actuator is typically considered an open kinematic chain, although some designs implement closed kinematic chains [310, 311]. An open kinematic chain is fixed at one end to the base, with the other end free to move within 3D space.

For a PCC soft actuator, a single segment can be represented using DH parameters as a serial rigid actuator [17] with $N_L = 5$ and $N_J = 4$, generating the homogeneous transformation matrix in (3.3). One of the joints is dependent, and the remaining three joints each have a single degree of freedom, $f_s = 1$. Substituting these values into (3.17) gives $DOF_k = 3$. The same result is achieved when considering the serial actuator with $N_L = 6$ and $N_J = 5$, which generates the rotation matrix in (3.4), as it has two dependent joints.

In other words, a single PCC segment in 3D space is considered non-holonomic since it has only 3 DOF_k . The IK solver can calculate 3 out of the 6 DOF_m required for 3D space. For instance, it can determine the segment configuration space q_s to achieve a specific end point 3D position within the workspace, but not the end point orientation. By contrast, a soft actuator with two PCC segments is considered holonomic, and additional segments provide redundancy for enhanced manipulation and obstacle avoidance. However, redundancy can result in an infinite number of IK solutions and may necessitate defining criteria to achieve an optimal solution [43].

3.2 Inverse Kinematics

The IK problem can be simplified to finding one or more configuration space solutions q_d that satisfy a single desired pose p_d or a sequence of poses along a path. Let $g(q) : \mathbb{R}^{n_s} \rightarrow \mathbb{R}^m$ be a *holonomic constraints* function. The variable $m \in \mathbb{Z}^+$ represents an arbitrary vector size to be defined later.

The function $g(q)$ can be either a *convex* or *non-convex* function [97], depending on the selected constraints. The goal is to find an optimal solution q_d such that

$$g(q_d) = 0 , \quad (3.18)$$

by implementing Newton's method for root finding. Assuming that $g(q)$ is twice differentiable, the following optimisation problem is addressed [97]:

$$\min_{q \in \mathbb{R}^{n_s}} g(q) . \quad (3.19)$$

To solve the problem, a sequence $\{q^{\mathbb{K}}\}$ is constructed, starting with an initial guess $q_o \in \mathbb{R}^{n_s}$ that is "close to" the solution q_d . Iterations continue until a solution meeting the accuracy limit is found. For position, the spatial accuracy limit d_s is

used, while for orientation, a separate angular accuracy limit a_s can be applied. For simplicity, a_s is assumed to be equal to d_s mathematically.

In the case of a multi-segment actuator, the desired end point pose p_d is specified, and the inner poses of each segment, i.e., $p_{sd} \forall s \in [1, \mathfrak{s}-1]$, are typically determined using the [IK](#) algorithm. Special scenarios, such as obstacle avoidance in path planning [\[43, 312, 313\]](#), may require specifying inner poses as well, but these are beyond the scope of this thesis.

3.2.1 Newton's Method

Given the initial guess q_o , the kinematics can be expressed using the Taylor expansion [\[97, 113\]](#) of $g(q)$ around q as:

$$g(q_d) = g(q) + \nabla g(q)^T \Delta q + \frac{1}{2} \Delta q^T \nabla^2 g(q) \Delta q + h.o.t, \quad (3.20)$$

where $\nabla g(q)$ and $\nabla^2 g(q)$ represent the first- and second-order gradients of the function $g(q)$ used for optimisation, respectively. The linear approximation of [\(3.20\)](#) is obtained by truncating the Taylor expansion at the first order. Substituting into [\(3.18\)](#) yields the following iteration scheme:

$$q^{\mathbb{K}+1} = q^{\mathbb{K}} - J_g(q^{\mathbb{K}})^{-1} g(q^{\mathbb{K}}), \quad (3.21)$$

where $J_g(q) = \nabla g(q)^T \in \mathbb{R}^{m \times ns}$ is the optimisation Jacobian matrix of $g(q)$ for the multi-segment actuator evaluated at $q^{\mathbb{K}}$. The derivation of $\nabla g(q)$ and $J_g(q)$ is provided in [Section 3.2.2](#) below. However, this iteration scheme may encounter mathematical issues if the Jacobian $J_g(q)$ is not square, is not invertible, or exhibits singularity. Therefore, modified solutions are required to address such problems, as will be discussed in [Section 3.4](#).

Similarly, for the second-order or quadratic approximation of the Taylor expansion, linearising the optimality condition $\nabla g(q_d) = 0$ near q yields:

$$\nabla g(q + \Delta q) \approx \nabla g(q) + \nabla^2 g(q) \Delta q = 0, \quad (3.22)$$

which represents the first-order Taylor expansion of the gradient $\nabla g(q)$ around q and it is a linear equation in Δq . Solving for Δq leads to the following iteration scheme:

$$q^{\mathbb{K}+1} = q^{\mathbb{K}} - H_g(q^{\mathbb{K}})^{-1} \nabla g(q^{\mathbb{K}}), \quad (3.23)$$

where $H_g(q) = \nabla^2 g(q) = J_g(\nabla g(q))^T \in \mathbb{R}^{ns \times m \times ns}$ is the Hessian tensor of $g(q)$ for a multi-segment actuator evaluated at $q^{\mathbb{K}}$. The Hessian $H_g(q)$ is either a second- or third-rank tensor, depending on the value of m .

For instance, if $g(q) \in \mathbb{R}$, then $m = 1$ and the Hessian $H_g(q)$ reduces to a second-rank tensor, which is basically a square matrix, i.e., $H_g(q) \in \mathbb{R}^{ns \times ns}$. Conversely,

when $m > 1$, the Hessian $H_g(q)$ becomes a third-rank tensor, i.e., $H_g(q) \in \mathbb{R}^{n_5 \times m \times n_5}$. The derivation of $H_g(q)$ is detailed in Section 3.2.2 below.

Generally, the quadratic approximation converges more rapidly compared to the linear approximation [97]. However, it incurs the additional cost of calculating the Hessian $H_g(q)$. Moreover, the quadratic approximation can fail if the Hessian $H_g(q)$ cannot be inverted. In such cases, the solver might oscillate around the solution and not be able to converge or could converge to a saddle point rather than a local or global minimum. Further details are provided in Section 3.6.5 below. The type of holonomic constraints function $g(q)$ can also influence the convergence of Newton's method, as to be described in the next section.

3.2.2 Convex Versus Non-Convex Holonomic Constraints

The definition of convex functions is discussed thoroughly in [97] and interested readers can refer to the related reference for further details. Newton's method may face challenges when minimising non-convex functions because it does not differentiate between various types of critical points. At these points where the gradient is zero, $\nabla g(q) = 0$, the function might be a local maximum, local minimum, or saddle point.

The type of critical points can be identified by examining the eigenvalues of the optimisation Hessian $H_g(q)$. Thus, the applicability of Newton's method is restricted to cases where the Hessian $H_g(q)$ is positive definite and Lipschitz continuous [94]. Additionally, the solution might converge to a local optimum rather than finding the global optimum. This outcome can depend on the choice of the initial seed.

A key attribute of convex functions within certain domains is that every locally optimal point is also globally optimal and represents a minimum [97]. If $g(q)$ is a strongly convex function with a Lipschitz Hessian $H_g(q)$, the sequence generated by Newton's method will converge to a global minimum [94, 97].

The simplest constraints function $g(q)$ is the pose error [43], which is simply the mathematical difference between the current pose $p_a = f(q)$ and the desired pose p_d . This is defined as:

$$g(q) = p_d - f(q) \in \mathbb{R}^m, \quad (3.24)$$

where $m = 6$ when both the position and orientation error vectors are included. Similarly, $m = 3$ when either the position or orientation error vector is included. Other combinations are also possible, such as including the position error vector and a single component from the orientation error vector. In such cases, m should be set to 6, and the error should be evaluated only in the selected components.

The constraints function in (3.24) is non-convex (NCX) and leads to the following optimisation Jacobian, gradient, and Hessian arrays, based on the *numerator layout convention* [314, 315]:

$$J_g(q) = -J_f(q) \quad \in \mathbb{R}^{m \times ns}, \quad (3.25)$$

$$\nabla g(q) = -\nabla f(q) = -J_f(q)^T \quad \in \mathbb{R}^{ns \times m}, \quad (3.26)$$

$$H_g(q) = -H_f(q) \quad \in \mathbb{R}^{ns \times m \times ns}. \quad (3.27)$$

The derivations of the actuator Jacobian $J_f(q)$ and Hessian $H_f(q)$ are provided in Sections 3.3.2 and 3.3.3 below, respectively. However, by comparing (3.21) and (3.23), it is obvious that p_d is omitted from (3.23). This means that p_d does not have any effect on the quadratic approximation. Hence, for the constraints function $g(q)$ of (3.24), the IK solution is feasible via the linear approximation only.

Alternatively, for the position error, the following constraints function is suggested:

$$g(q) = \frac{1}{2} \|p_d - f(q)\|_2^2 \quad \in \mathbb{R}, \quad (3.28)$$

where $\|\cdot\|_2$ is the Euclidean norm, and $p_d, f(q) \in \mathbb{R}^3$. This function is known as the *least-squares approximation* and it is a convex (CVX) function [97, 124]. The constraints function (3.28) is suitable for position error only, i.e., $m = 3$ and $f(q) = f_c(q)$, since the position and orientation vectors of p_a have different units.

For orientation, a combination of NCX and CVX constraints functions can be used as an example. The position error is represented by (3.28), and the orientation error is represented by (3.24). The optimisation algorithm must satisfy both error conditions to achieve a solution. Such an optimisation scenario is outside the scope of this thesis.

By using the identity $\|X\|_2 = \sqrt{X^T X}$, the constraints function of (3.28) can be simplified to [124]:

$$g(q) = \frac{1}{2} (p_d^T p_d - 2 p_d^T f(q) + f(q)^T f(q)). \quad (3.29)$$

The Jacobian can be derived using the identity $J(X^T X) = 2X^T J_X$ as follows:

$$\begin{aligned} J_g(q) &= \frac{1}{2} (-2 p_d^T J_f(q) + 2 f(q)^T J_f(q)) \\ &= (f(q) - p_d)^T J_f(q) \quad \in \mathbb{R}^{1 \times ns}. \end{aligned} \quad (3.30)$$

Another approach can be used to find $J_g(q)$ based on the Jacobian chain rule where $J_g(f(q)) = J_g(f) J_f(q)$, leading to the same results as in (3.30). Furthermore, similar to (3.26), the gradient is:

$$\nabla g(q) = J_g(q)^T = \nabla f(q) (f(q) - p_d) \quad \in \mathbb{R}^{ns}. \quad (3.31)$$

However, since the Jacobian $J_g(q)$ in (3.30) does not have a square shape, a modified solution for the iteration scheme (3.21) is required in order to solve the IK. More on that in Section 3.4.

Finally, the optimisation Hessian can be derived using the chain rule [126] as follows:

$$\begin{aligned} H_g(q) &= H(g \circ f)(q) \\ &= J_f(q)^T H_g(f) J_f(q) + \sum_{\mu=1}^m \frac{\partial g}{\partial f_\mu} H_{f_\mu}(q) \in \mathbb{R}^{n_s \times n_s}, \end{aligned} \quad (3.32)$$

where $H_g(f) = I_{m \times m}$ and f_μ represents the μ -th component of f . For (3.28), $m = 3$ and $\frac{\partial g}{\partial f}$ can be expressed as:

$$\begin{aligned} \frac{\partial g}{\partial f} &= J_g(f) = (f(q) - p_d)^T \\ &= [(f[1] - p_d[1]) \quad (f[2] - p_d[2]) \quad (f[3] - p_d[3])], \end{aligned} \quad (3.33)$$

where each $\frac{\partial g}{\partial f_\mu} \in \mathbb{R}$. Substituting in (3.32) gives the optimisation Hessian $H_g(q)$ as:

$$H_g(q) = J_f(q)^T J_f(q) + \sum_{\mu=1}^m (f[\mu] - p_d[\mu]) H_{f_\mu}(q) \in \mathbb{R}^{n_s \times n_s}. \quad (3.34)$$

An example of deriving the Hessian chain rule is provided in Appendix A.

Previous works attempted to simplify the calculation of the Hessian by ignoring the second term of (3.34), leading to approximations using methods such as Gauss-Newton or BFGS methods [119]. More on Hessian approximation can be found in Section 3.6.5 below. Furthermore, similar to (3.34), the Hessian chain rule is discussed in multivariable calculus textbooks such as [316, Ch 6.10] and [317, Ch 4.4.3.2], as well as in inverse kinematics for computer graphics [119]. Readers who wish to explore further details can consult these references.

3.2.3 Manipulability Index

When an actuator is at a singular configuration, certain directions of movement may require high joint speeds and forces. The term 'joint' refers to the variables of the configuration space $q \in \mathbb{R}^{n_s}$, as defined earlier in Section 3.1.2. Near a singularity, movement can also become difficult in specific directions. The manipulability of an actuator describes its ability to move freely in all directions within the workspace [307].

Manipulability provides insights into the actuator's capability to change its position or orientation at or near a given configuration. Manipulability can be represented as an ellipsoid, which visually illustrates the directions in which the actuator's end point can move with the least or greatest effort [43].

The manipulability ellipsoid provides a geometric representation of the end point velocities $\dot{p}_a$ based on the joint velocities $\dot{q}$, which satisfy $|\dot{q}| = 1$, representing a unit sphere in the n_s -dimensional joint-velocity space [43]. Assuming joint velocities have a unit magnitude simplifies the manipulability analysis. Without this condition,

manipulability is influenced by the configuration of the actuator and the magnitude of its joint velocities.

For simplicity, J_f and J_g are used in this section to represent $J_f(q)$ and $J_g(q)$, respectively. Assuming the actuator Jacobian J_f is invertible, the unit joint-velocity condition can be expressed as [43]:

$$\begin{aligned}
 1 &= \dot{q}^T \dot{q} \\
 &= (J_f^{-1} \dot{p}_a)^T (J_f^{-1} \dot{p}_a) \\
 &= \dot{p}_a^T (J_f^{-1})^T J_f^{-1} \dot{p}_a \\
 &= \dot{p}_a^T (J_f J_f^T)^{-1} \dot{p}_a \\
 &= \dot{p}_a^T J_{\square}^{-1} \dot{p}_a.
 \end{aligned} \tag{3.35}$$

When J_f is of rank m , or full rank, the matrix $J_{\square} = J_f J_f^T \in \mathbb{R}^{m \times m}$ is square, symmetric, and positive definite, as is its inverse $J_{\square}^{-1}$. For such a matrix, the set of vectors $\dot{p}_a$ that satisfy (3.35) defines the manipulability ellipsoid in the m -dimensional space.

Since the Jacobian J_f incorporates both linear and angular Jacobians, two distinct three-dimensional manipulability ellipsoids arise due to the different units for linear and angular velocities [43]. It is often more practical to focus on the linear manipulability ellipsoid when the IK target involves following a specific path, regardless of the end point's orientation.

Let $\vec{v}_{\mu}$ and λ_{μ} denote the eigenvectors and eigenvalues of $J_{\square}$, respectively, where $\mu = 1, 2, \dots, m$. For the linear velocity manipulability, where $m = 3$, the directions of the ellipsoid's principal axes are given by the eigenvectors $\vec{v}_{\mu}$, and the length of each principal semi-axis corresponds to $\sqrt{\lambda_{\mu}}$, as illustrated in Figure 3.5 below. The volume of this manipulability ellipsoid is:

$$\begin{aligned}
 V_{me} &= \frac{4\pi}{3} \sqrt{\lambda_1 \lambda_2 \lambda_3} \\
 &= \frac{4\pi}{3} \sqrt{\det(J_{\square})} \\
 &= \frac{4\pi}{3} \sqrt{\det(J_f J_f^T)}.
 \end{aligned} \tag{3.36}$$

where $J_{\square} \in \mathbb{R}^{3 \times 3}$. In scenarios where $m > 3$, the ellipsoid extends to a hyper-ellipsoid. Its volume V_{me} is proportional to the product of the lengths of the semi-axes, which are given by $\sqrt{\lambda_1 \lambda_2 \dots \lambda_m}$, as detailed in [43, 318].

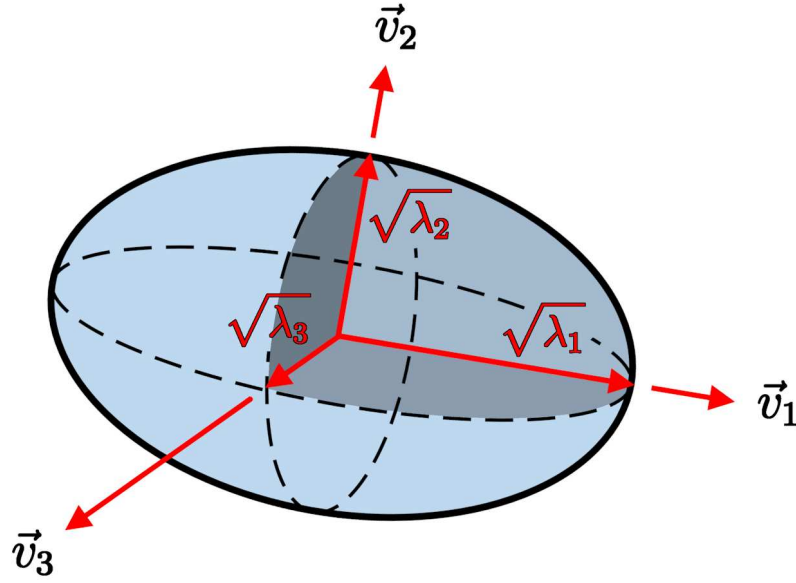


Figure 3.5: A triaxial ellipsoid representation of $\dot{p}_a^T J_\square^{-1} \dot{p}_a = 1$ in the $\dot{p}_a \in \mathbb{R}^3$ space. The principal axes are the eigenvectors $\vec{v}_\mu$ of $J_\square$, and the length of each principle semi-axis is equal to the square root of the corresponding eigenvalue, $\sqrt{\lambda_\mu}$.

Several measures for quantifying manipulability were proposed in the literature [319–323], some of which rely on the singular values of $J_\square$. One commonly used measure is Yoshikawa’s Manipulability Index (YMI) [324], which indicates how far the actuator is from singular configurations. This index is based on the volume of the manipulability ellipsoid, as follows:

$$\text{YMI} = \sqrt{\det(J_f J_f^T)} = \sqrt{\lambda_1 \lambda_2 \cdots \lambda_m}, \quad (3.37)$$

where YMI value typically approaches zero when the actuator is near a singular configuration. However, YMI can be an inadequate measure of manipulability because the volume of the ellipsoid does not necessarily decrease near singularities. This occurs because, near singularities, one of the ellipsoid’s radii becomes small while another becomes large [319].

Alternatively, the inverted conditioning index (ICI) [307] of J_f offers a measure of the spatial uniformity of the manipulability ellipsoid, and defined as:

$$\text{ICI} = \frac{\sigma_{\min}(J_f)}{\sigma_{\max}(J_f)} = \sqrt{\frac{\lambda_{\min}}{\lambda_{\max}}} \in [0, 1]. \quad (3.38)$$

This measure represents the ratio of the smallest to largest singular values of J_f . It is also equivalent to the square root of the ratio of the smallest to largest eigenvalues of $J_\square$. The ICI measure provides a normalised indication of the actuator’s manipulability. Specifically, the ICI is zero at singular configurations and reaches one when the manipulability ellipsoid is nearly spherical (or isotropic), indicating that it is equally easy to move in any direction [43]. It is important to remember that the manipulability index is based on the actuator’s configuration at any given

time, regardless of the chosen IK solver. Therefore, it is derived from the actuator Jacobian J_f rather than the optimisation Jacobian J_g .

The ICI measure, however, has a limitation as it only considers motion in two directions compared to YMI, which considers all of the radii of the manipulability ellipsoid. This limitation becomes important when the manipulability index is used to control the movement of the actuator, especially when avoiding obstacles along the movement path. This topic is outside the scope of this thesis, and interested readers can refer to [319, 320, 322] for further details. For the rest of this chapter, the ICI measure is considered in quantifying the manipulability of the soft actuator.

3.3 Jacobian and Hessian Arrays

For a single segment, the Jacobian $J_s(q_s)$ and Hessian $H_s(q_s)$ arrays of f can be determined via the first and second derivatives of the pose T_{s-1}^s with respect to time t , respectively. An inverse skew function $\vee_{\times}(\cdot)$ is required for determining the rotational parts of $J_s(q_s)$, as described in the following sections.

3.3.1 Inverse Skew Function

A matrix $\mathcal{S}$ that satisfies both the identities $\mathcal{S}^T = -\mathcal{S}$ and $\det(\mathcal{S}) = 0$ is called a skew-symmetric matrix [98]. For an arbitrary vector $\vec{s} \in \mathbb{R}^3$, the skew function $[\cdot]_{\times} : \mathbb{R}^3 \mapsto \mathbb{R}^{3 \times 3}$ maps the vector $\vec{s}$ to the skew-symmetric matrix $\mathcal{S}$. Conversely, the inverse skew function $\vee_{\times}(\cdot) : \mathbb{R}^{3 \times 3} \mapsto \mathbb{R}^3$ maps the skew-symmetric matrix $\mathcal{S}$ to the vector $\vec{s}$ as follows:

$$\begin{aligned} \vec{s} = \vee_{\times}(\mathcal{S}) &= \begin{bmatrix} \vec{s}_1 \\ \vec{s}_2 \\ \vec{s}_3 \end{bmatrix}, \\ \mathcal{S} = [\vec{s}]_{\times} &= \begin{bmatrix} 0 & -\vec{s}_3 & \vec{s}_2 \\ \vec{s}_3 & 0 & -\vec{s}_1 \\ -\vec{s}_2 & \vec{s}_1 & 0 \end{bmatrix}. \end{aligned} \tag{3.39}$$

3.3.2 First Derivative of Pose

For simplicity, the notations T , R , and $\tilde{t}$ are used instead of T_{s-1}^s , R_{s-1}^s , and $\tilde{t}_{s-1}^s$ in this section, respectively. Similar to the procedure of [325], the first derivative of the pose with respect to time t is:

$$\dot{T} = \frac{dT}{dt} = \frac{\partial T}{\partial q_{s1}} \dot{q}_{s1} + \cdots + \frac{\partial T}{\partial q_{sn}} \dot{q}_{sn} \in \mathbb{R}^{4 \times 4}. \tag{3.40}$$

Each $\frac{\partial T}{\partial q_{si}} \in \mathbb{R}^{4 \times 4}$ can be written in partitioned form as:

$$\frac{\partial T}{\partial q_{si}} = \begin{bmatrix} J_{sRi} & J_{sti} \\ \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} & 0 \end{bmatrix}, \quad i = 1, \dots, n, \quad (3.41)$$

where $J_{sRi} \in \mathbb{R}^{3 \times 3}$ and $J_{sti} \in \mathbb{R}^{3 \times 1}$ are the time derivative Jacobians of the rotation matrix R and the translation vector $\tilde{t}$, respectively, and i is the parameter index of the configuration space vector q_s .

Similarly, the first derivative of (3.2) with respect to time t is:

$$\dot{T} = \begin{bmatrix} \dot{R} & \dot{\tilde{t}} \\ \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} & 0 \end{bmatrix}. \quad (3.42)$$

Re-writing (3.40) using (3.41) and (3.42) gives:

$$\begin{bmatrix} \dot{R} & \dot{\tilde{t}} \\ \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} & 0 \end{bmatrix} = \begin{bmatrix} J_{sR1} & J_{st1} \\ \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} & 0 \end{bmatrix} \dot{q}_{s1} + \dots + \begin{bmatrix} J_{sRn} & J_{stn} \\ \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} & 0 \end{bmatrix} \dot{q}_{sn}, \quad (3.43)$$

then, writing a matrix equation for each non-zero partition gives:

$$\dot{R} = J_{sR1} \dot{q}_{s1} + \dots + J_{sRn} \dot{q}_{sn}, \quad (3.44)$$

$$\dot{\tilde{t}} = J_{st1} \dot{q}_{s1} + \dots + J_{stn} \dot{q}_{sn}, \quad (3.45)$$

where each term represents the contribution of the corresponding parameter within the configuration space vector q_s to the end point velocity of each segment.

Starting with (3.45), simplifying and re-writing gives:

$$\begin{aligned} \dot{\tilde{t}} &= [J_{st1} \dots J_{stn}] \begin{bmatrix} \dot{q}_{s1} \\ \vdots \\ \dot{q}_{sn} \end{bmatrix}, \\ &= J_{sv}(q_s) \dot{q}_s, \end{aligned} \quad (3.46)$$

where $J_{sv}(q_s) \in \mathbb{R}^{3 \times n}$ is the translational part of the coordinates Jacobian matrix $J_s(q_s)$ of the individual segments within the actuator. The complete derivation of $J_{sv}(q_s)$ is provided in Appendix B, Section B.5, and it is defined as:

$$J_{sv}(q_s) = \begin{bmatrix} \frac{s_{\phi_s}(c_{\theta_s}-1)L_s}{\theta_s} & \frac{c_{\phi_s}(c_{\theta_s}+\theta_s s_{\theta_s}-1)L_s}{\theta_s^2} & \frac{c_{\phi_s}(1-c_{\theta_s})}{\theta_s} \\ \frac{c_{\phi_s}(1-c_{\theta_s})L_s}{\theta_s} & \frac{s_{\phi_s}(c_{\theta_s}+\theta_s s_{\theta_s}-1)L_s}{\theta_s^2} & \frac{s_{\phi_s}(1-c_{\theta_s})}{\theta_s} \\ 0 & \frac{(\theta_s c_{\theta_s}-s_{\theta_s})L_s}{\theta_s^2} & \frac{s_{\theta_s}}{\theta_s} \end{bmatrix}. \quad (3.47)$$

On the other hand, for the rotational part, recall that for special orthogonal SO group, the rotation matrix satisfies the orthonormality condition $RR^T = I$. Taking the time derivative and simplifying leads to the identity $\dot{R} = [\omega_s]_{\times} R$ where $[\omega_s]_{\times} \in so(3)$ is the skew-symmetric matrix defined as:

$$[\omega_s]_{\times} = \begin{bmatrix} 0 & -\omega_{sz} & \omega_{sy} \\ \omega_{sz} & 0 & -\omega_{sx} \\ -\omega_{sy} & \omega_{sx} & 0 \end{bmatrix}, \quad (3.48)$$

in which $\omega_s = [\omega_{sx} \ \omega_{sy} \ \omega_{sz}]^T \in \mathbb{R}^3$ is the rotational speed at the end point of the individual segments relative to the base frame of the segment $\{S_{s-1}\}$. Substituting in (3.44) and re-writing gives:

$$[\omega_s]_{\times} R = J_{sR1} \dot{q}_{s1} + \cdots + J_{sRn} \dot{q}_{sn}, \quad (3.49)$$

then re-arranging to:

$$[\omega_s]_{\times} = (J_{sR1} R^T) \dot{q}_{s1} + \cdots + (J_{sRn} R^T) \dot{q}_{sn}, \quad (3.50)$$

where each $J_{sRi} R^T$ term must be skew-symmetric for the summation to result in a skew-symmetric matrix..

The matrix equation (3.50) has an order of 3×3 which leads to 3 unique equations. Applying the inverse skew function, Section 3.3.1 above, to both sides yields:

$$\begin{aligned} \omega_s &= \vee_{\times} (J_{sR1} R^T) \dot{q}_{s1} + \cdots + \vee_{\times} (J_{sRn} R^T) \dot{q}_{sn} \\ &= \left[\vee_{\times} (J_{sR1} R^T) \quad \cdots \quad \vee_{\times} (J_{sRn} R^T) \right] \begin{bmatrix} \dot{q}_{s1} \\ \vdots \\ \dot{q}_{sn} \end{bmatrix} \\ &= J_{s\omega}(q_s) \dot{q}_s, \end{aligned} \quad (3.51)$$

where $J_{s\omega}(q_s) \in \mathbb{R}^{3 \times n}$ is the rotational part of the coordinates Jacobian matrix $J_s(q_s)$ of the individual segments within the actuator.

The complete derivation of $J_{s\omega}(q_s)$ is provided in Appendix B, Section B.6, and it is defined as:

$$J_{s\omega}(q_s) = \begin{bmatrix} 0 & -s_{\phi_s} & 0 \\ 0 & c_{\phi_s} & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad (3.52)$$

Combining both (3.46) and (3.51) provides the time derivative of the individual segment forward kinematics as:

$$\begin{bmatrix} v_s \\ \omega_s \end{bmatrix} = J_s(q_s) \dot{q}_s \quad (3.53)$$

where $v_s = [v_{sx} \ v_{sy} \ v_{sz}]^T \in \mathbb{R}^3$ represents the translational speed at the end point of the individual segments, and $J_s(q_s)$ is the coordinate Jacobian matrix of the individual segments relative to the base frame of the segment $\{S_{s-1}\}$, as follows:

$$J_s(q_s) = \begin{bmatrix} J_{sv}(q_s) \\ J_{s\omega}(q_s) \end{bmatrix} \in \mathbb{R}^{6 \times n} \quad (3.54)$$

3.3.3 Second Derivative of Pose

To provide a converged IK solution, the CVX constraints function (3.28) requires only the Hessian $H_f(q)$ translational part, $H_{sv}(q_s)$, of each segment. Consequently, deriving the Hessian $H_f(q)$ rotational part is unnecessary and outside the scope of this thesis. Interested readers can follow the procedure of [325] to derive the complete Hessian $H_f(q)$.

Starting from (3.46) and applying the product rule, the second derivative of the translation vector $\tilde{t}$ with respect to time t is:

$$\underbrace{\ddot{\tilde{t}}}_{m \times 1} = \underbrace{\frac{d}{dt}(J_{sv}(q_s))}_{m \times n} \underbrace{\dot{q}_s}_{n \times 1} + \underbrace{J_{sv}(q_s)}_{m \times n} \underbrace{\ddot{q}_s}_{n \times 1}. \quad (3.55)$$

The term $\frac{d}{dt}(J_{sv}(q_s))$ represents the time derivative of the Jacobian $J_{sv}(q_s)$, and must be of size $m \times n$ to facilitate the matrix multiplication in (3.55). Applying the chain rule, the time derivative of the Jacobian $J_{sv}(q_s)$ is:

$$\frac{d}{dt}(J_{sv}(q_s)) = \frac{\partial J_{sv}}{\partial q_s} \bar{\times}_n \dot{q}_s, \quad (3.56)$$

where $\frac{\partial J_{sv}}{\partial q_s} \in \mathbb{R}^{m \times n \times n}$ is a third-rank tensor representing the partial derivatives of the Jacobian $J_{sv}(q_s)$, also known as the *second-order Jacobian* $J_{sv}^2(q_s)$ [326]. The term $\bar{\times}_n$ denotes the n -mode product [125] between the tensor and the vector $\dot{q}_s$ as described in Appendix B, Section B.3.

There are two options to continue the analysis beyond (3.56):

- Solve (3.56) for $J_{sv}^2(q_s)$, and then transpose it to obtain the Hessian $H_{sv}(q_s)$ tensor.
- Alternatively, derive the Hessian $H_{sv}(q_s)$ using the standard definition $H_{sv}(q_s) = \nabla^2 f(q_s)$, which is simpler, and then verify it using (3.56). The verification can

be done by transposing the Hessian $H_{sv}(q_s)$ to get the second-order Jacobian $J_{sv}^2(q_s)$, then substitute it in (3.56) and check if the left side of the equation matches the right side.

The choice in this thesis was to proceed with the second option, as it is easier to implement.

The Hessian $H_{sv}(q_s)$ is a third-rank tensor with fibres corresponding to rows $H_{:jk}$, columns $H_{i:k}$, and tubes $H_{ij:}$, as illustrated in Figure 3.6(a). The Hessian $H_{sv}(q_s)$ tensor can be arranged into an array of m Hessian matrices as follows:

$$H_{sv}(q_s) = [H_{f_1}(q_s) \quad H_{f_2}(q_s) \quad \cdots \quad H_{f_m}(q_s)] \in \mathbb{R}^{n \times m \times n}, \quad (3.57)$$

where $m = 3$ as mentioned earlier in Section 3.2.2. Since the Hessian tensor $H_{sv}(q_s)$ is specific to the translational part only, the functions $f_1, \dots, f_m$ are the components of the translation vector $\tilde{t}$ as defined in (3.3).

The Hessian $H_{sv}(q_s)$ tensor can be derived using the element-wise definition provided in Appendix B, Section B.4.2, as follows:

$$H_{sv}(q_s)_{ijk} = \frac{\partial^2 f_j}{\partial q_i \partial q_k}, \quad (3.58)$$

where q_i and q_k are components of the configuration space vector q_s as defined in (3.1), with indices $i, k = 1, \dots, n$, and $j = 1, \dots, m$. The red, green, and blue colours illustrate the location of each element as shown in Figure 3.6(b). The element-wise definition yields the Hessian tensor as:

$$\begin{aligned} H_{sv}(q_s)[:, 1, :] &= \begin{bmatrix} \frac{c_{\phi_s}(c_{\theta_s}-1)L_s}{\theta_s} & \frac{-s_{\phi_s}(c_{\theta_s}+\theta_s s_{\theta_s}-1)L_s}{\theta_s^2} & \frac{s_{\phi_s}(c_{\theta_s}-1)}{\theta_s} \\ \frac{-s_{\phi_s}(c_{\theta_s}+\theta_s s_{\theta_s}-1)L_s}{\theta_s^2} & \frac{-c_{\phi_s}(2c_{\theta_s}+2\theta_s s_{\theta_s}-\theta_s^2 c_{\theta_s}-2)L_s}{\theta_s^3} & \frac{c_{\phi_s}(c_{\theta_s}+\theta_s s_{\theta_s}-1)}{\theta_s^2} \\ \frac{s_{\phi_s}(c_{\theta_s}-1)}{\theta_s} & \frac{c_{\phi_s}(c_{\theta_s}+\theta_s s_{\theta_s}-1)}{\theta_s^2} & 0 \end{bmatrix} \\ H_{sv}(q_s)[:, 2, :] &= \begin{bmatrix} \frac{s_{\phi_s}(c_{\theta_s}-1)L_s}{\theta_s} & \frac{c_{\phi_s}(c_{\theta_s}+\theta_s s_{\theta_s}-1)L_s}{\theta_s^2} & \frac{-c_{\phi_s}(c_{\theta_s}-1)}{\theta_s} \\ \frac{c_{\phi_s}(c_{\theta_s}+\theta_s s_{\theta_s}-1)L_s}{\theta_s^2} & \frac{-s_{\phi_s}(2c_{\theta_s}+2\theta_s s_{\theta_s}-\theta_s^2 c_{\theta_s}-2)L_s}{\theta_s^3} & \frac{s_{\phi_s}(c_{\theta_s}+\theta_s s_{\theta_s}-1)}{\theta_s^2} \\ \frac{-c_{\phi_s}(c_{\theta_s}-1)}{\theta_s} & \frac{s_{\phi_s}(c_{\theta_s}+\theta_s s_{\theta_s}-1)}{\theta_s^2} & 0 \end{bmatrix} \\ H_{sv}(q_s)[:, 3, :] &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & \frac{(2s_{\theta_s}-2\theta_s c_{\theta_s}-\theta_s^2 s_{\theta_s})L_s}{\theta_s^3} & \frac{\theta_s c_{\theta_s}-s_{\theta_s}}{\theta_s^2} \\ 0 & \frac{\theta_s c_{\theta_s}-s_{\theta_s}}{\theta_s^2} & 0 \end{bmatrix}, \end{aligned} \quad (3.59)$$

in which each Hessian matrix in this tensor is symmetric. The second-order Jacobian $J_{sv}^2(q_s)$, in contrast, can be arranged as an array of n Jacobian derivatives as follows:

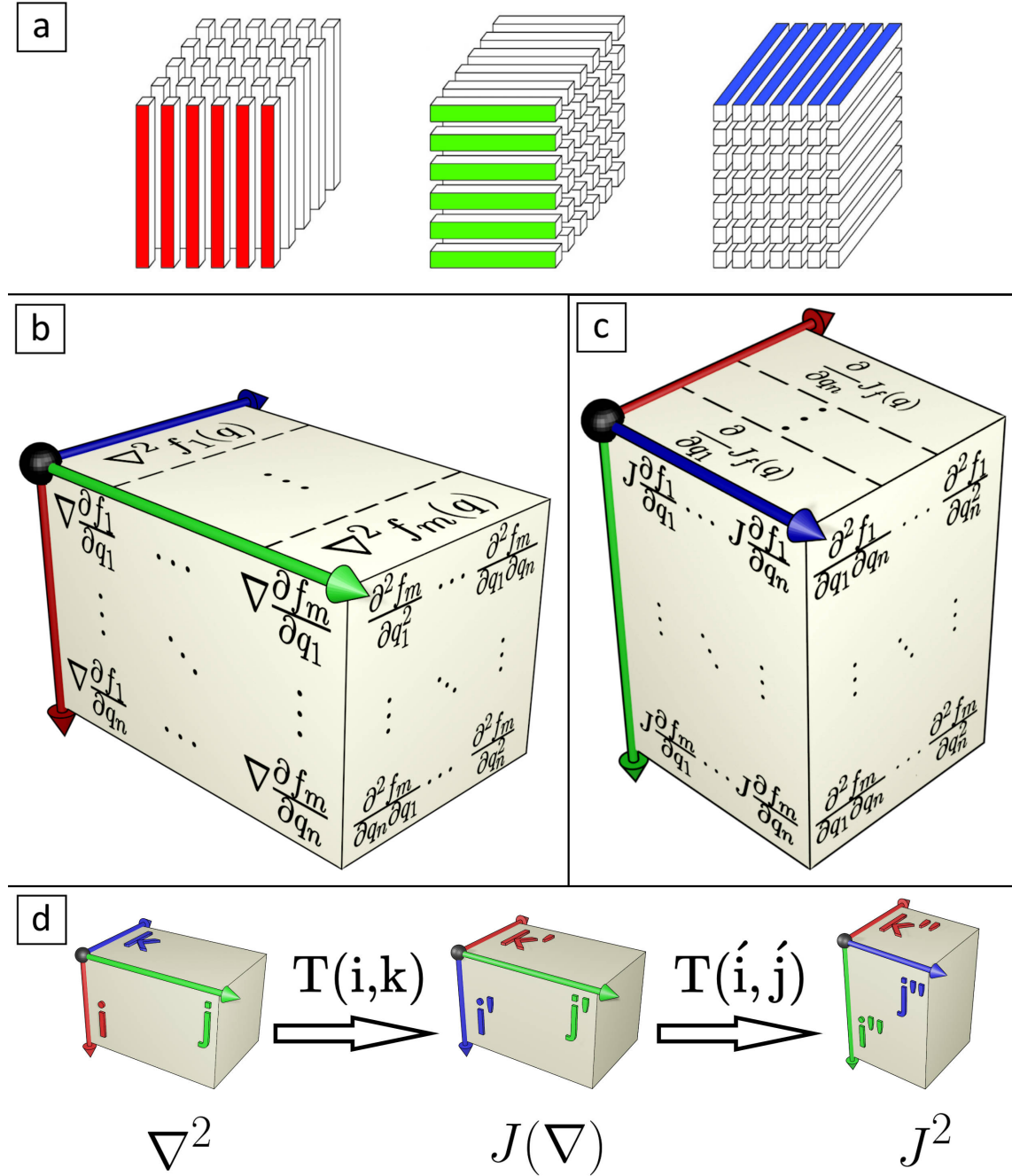


Figure 3.6: The Hessian $H_{sv}(q_s)$ array: (a) illustrates the fibres of the columns $H_{:jk}$ (in red), the rows $H_{i:k}$ (in green), and the tubes $H_{ij:}$ (in blue), (b) depicts the partial derivatives representation as a third-order tensor. (c) shows the visualisation of the second-order Jacobian $J^2_{sv}(q_s)$, generated by a double transpose of the Hessian $H_f(q)$ tensor, while (d) demonstrates the double transpose sequence.

$$J_{sv}^2(q_s) = \begin{bmatrix} \frac{\partial J_{sv}(q_s)}{\partial q_1} & \frac{\partial J_{sv}(q_s)}{\partial q_2} & \dots & \frac{\partial J_{sv}(q_s)}{\partial q_n} \end{bmatrix} \in \mathbb{R}^{m \times n \times n}, \quad (3.60)$$

where the first and second dimensions, $m \times n \times \dots$, represent the Jacobian $J_{sv}(q_s)$, while the third dimension, $\dots \times \dots \times n$, corresponds to each partial derivative of the Jacobian, as shown in Figure 3.6(c).

Similarly, the $J_{sv}^2(q_s)$ tensor can be derived using the element-wise definition described in Appendix B, Section B.4.2, as follows:

$$J_{sv}^2(q_s)_{ijk} = \frac{\partial^2 f_i}{\partial q_k \partial q_j}, \quad (3.61)$$

where q_j and q_k are components of the configuration space vector q_s as defined in (3.1), with indices $j, k = 1, \dots, n$, and $i = 1, \dots, m$. The red, green, and blue colours illustrate the location of each element as shown in Figure 3.6(c).

To generate the $J_{sv}^2(q_s)$ tensor from the $H_{sv}(q_s)$ tensor, two transpose operations are required since $H_{sv}(q_s) = \nabla^2 f(q_s)$. The first transpose is performed on the outer gradient, while the second transpose is applied to the inner gradient as follows:

$$\underbrace{\frac{\partial^2 f_j}{\partial q_i \partial q_k}}_{\nabla^2} \xrightarrow{T(i,k)} \underbrace{\frac{\partial^2 f_{j'}}{\partial q_{i'} \partial q_{k'}}}_{J(\nabla)} \xrightarrow{T(i',j')} \underbrace{\frac{\partial^2 f_{i''}}{\partial q_{k''} \partial q_{j''}}}_{J^2}, \quad (3.62)$$

and as shown in Figure 3.6(d). However, The first transpose, $T(i, k)$, is not needed since the Hessian $H_{sv}(q_s)$ is symmetric. The remaining transpose can be executed in MATLAB using the command `JJs = permute(Hs, [2 1 3])` which switches the first and second dimensions while keeping the third dimension unchanged.

The next step is the n -mode product. Since $J_{sv}^2(q_s)$ is arranged as an array of n Jacobian derivatives along its third dimension, the n -mode product is applied using the third mode, $\bar{\times}_3$, in (3.55). Simplifying both sides of (3.55) confirms their equality, verifying that the derivation of the Hessian $H_{sv}(q_s)$ tensor was correct.

The entire verification process can be performed using the *Symbolic Math Toolbox* in MATLAB, as detailed in the code provided in Appendix C.

3.3.4 Multi-Segment Actuator

As described earlier in Section 3.1.5, a single PCC segment in 3D space is considered non-holonomic since it has only 3 DOF_k in its configuration. Adding more segments provides additional DOF_k to the kinematic chain. A soft actuator with two PCC segments is considered holonomic, and further segments provide redundancy, thereby extending the manipulation capabilities.

The actuator Jacobian $J_f(q)$ is obtained by combining the Jacobians of the individual segments $J_s(q_s)$ after transferring their reference frames to the actuator base frame $\{S_0\}$. Following the notations of [307], the spatial velocity of the end pose p_s of each individual segment relative to the segment base frame $\{S_{s-1}\}$ is:

$$\mathcal{V}_s = \begin{bmatrix} v_s \\ \omega_s \end{bmatrix} \in \mathbb{R}^6. \quad (3.63)$$

Similarly, the spatial velocity of the same end point relative to the actuator base frame $\{S_0\}$, or the origin, is:

$$\mathcal{V}_0 = \begin{bmatrix} v_0 \\ \omega_0 \end{bmatrix} \in \mathbb{R}^6, \quad (3.64)$$

and the homogeneous transformation matrix between the two frames $T_0^{s-1} \in SE(3)$, as shown in Figure 3.7, is defined as:

$$T_0^{s-1} = T_0^1 T_1^2 \dots T_{s-2}^{s-1} = \begin{bmatrix} \Omega & \Psi \\ \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} & 1 \end{bmatrix}, \quad (3.65)$$

where $\Omega \in SO(3)$ is the rotation matrix and $\Psi \in \mathbb{R}^3$ is the translation vector.

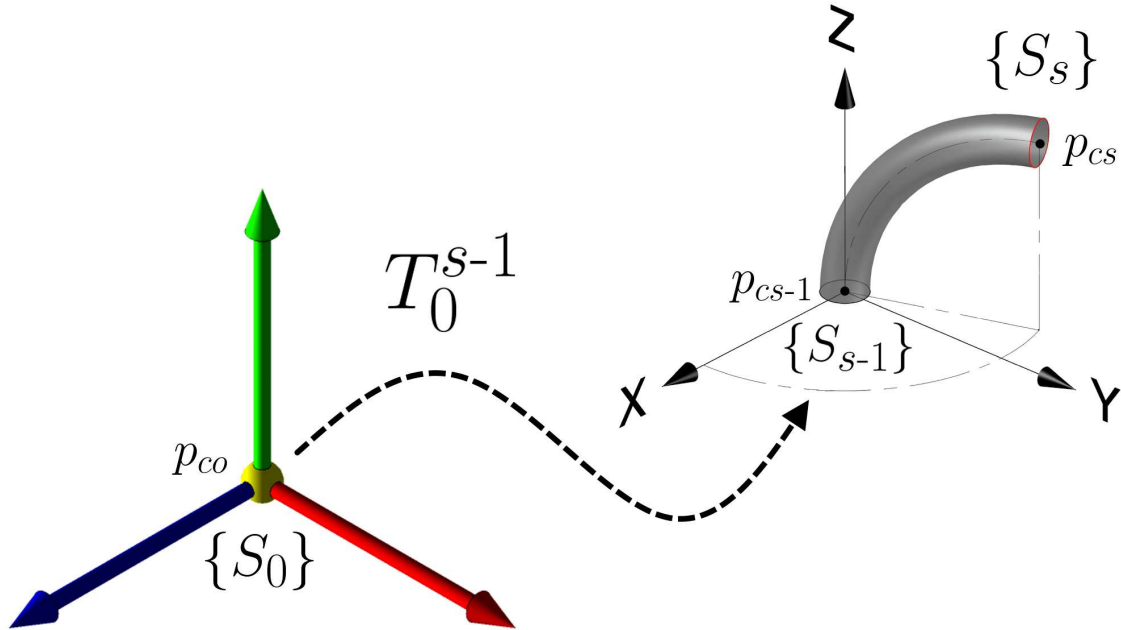


Figure 3.7: The location of the segment end frame $\{S_s\}$ and base frame $\{S_{s-1}\}$ relative to the actuator base frame $\{S_0\}$ in the 3D space, along with the Cartesian components of the pose at each frame, p_{cs} , p_{cs-1} , and p_{co} , respectively.

Similar to (3.10), the coordinates of the point p_{cs-1} , the Cartesian part of the pose p_{s-1} , at the origin of the segment base frame $\{S_{s-1}\}$ can be represented relative to the point p_{co} at the origin of the actuator base frame $\{S_0\}$ as follows:

$$\begin{bmatrix} p_{cs-1} \\ 1 \end{bmatrix} = T_0^{s-1} \begin{bmatrix} p_{co} \\ 1 \end{bmatrix}. \quad (3.66)$$

For simplicity, the notation T is used instead of T_0^{s-1} for the remainder of this section. Differentiating (3.66) with respect to time t and simplifying yields:

$$\begin{bmatrix} \dot{p}_{cs-1} \\ 0 \end{bmatrix} = \dot{T} \begin{bmatrix} p_{co} \\ 1 \end{bmatrix} = \dot{T} T^{-1} \begin{bmatrix} p_{cs-1} \\ 1 \end{bmatrix}. \quad (3.67)$$

Now, let us evaluate $\dot{T}T^{-1}$:

$$\begin{aligned} \dot{T}T^{-1} &= \begin{bmatrix} \dot{\Omega} & \dot{\Psi} \\ [0 & 0 & 0] & 0 \end{bmatrix} \begin{bmatrix} \Omega^T & -\Omega^T \Psi \\ [0 & 0 & 0] & 1 \end{bmatrix} \\ &= \begin{bmatrix} \dot{\Omega}\Omega^T & -\dot{\Omega}\Omega^T\Psi + \dot{\Psi} \\ [0 & 0 & 0] & 0 \end{bmatrix}, \end{aligned} \quad (3.68)$$

which has the form of a 4×4 matrix representing the spatial velocity relative to the base frame of the actuator $\{S_0\}$, as defined earlier in 3.64, and denoted as $\hat{\mathcal{V}}_0 \in se(3)$. This matrix is associated with the configuration $SE(3)$, or the Lie algebra of the Lie group $SE(3)$. Therefore, (3.68) can be expressed as:

$$\hat{\mathcal{V}}_0 = \begin{bmatrix} [\omega_0]_{\times} & v_0 \\ [0 & 0 & 0] & 0 \end{bmatrix} = \dot{T}T^{-1}. \quad (3.69)$$

Since ω_0 and ω_s represent the same angular velocity of the end point pose p_s but in two different frames, these two vectors are related as follows:

$$\omega_s = \Omega^{-1}\omega_0 = \Omega^T\omega_0. \quad (3.70)$$

Then, by utilising [43, Proposition 3.8], the relation in 3.70 can be expressed in skew-symmetric matrix form as:

$$\begin{aligned} [\omega_s]_{\times} &= [\Omega^T\omega_0]_{\times} = \Omega^T[\omega_0]_{\times}\Omega \\ &= \Omega^T(\dot{\Omega}\Omega^T)\Omega \\ &= \Omega^T\dot{\Omega} = \Omega^{-1}\dot{\Omega}. \end{aligned} \quad (3.71)$$

In the same fashion, let us evaluate $T^{-1}\dot{T}$:

$$T^{-1}\dot{T} = \begin{bmatrix} \Omega^T & -\Omega^T\Psi \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{\Omega} & \dot{\Psi} \\ 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} \Omega^T\dot{\Omega} & \Omega^T\dot{\Psi} \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad (3.72)$$

which takes the form of a 4×4 matrix representing the spatial velocity relative to the base frame of the segment $\{S_{s-1}\}$, as defined earlier in 3.63, and denoted as $\widehat{\mathcal{V}}_s \in se(3)$. Again, this matrix is associated with the configuration $SE(3)$, or the Lie algebra of the Lie group $SE(3)$. Therefore, (3.72) can be expressed as:

$$\widehat{\mathcal{V}}_s = \begin{bmatrix} [\omega_s]_{\times} & v_s \\ 0 & 0 & 0 & 0 \end{bmatrix} = T^{-1}\dot{T}. \quad (3.73)$$

Combining (3.69) and (3.73) yields the following:

$$\widehat{\mathcal{V}}_0 = T \widehat{\mathcal{V}}_s T^{-1} = \begin{bmatrix} \Omega[\omega_s]_{\times}\Omega^T & -\Omega[\omega_s]_{\times}\Omega^T\Psi + \Omega v_s \\ 0 & 0 & 0 & 0 \end{bmatrix}. \quad (3.74)$$

Considering that cross product is antisymmetric, $[\omega]_{\times}\Psi = -[\Psi]_{\times}\Omega$, and by utilising [43, Proposition 3.8], (3.74) can be simplified to the following form:

$$\begin{bmatrix} v_0 \\ \omega_0 \end{bmatrix} = \begin{bmatrix} \Omega & [\Psi]_{\times}\Omega \\ 0_{3 \times 3} & \Omega \end{bmatrix} \begin{bmatrix} v_s \\ \omega_s \end{bmatrix} = Ad_T \begin{bmatrix} v_s \\ \omega_s \end{bmatrix}, \quad (3.75)$$

where Ad_T is a 6×6 matrix known as the *adjoint transformation* and can be generalised to the following form [43, Proposition 3.21]:

$$Ad_T = Ad_{T_0^1 T_1^2 \dots T_{s-2}^{s-1}} = Ad_{T_0^1} Ad_{T_1^2} \dots Ad_{T_{s-2}^{s-1}}. \quad (3.76)$$

Substituting (3.53) in (3.75) yields the full Jacobian of the actuator as follows:

$$J_f(q) = \begin{bmatrix} J_v(q) \\ J_w(q) \end{bmatrix} = \begin{bmatrix} J_1(q_1) & Ad_{T_0^1} J_2(q_2) & \dots & Ad_{T_0^{s-1}} J_s(q_s) \end{bmatrix} \in \mathbb{R}^{m \times ns}, \quad (3.77)$$

where $m = 6$, and s is the total number of segments in the actuator as defined earlier in (3.1). The full Jacobian $J_f(q)$ can then be substituted in (3.25) to calculate the inverse kinematics as discussed in Section 3.2.2 above.

For the full Hessian $H_f(q)$ of the actuator, since only the translational part $H_v(q)$ is required as explained in Section 3.2.2, let us expand (3.77) to obtain the full translational Jacobian $J_v(q)$ as follows:

$$[J_v(q)]_s = \Omega J_{sv}(q_s) + [\Psi]_\times \Omega J_{sw}(q_s), \quad (3.78)$$

where $[J_v(q)]_s \in \mathbb{R}^{3 \times n}$ is a sub-matrix of the full translational Jacobian $J_v(q)$ associated with the s -th segment. Applying the partial derivative $\frac{\partial}{\partial q_s}$ to both sides yields the second-order Jacobian:

$$\begin{aligned} [J_v^2(q)]_s &= \frac{\partial \Omega}{\partial q_s} J_{sv}(q_s) + \Omega J_{sv}^2(q_s) + \frac{\partial [\Psi]_\times}{\partial q_s} \Omega J_{sw}(q_s) \\ &\quad + [\Psi]_\times \frac{\partial \Omega}{\partial q_s} J_{sw}(q_s) + [\Psi]_\times \Omega J_{sw}^2(q_s), \end{aligned} \quad (3.79)$$

However, simplifying (3.79) necessitates determining the correct n -mode products and the appropriate transpose to derive $H_v(q)$. This mathematical problem remains unsolved despite significant progress in analysing it and reviewing related studies in the literature [119, 325, 327–331]. A complete solution remains elusive due to its complexity and the time constraints of this project. It is believed that solving this problem could potentially enhanced the **IK** solver for redundant soft actuators, specially when obstacle avoidance is required. The work presented in this chapter can serve as a stepping stone for solving this open problem, as it presents an exciting opportunity for future research.

3.3.5 The Actuator Physical Limits

SPAs have physical limits that define the size of the workspace. Consequently, any **IK** solutions beyond these limits are generally not feasible [58]. The **IK** algorithm must account for these limits during iterations to achieve the best possible configuration that minimises the error to the target pose.

SPAs are typically designed to be non-compressible. Hence, the minimum segment length is equal to 100% of the initial length. Similarly, the maximum length is determined by the actuator design, which may be constant (i.e., 100% of the initial length) or longer if extension is permitted. For the bending angle θ , the minimum limit is usually zero at the straight position, while the maximum limit is also determined by the actuator design; see [332] for example.

Additionally, in the **IK** algorithm, the physical limits must be applied individually to each segment rather than to the entire actuator as a whole. Otherwise, some segments may yield infeasible results even if the target pose is within the actuator's workspace; see [58, Figure 9] for example.

Projecting the workspace onto the XY , XZ , and YZ planes aids in understanding the feasible results. For example, consider a single-segment **SPA** with a unit length, $L_o = 1$, rotating anticlockwise about the z -axis. The end point traces a spiral path, as illustrated in Figure 3.8. This path was generated by varying the segment shape from the initial, straight, configuration where $\phi_s = 0$, $\theta_s = 0$, and $L_s = 2L_o$, to the final configuration where $\phi_s = \frac{3}{2}\pi$, $\theta_s = \frac{3}{2}\pi$, and $L_s = L_o$. Note that the starting length was chosen to be longer than the initial length, which was achieved

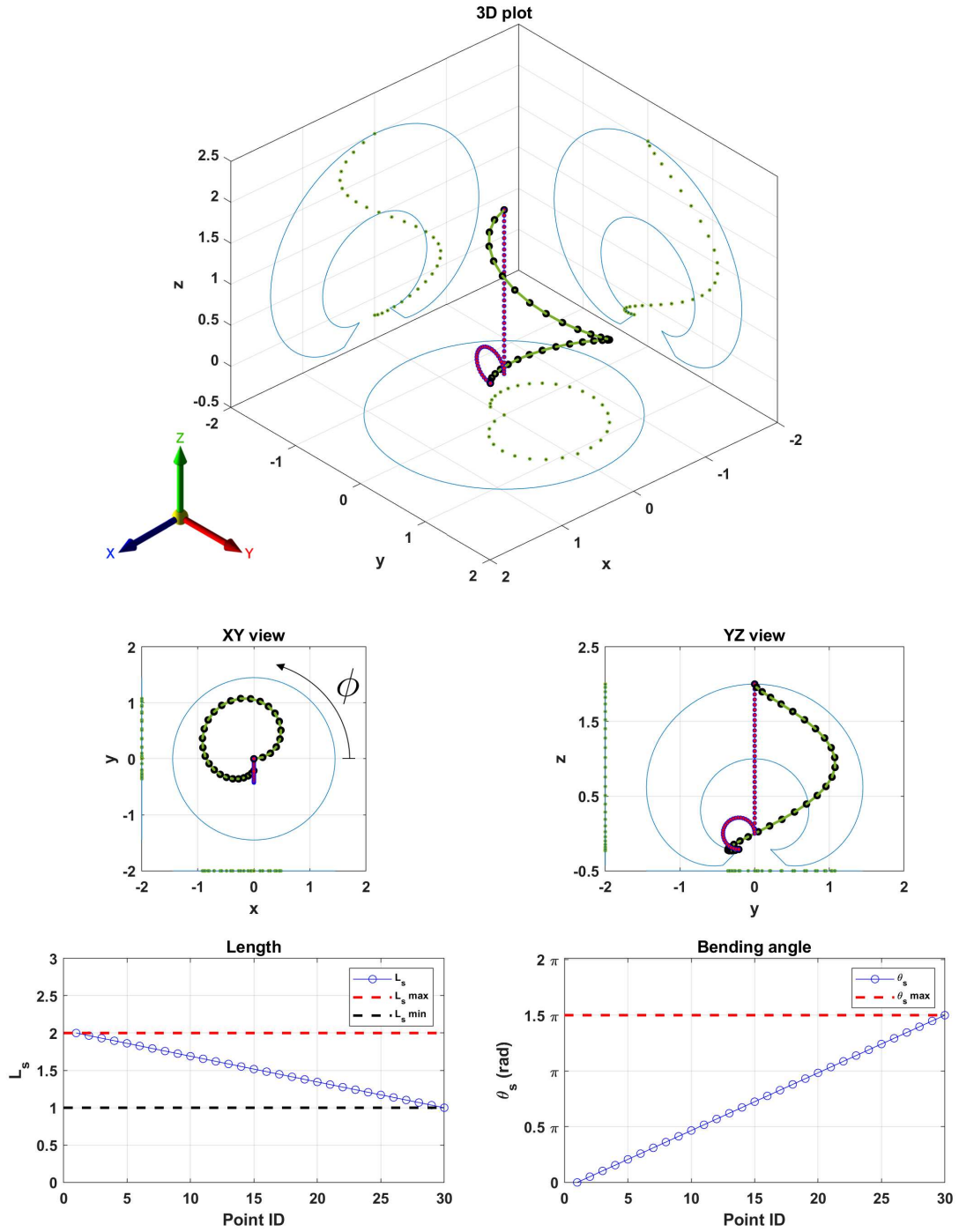


Figure 3.8: A single-segment actuator rotating about the z -axis, with its end point generating a spiral path marked in green. The actuator base is positioned at the origin (0,0,0). The path, consisting of 30 points, is projected onto the XY , XZ , and YZ planes, along with the actuator's workspace. The actuator's initial, straight, configuration and final configuration are plotted in purple. The XY and YZ views provide additional viewpoints of the spiral path, actuator, and projections. The length L_s and bending angle θ_s values along the path are also plotted against their minimum and maximum limits for improved visualisation.

by extending the SPA while it was still in its straight position, prior to applying rotation and bending.

The workspace limits, maximum and minimum boundaries, are projected onto the XZ and YZ planes. The projection onto the XY plane generates four concentric circles, corresponding to the two maximum and two minimum boundaries of the workspace volume. However, only the maximum limit is shown on the XY plane to reduce clutter in the plot. The end point path is divided into 30 points, calculated based on the FK of the SPA, ensuring that each point is within the workspace of the SPA.

Nevertheless, some points on the path appear outside the workspace projections. This discrepancy arises as a result of the semi-spherical shape of the workspace. To provide better visualisation, the length L_s and bending angle θ_s values were plotted separately along with their limits.

3.4 Jacobian Algorithms

As Newton's method requires the inversion of the Jacobian $J_g(q)$ in the linear iteration scheme (3.21), the Jacobian may not be square or invertible. Additionally, the Jacobian can encounter singularity issues, as discussed earlier in Section 3.1.2.

Alternative solutions were presented in the literature [99, 100], as indicated earlier in Chapter 2, Section 2.1.5. Each solution offers a modified version of the iteration scheme (3.21) and can serve as a basis for comparison with the minimum bending limit approach suggested earlier in Section 3.1.3. Let us re-arrange (3.21) in a general form and use an arbitrary Jacobian matrix $J^\diamond$ as follows:

$$\Delta q = -J^\diamond g(q^{\mathbb{K}}), \quad (3.80)$$

where $\Delta q = q^{\mathbb{K}+1} - q^{\mathbb{K}}$, and $J^\diamond = J_g(q^{\mathbb{K}})^{-1}$ if the JIV method is used, as in (3.21). For simplicity, the notations J_g and g are used instead of $J_g(q^{\mathbb{K}})$ and $g(q^{\mathbb{K}})$, respectively, for the remainder of this section.

Previous studies suggested alternative solution to the JIV method, and five of these were addressed earlier in Chapter 2, Section 2.1.5. A summary of these five solutions is provided below, where each solution proposes a different definition for $J^\diamond$ along with their acronyms to identify related results later in Section 3.6.

3.4.1 Jacobian Transpose (JTR)

This solution simply uses the transpose of J_g instead of the inverse to ensure an invertible Jacobian, as follows:

$$J^\diamond = \alpha_s J_g^T, \quad (3.81)$$

where α_s is a scalar representing the iteration size, calculated as:

$$\alpha_s = \frac{\langle g, J_g J_g^T g \rangle}{\langle J_g J_g^T g, J_g J_g^T g \rangle}, \quad (3.82)$$

in which $\langle \cdot, \cdot \rangle$ indicates the dot product between two vectors. Further details are provided in Section 3.5 below.

3.4.2 Jacobian Pseudo-Inverse (JPI)

The Jacobian pseudo-inverse J^+ is defined for all Jacobian matrices, even if they are not square or do not have full rank. The Jacobian calculation is as follows:

$$J^\circ = J^+ = J_g^T (J_g J_g^T)^{-1}. \quad (3.83)$$

The pseudo-inverse has a feature where the matrix $(I - J^+ J)$ performs a projection into the null space of J , which is useful for controlling secondary manipulation tasks [333]. However, near a singularity, the JPI method can cause instability and oscillations in motion. This occurs because very small changes in the task space p discrepancy can result in disproportionately large changes in the configuration space q , leading to unpredictable and abrupt movements.

3.4.3 Damped Least-Squared (DLS)

This method utilises damping to stabilise Δq near singularities, which helps to avoid many of JPI method. The Jacobian calculation is as follows:

$$J^\circ = J_g^T (J_g J_g^T + \lambda_d I)^{-1}, \quad (3.84)$$

where $\lambda_d \in \mathbb{R}$ is a positive scalar acting as a damping factor that must be chosen carefully to stabilise (3.80) numerically, and $I \in \mathbb{R}^{m \times m}$ is the identity matrix matching the size of $J_g J_g^T$. Although a large λ_d improves the behaviour of the solutions for Δq near singularities, it also slows convergence, reduces target tracking accuracy, and causes oscillation and shaking. Further details on damping factors are provided in Section 3.5.

3.4.4 Singular Value Decomposition (SVD)

Due to the limitations of the JPI near singularities, the SVD method provides an alternative approach for calculating the Jacobian pseudo-inverse J^+ . The SVD method is advantageous as it decomposes the Jacobian matrix into its fundamental sub-spaces, providing orthonormal bases for these sub-spaces. Specifically, the SVD method involves expressing an $m \times n$ Jacobian J in the factorisation form $J = U \Sigma V^T$, where U and V are $m \times m$ and $n \times n$ orthogonal (or unitary, if complex) matrices, respectively, and Σ is a diagonal $m \times n$ matrix containing non-negative real numbers along its diagonal.

The diagonal entries of the matrix Σ , denoted $\sigma_u = \Sigma_{uu}$, represent the singular values of J . The rank of J is determined by the largest value $\mathcal{R}$ for which $\sigma_{\mathcal{R}} \neq 0$,

implying that $\sigma_u = 0$ for $u > \mathcal{R}$. Using the [SVD](#), the Jacobian pseudo-inverse can be expressed as $J^\dagger = V\Sigma^\dagger U^T$, where it is computed as follows:

$$J^\diamond = J^\dagger = \sum_{u=1}^{\mathcal{R}} \sigma_u^{-1} V_u U_u^T, \quad (3.85)$$

where V_u and U_u denote the u -th columns of V and U , respectively. In MATLAB, the [SVD](#) variation of the Jacobian pseudo-inverse, $J^\dagger$, can be computed using the `pinv(J)` function.

3.4.5 Pseudo-Inverse Damped Least Squares ([PDL](#))

The [PDL](#) method extends the [DLS](#) approach by combining its benefits with the stabilisation features of the [SVD](#) method. When away from singularities, $\sigma_u \gg \lambda_d$, [PDL](#) behaves similarly to [SVD](#), but it provides improved stability near singularities. The method incorporates a damping factor λ_d to stabilise the Jacobian $J^\diamond$ when σ_u approaches zero. The Jacobian calculation is as follows:

$$J^\diamond = \sum_{u=1}^{\mathcal{R}} \frac{\sigma_u}{\sigma_u^2 + \lambda_d^2} V_u U_u^T. \quad (3.86)$$

Other solutions were also developed [100], including some that enhance the five methods discussed here and others that propose different approaches. Interested readers can explore the related literature for further details.

3.4.6 Convex Holonomic Constraints

As previously mentioned in Section 3.2.2, the optimisation Jacobian $J_g(q)$ for a [CVX](#) constraints function $g(q) \in \mathbb{R}$ is reduced to a vector of size $1 \times n\mathfrak{s}$. Consequently, the iteration scheme (3.21) becomes inapplicable, as the Jacobian $J_g(q)$ is not invertible.

In this case, the suitable alternative solutions for handling non-invertible Jacobians are the [JPI](#) and [SVD](#) methods. Additionally, Gradient Descent ([GRD](#)) is proposed as a completely different approach from Newton's method, as it does not involve Jacobian inversion.

Several variations of the [GRD](#) method were explored in the literature, as outlined in Chapter 2, Section 2.1.5. Given the focus of this thesis on Newton's method, the comparison will be made using one specific variation of the [GRD](#) method. The AdaGrad algorithm [96] is chosen for this comparison due to its efficiency and its compatibility with damping profiles, as will be elaborated in Section 3.5.2 below.

The AdaGrad algorithm is defined by the following iterative scheme [96]:

$$q^{\mathbb{K}+1} = q^{\mathbb{K}} - \frac{\alpha_g}{\epsilon + \sqrt{G^{\mathbb{K}}}} \nabla_g(q^{\mathbb{K}}), \quad (3.87)$$

where α_g represents the step size, which can be replaced with the damping factor λ_d to control the convergence speed. The term $\epsilon = 10^{-8}$ is a small constant used

to avoid division by zero. The term $\nabla_g(q)$ is the optimisation gradient defined in (3.31), evaluated at $q^{\mathbb{K}}$, and $G^{\mathbb{K}}$ is a vector that accumulates the sum of the squares of the gradient up to the time step $\mathbb{K}$, defined as follows:

$$G^{\mathbb{K}} = G^{\mathbb{K}-1} + (\nabla_g(q^{\mathbb{K}}))^2 \in \mathbb{R}^{n_s}. \quad (3.88)$$

The initial value of $G^{\mathbb{K}}$ is set to $G^o = 0_{n_s \times 1}$, a zero vector that matches the size of the optimisation gradient $\nabla_g(q)$. The results of the IK analysis using JPI, SVD, and GRD methods are presented and discussed in Section 3.6.

3.5 Damping of Inverse Kinematic

As stated earlier in Chapter 2, Section 2.1.4, the damping factor can prevent large motions or oscillations near singularities or out of the workspace while performing the IK algorithm. The damping factor needs to be carefully chosen to ensure numerical stability.

3.5.1 The Jacobian with Further Damping

While the JTR, DLS, and PDL methods incorporate damping into their equations, the remaining methods can also be modified to include the damping factor λ_d . For the JIV method in (3.80), the Jacobian becomes:

$$J^\diamond = \lambda_d J_g^{-1}. \quad (3.89)$$

For the JTR method in (3.81), the Jacobian becomes:

$$J^\diamond = \lambda_d J_g^T, \quad (3.90)$$

where λ_d can be equivalent to α_s , defined in (3.82), or other values to be introduced below. Additionally, for the JPI method in (3.83), the Jacobian becomes:

$$J^\diamond = \lambda_d J^+. \quad (3.91)$$

Finally, for the SVD method in (3.85), the Jacobian becomes:

$$J^\diamond = \lambda_d J^\dagger. \quad (3.92)$$

The damping factor λ_d can be constant or variable, changing based on either the objective function value at each iteration, see (3.82) for example, or following an iterative profile [104, 334].

3.5.2 Damping Profiles

While a large damping factor produces well-behaved solutions close to singularities, it also slows down convergence, makes tracking targets less accurate, and causes shaking and oscillation [100]. Therefore, several damping profiles are suggested in this thesis, each serving a specific purpose. The description of each profile, along with the recursive form of calculating the damping factor λ_d , is as follows:

- Constant damping (CND)

The idea is to use a constant damping factor λ_c to achieve the fastest possible convergence. The damping factor in this profile is:

$$\lambda_d^{\mathbb{K}} = \lambda_c. \quad (3.93)$$

- Step function damping (SFD)

The idea is to reduce the damping factor incrementally. The step size $\mathbb{K}_{span}$, chosen by the user, determines the number of iterations per step. The reduction amount is automatically calculated based on this step size, allowing for a constant damping factor during each step to minimise its impact on convergence speed. The initial damping value can be set to λ_c or to one, no damping, before starting the convergence iteration. The damping factor in this profile is:

$$\lambda_d^{\mathbb{K}} = \lambda_d^{\mathbb{K}-1} - \text{fix}\left(\frac{\mathbb{K}}{\mathbb{K}_{span}}\right) * \frac{\mathbb{K}_{span}}{\mathbb{K}_{max}}, \quad (3.94)$$

where the `fix()` function in MATLAB rounds numbers to the nearest integer towards zero, and $\mathbb{K}_{max}$ represents the maximum number of iterations allowed before stopping the algorithm. A minimum limit for the damping factor, $\lambda_d^{\mathbb{K}} \geq \lambda_{min}$, can be set manually to avoid excessive impact on convergence speed.

- Linear function damping (LFD)

The idea is to use a smooth linear reduction in the damping factor λ_d based on the number of iterations. This profile maintains a gradual reduction in convergence speed. The initial damping value can be set to λ_c or to one, no damping, before starting the convergence iteration. The damping factor in this profile is:

$$\lambda_d^{\mathbb{K}} = \lambda_d^{\mathbb{K}-1} - \frac{1 - \lambda_{min}}{\mathbb{K}_{max}}, \quad (3.95)$$

where the minimum limit for the damping factor is incorporated into the recursive form, eliminating the need for a separate manual setting.

- Halving function damping ([HFD](#))

This profile is similar to the [SFD](#) profile but features a more rapid reduction. The damping factor λ_d is halved at each $\mathbb{K}_{span}$. This halving reduction has a more significant impact on the convergence speed compared to the linear reduction due to its non-linear behaviour. The damping factor in this profile is:

$$\lambda_d^{\mathbb{K}} = \frac{1}{2^{\text{fix}(\frac{\mathbb{K}}{\mathbb{K}_{span}})}}. \quad (3.96)$$

Similar to the [SFD](#) profile, a minimum limit for the damping factor, $\lambda_d^{\mathbb{K}} \geq \lambda_{min}$, can be manually set to prevent excessive impact on the convergence speed.

- Quadratic function damping ([QFD](#))

This profile builds upon the [LFD](#) profile but exhibits distinct behaviour due to its quadratic nature. Initially, the quadratic function results in a gradual reduction, which has a minimal impact on convergence speed. As iterations progress, the rate of reduction accelerates. This characteristic can facilitate faster overall convergence compared to the [LFD](#) profile, particularly if certain poses in the [IK](#) path require fewer iterations. The damping factor in this profile is:

$$\lambda_d^{\mathbb{K}} = 1 - (1 - \lambda_{min}) \left(\frac{\mathbb{K}}{\mathbb{K}_{max}} \right)^2, \quad (3.97)$$

where the minimum limit for the damping factor is integrated within the recursive form, similar to the [LFD](#) profile, eliminating the need for manual adjustment.

- First order rational function damping ([FRD](#))

This profile achieves a rapid reduction in the damping factor, faster than the [QFD](#) profile. The first-order rational function is represented arbitrarily as $f(x) = \frac{a}{x+b}$, with a and b as constants. These constants are determined by substituting two known points $(x, f(x))$ into the function. Using the values λ_{min} and $\mathbb{K}_{max}$ as boundary conditions, the recursive form of the damping factor is:

$$\lambda_d^{\mathbb{K}} = \frac{\lambda_{rat}}{(\mathbb{K}/\mathbb{K}_{max}) + \lambda_{rat}}, \quad (3.98)$$

where $\lambda_{rat} = \frac{\lambda_{min}}{1-\lambda_{min}}$, and the minimum limit for the damping factor is also integrated within the recursive form.

- Adaptable damping based on the iteration size ([ADD](#))

This type of damping, defined earlier in (3.82), is based on the iteration size [99, 100] and can be represented as:

$$\lambda_d^{\mathbb{K}} = \alpha_s. \quad (3.99)$$

Finally, Figure 3.9 illustrates a comparison between the first six profiles using a constant damping factor of $\lambda_c = 1.0$, a minimum damping factor of $\lambda_{min} = 0.1$, a step size of $\mathbb{K}_{span} = 100$ iterations, and a maximum iteration limit of $\mathbb{K}_{max} = 2,000$.

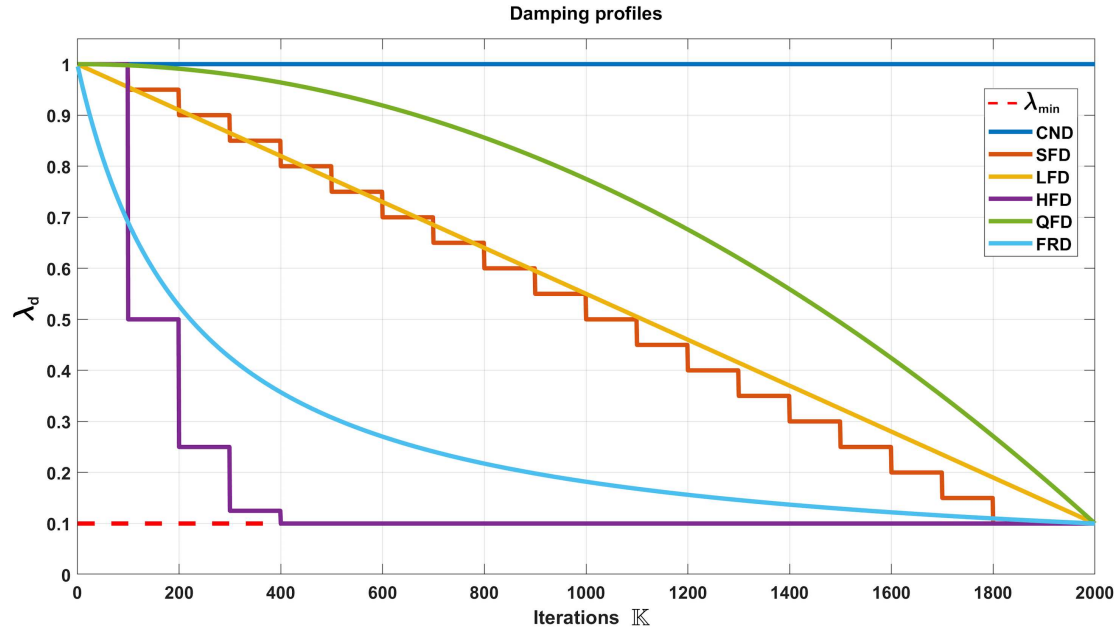


Figure 3.9: Comparison of six damping profiles, illustrating how the damping factor λ_d is either kept constant or reduced in each iteration according to the chosen criteria: $\lambda_c = 1.0$, $\lambda_{min} = 0.1$, $\mathbb{K}_{span} = 100$, and $\mathbb{K}_{max} = 2,000$.

3.6 Simulation Results and Discussion

This section presents the results of simulations conducted to analyse the performance of the IK solver across four different paths with varying complexities. Each path is initially described to highlight its impact on the IK solver. The simulations are organised into four groups: three focusing on the linear approximation and one on the quadratic approximation.

For the linear approximation, the first group of simulations examines the IK of a single-segment actuator without physical limits. These simulations use various combinations of Jacobian algorithms and damping profiles, with differences in results discussed. In the second group, the first path is re-evaluated to simulate the effects of enforcing actuator physical limits.

The third group focuses on the IK of a multi-segment actuator, consisting of two segments and subjected to physical limits. Simulations for each path are conducted, and the differences in results are thoroughly discussed, highlighting various related aspects.

Finally, the fourth group conducts IK simulations for a single-segment actuator using the quadratic approximation across all four paths, without enforcing physical limits. Suggestions for enhancing convergence are provided, and the effects on the IK solver, as well as the differences between linear and quadratic approximations, are discussed.

3.6.1 Inverse Kinematic Paths

Four paths were chosen to study the performance of various Jacobian algorithms in solving the **IK** using the linear approximation. Let us consider a single-segment actuator, $\mathfrak{s} = 1$, and only the position error, $m = 3$, when solving the **IK** for each path. The actuator has a unit length, $L_o = 1$, with its base located at the origin $(0, 0, 0)$, and the endpoint following the chosen paths.

When considering the **NCX** constraints function, defined in (3.24), the optimisation Jacobian $J_g(q)$ is related to the actuator Jacobian $J_f(q)$ via the relation defined in (3.25). Each component of the error vector $g(q)$ is compared with the spatial accuracy ratio SAR at each iteration of the convergence, which continues until all components are equal to or less than SAR or the maximum number of iterations is reached.

Comparatively, when considering the **CVX** constraints function, defined in (3.28), the optimisation Jacobian $J_g(q)$ is related to the actuator Jacobian $J_f(q)$ via the relation defined in (3.30). The error $g(q)$ is compared with the spatial accuracy ratio SAR at each iteration of the convergence, which continues until it is equal to or less than SAR or the maximum number of iterations is reached.

Each path consists of 100 points after the start point, totalling 101 points, and has a unique feature distinguishing it from the other paths. The first path is a horizontal line with no curvature, passing through the actuator's singular, straight, configuration at one of its points, as shown in Figure 3.10(a1).

Contrarily, the second path is a helical curve that passes near the actuator's singular configuration at one of its points, as shown in Figure 3.11(a1). The helix maintains a constant path curvature due to its uniform radius and pitch.

Moreover, the third path is a 3D curve with variable curvature throughout its length, also passing near the actuator's singular configuration at one point, as depicted in Figure 3.12(a1).

Finally, the fourth path is a figure-eight curve, known as the *Lemniscate of Gerono*, combining features from the other three paths. This path has sections with zero, constant, and variable curvatures. It is located near the actuator's base to increase the convergence difficulty, as illustrated in Figure 3.13(a1).

Similar to Figure 3.8, the 3D plots also include the projections of each path on the XY , XZ , and YZ planes for better visualisation. The start point of each path is marked with red circles in the projections. The actuator workspace is not projected since no physical limits were implemented at this stage. Only the first and last configurations of the actuator are plotted in purple to avoid cluttering the 3D plots.

For each path, the manipulability index (ICI) across the entire length is plotted as well, as shown in Figures 3.10(a2), 3.11(a2), 3.12(a2), and 3.13(a2). This index provides a visual illustration of the points that are at or near singular configurations, which can be helpful to understand the performance of the Jacobian algorithms when solving the **IK** at these points.

Each path was solved using all the presented Jacobian algorithms and damping profiles. The overall results are presented below in Section 3.6.2 using tables, which

offer a structured and efficient way to convey large amounts of data, making it easier for readers to quickly grasp the overall trends and comparisons.

However, for the purpose of illustrating a wide scope of differences in the figures, random Jacobian algorithms and damping profiles were chosen to ensure a concise and impactful presentation. The first three paths were solved using **NCX** constraints, while the fourth path was solved using **CVX** constraints.

The **JIV** method along with the **CND** profile were used for the first path. Instead, the **DLS** method along with the **FRD** profile were used for the second path. Conversely, the **JTR** method along with the **LFD** profile were used for the third path. Lastly, the **GRD** method along with the **SFD** profile were used for the fourth path.

The number of iterations required for convergence at each point along each path, as well as the total number of iterations, is illustrated in Figures 3.10(b1), 3.11(b1), 3.12(b1), and 3.13(b1). These figures correspond to the first, second, third, and fourth paths, respectively, based on the selected Jacobian algorithms and damping profiles. The related convergence error for each path, in comparison with a spatial accuracy ratio of $SAR = 10^{-4}$, is plotted in Figures 3.10(b2), 3.11(b2), 3.12(b2), and 3.13(b2), respectively.

Finally, the actuator length L_s and the bending angle θ_s results, with their minimum and maximum limits, are plotted in Figures 3.10(c1, c2), 3.11(c1, c2), 3.12(c1, c2), and 3.13(c1, c2) for the first, second, third, and fourth paths, respectively. The minimum limit for the actuator length L_s was set to one unit, no compression, while the maximum limit was set to two units. Similarly, the maximum limit for the bending angle θ_s was set to $\frac{3}{2}\pi$. However, the **IK** results were allowed to exceed these minimum and maximum limits since no physical constraints were enforced.

3.6.2 Single-Segment Actuator with no Physical Limits

The number of iterations required to solve the **IK** problem using each Jacobian algorithm is detailed in Tables 3.1, 3.2, 3.3, and 3.4 for the first, second, third, and fourth paths, respectively. The results exhibit three distinct types of outputs. Firstly, results are listed in black when convergence is achieved for all points along the path. Secondly, outputs are shown in **red** if the algorithm fails to converge at one or more points along the path. Thirdly, results marked as **NaN** (Not a Number) indicate that the algorithm's convergence was interrupted due to a mathematical error. The optimal algorithm for each path is highlighted in **orange**.

The Jacobian algorithms listed in each table are arranged from left to right based on their ease of implementation, starting with **JIV** as it is the simplest to implement. The algorithms are organised into two categories. The first category consists of six algorithms intended for solving the **IK** problem with **NCX** constraints, whereas the second category comprises three algorithms intended for solving the **IK** problem with **CVX** constraints. Similarly, the damping profiles are listed from top to bottom according to the rate of reduction in the damping factor λ_d , beginning with **CND** as it exhibits no reduction. The **ADD** damping profile is listed at the bottom due to its behaviour not following a specific reduction profile.

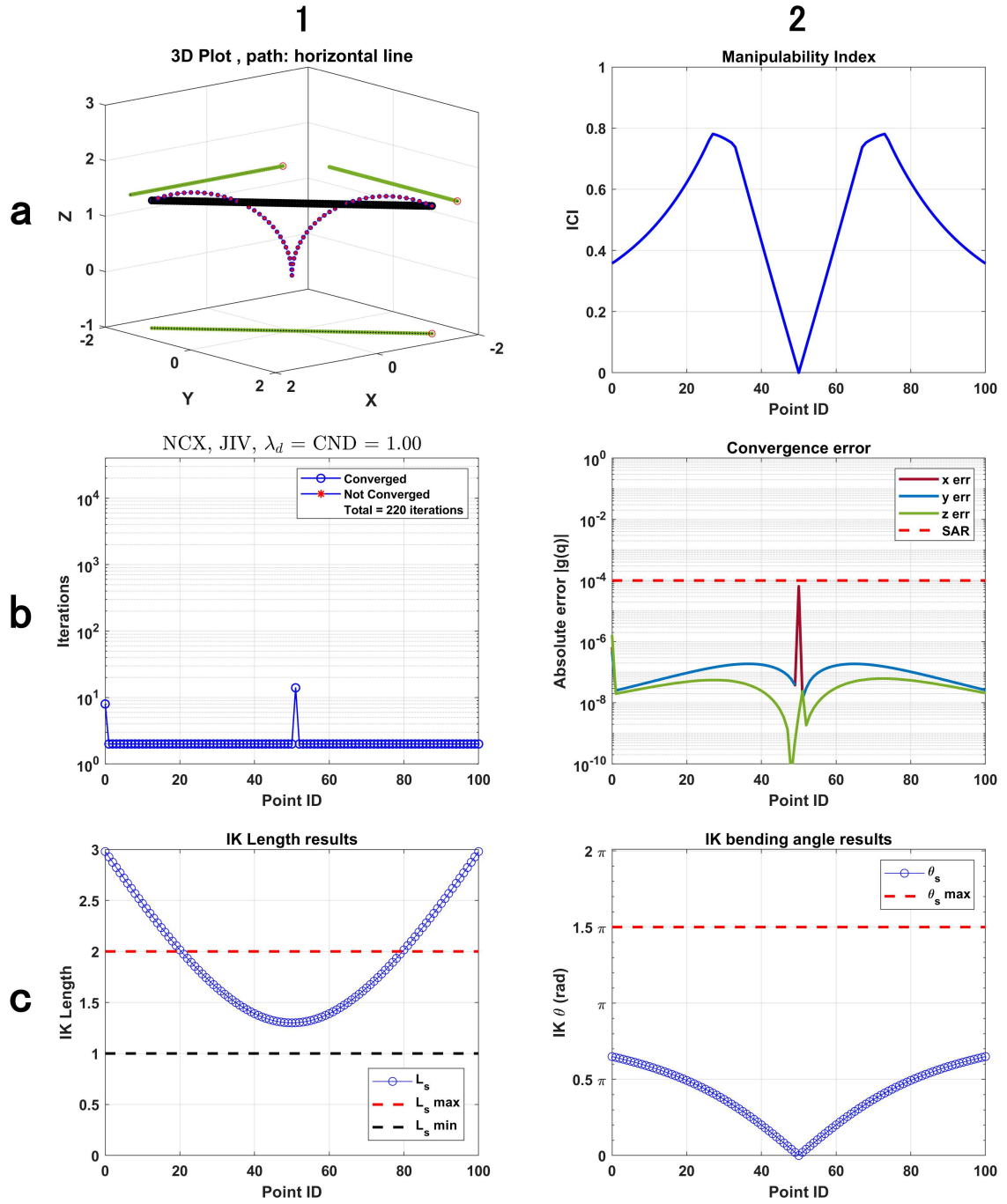


Figure 3.10: IK results for a single-segment actuator following a horizontal line (first path) using the linear approximation without enforcing the actuator's physical limits. (a1) The 3D path and its projections, with the actuator arcs for the first and last points in the path marked in purple colour. (a2) The manipulability index. (b1) The convergence iterations for each point in the path using NCX constraints, the JIV method, and the CND profile. All points converged. (b2) The components of the convergence error vector at each point. All points have an error less than SAR. The results for the length L_s and bending angle θ_s are shown in (c1) and (c2), respectively.

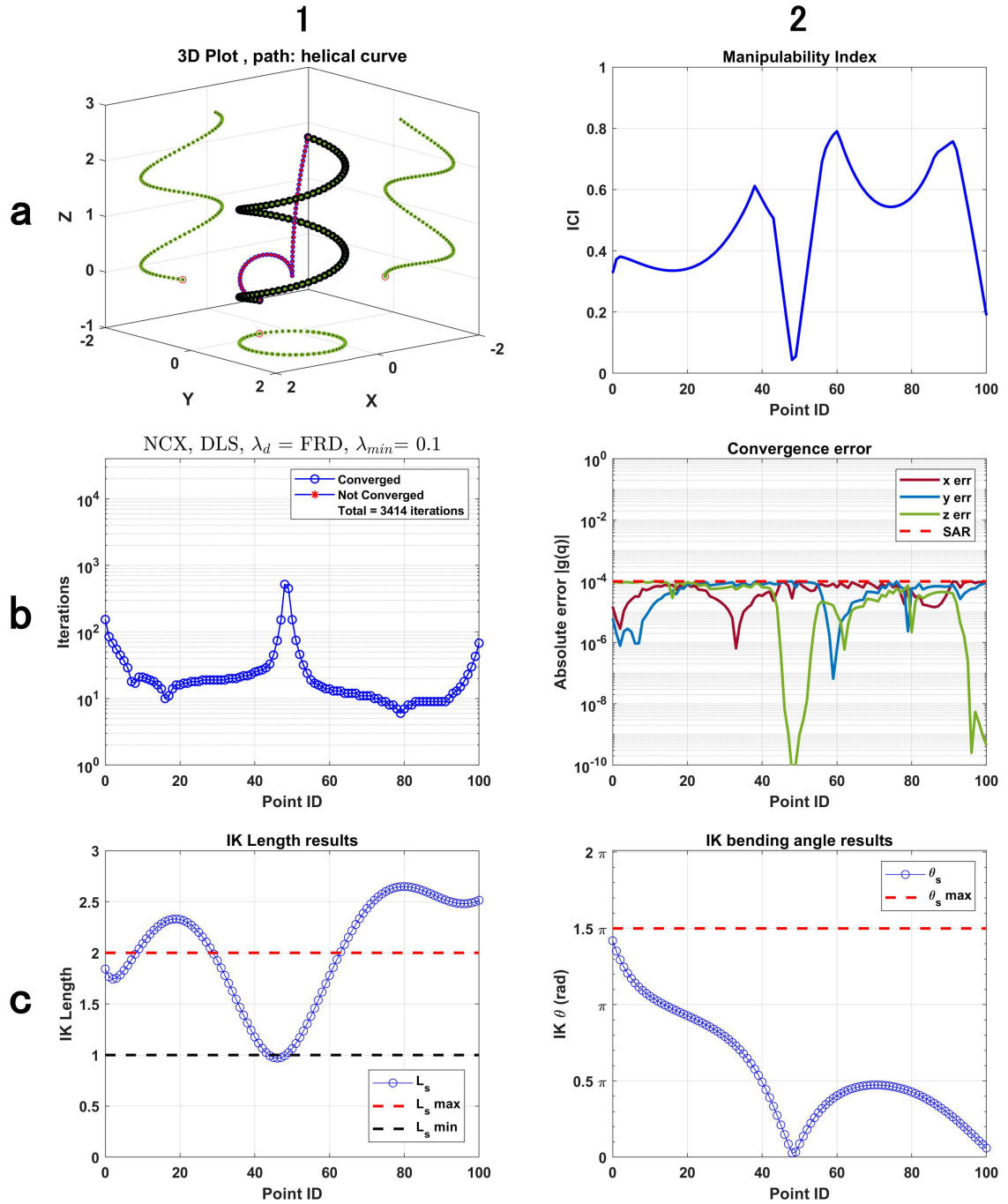


Figure 3.11: IK results for a single-segment actuator following a helical curve (second path) using the linear approximation without enforcing the actuator's physical limits. (a1) The 3D path and its projections, with the actuator arcs for the first and last points in the path marked in purple colour. (a2) The manipulability index. (b1) The convergence iterations for each point in the path using NCX constraints, the DLS method, and the FRD profile. All points converged. (b2) The components of the convergence error vector at each point. All points have an error less than or equal to SAR. The results for the length L_s and bending angle θ_s are shown in (c1) and (c2), respectively.

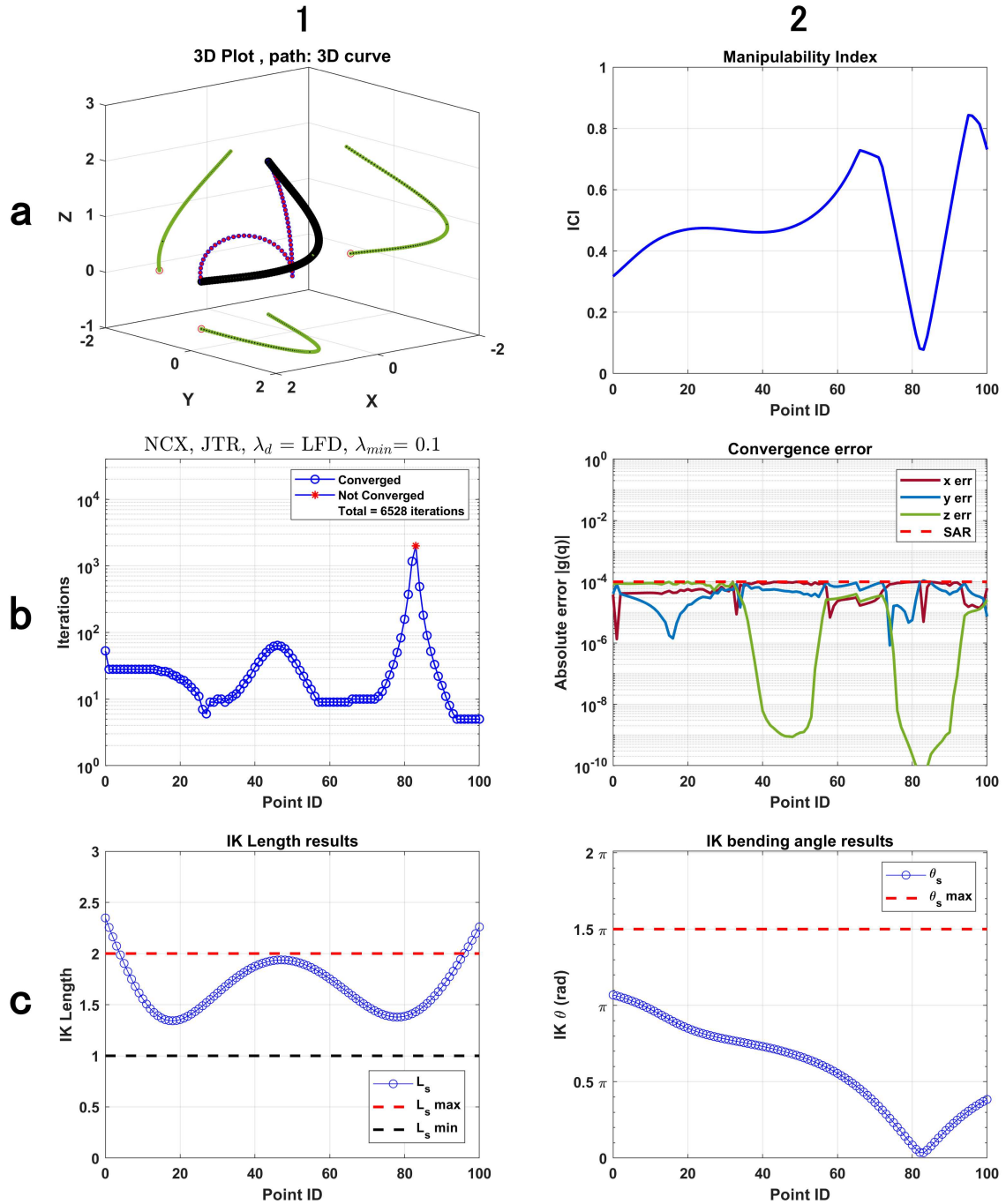


Figure 3.12: IK results for a single-segment actuator following a 3D curve (third path) using the linear approximation without enforcing the actuator's physical limits. (a1) The 3D path and its projections, with the actuator arcs for the first and last points in the path marked in purple colour. (a2) The manipulability index. (b1) The convergence iterations for each point in the path using NCX constraints, the JTR method, and the LFD profile. All points converged except for one point that reached the iteration limit K_{max} . (b2) The components of the convergence error vector at each point. All points have an error less than or equal to SAR except for one, where the y component of the error was slightly larger than SAR. The results for the length L_s and bending angle θ_s are shown in (c1) and (c2), respectively.

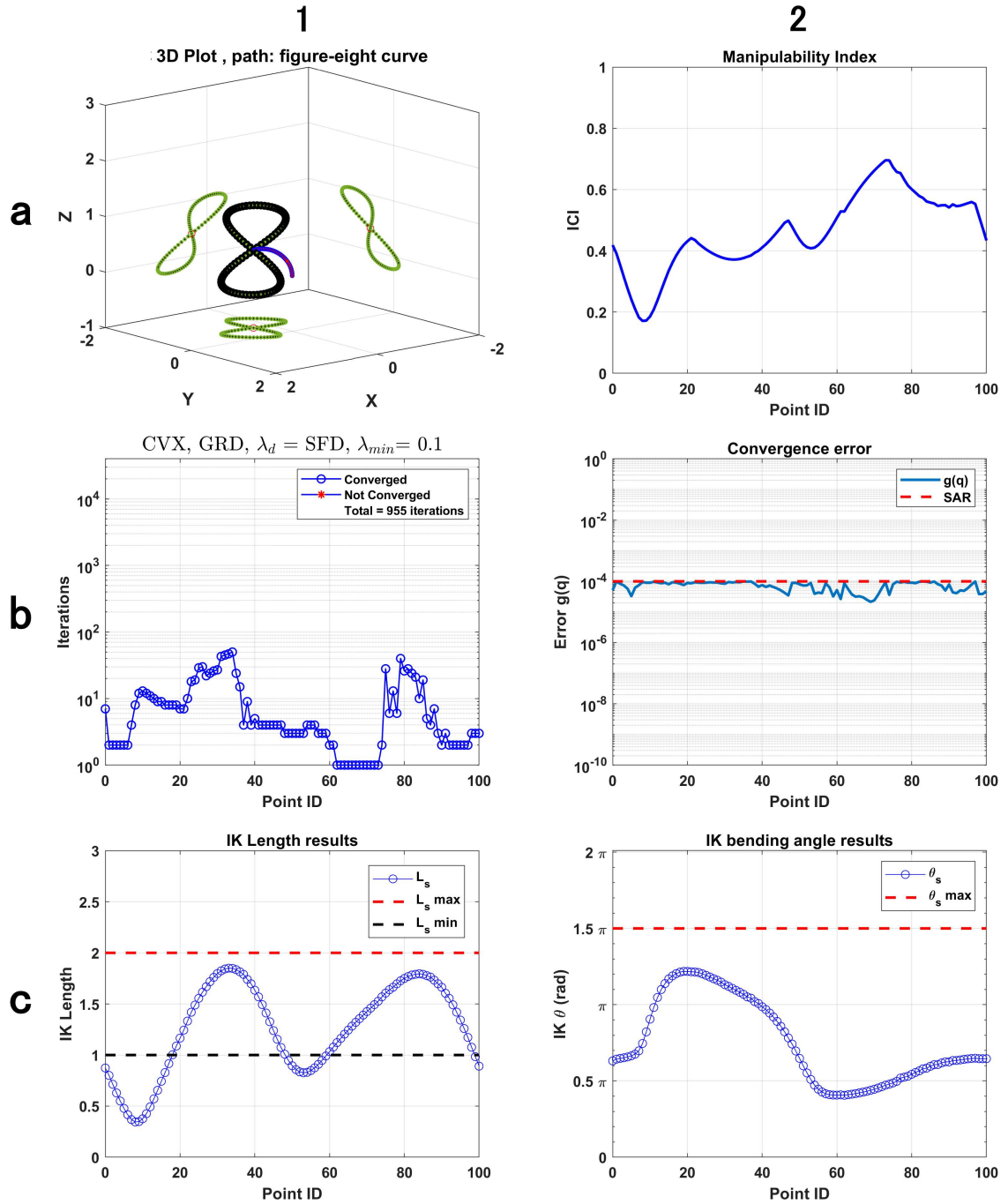


Figure 3.13: IK results for a single-segment actuator following a figure-eight curve (fourth path) using the linear approximation without enforcing the actuator's physical limits. (a1) The 3D path and its projections. The actuator arcs for the first and last points, which have similar coordinates, in the path are marked in purple colour. (a2) The manipulability index. (b1) The convergence iterations for each point in the path using CVX constraints, the GRD method, and the SFD profile. All points converged. (b2) The convergence error at each point. All points have an error less than or equal to SAR. The results for the length L_s and bending angle θ_s are shown in (c1) and (c2), respectively.

In each table, two **CND** values are listed: one with $\lambda_c = 1$, indicating no damping, and the other with varying values specific to each table, which will be discussed further below. For the remaining damping profiles except **ADD**, the initial damping factor was set to one, with a minimum damping factor of $\lambda_{min} = 0.1$. Moreover, the convergence was assessed with a maximum iteration limit of $\mathbb{K}max = 2,000$, and the step size for both **SFD** and **HFD** was $\mathbb{K}span = 100$ iterations. Consequently, the total number of iterations for each path can reach up to 202,000 since each path consists of 100 points in addition to the start point.

3.6.2.1 First Path: Horizontal Line

The results in Table 3.1 indicate that almost all algorithms were able to converge, given the path's low complexity. The **JIV** algorithm with **NCX** constraints appears to be the optimal choice due to its simplicity and its ability to converge without damping, $\lambda_c = 1$. On average, each point in the path converged within just two iterations, apart from the first and middle points, as shown in Figure 3.10(b1). Additional iterations were required for the first point because the actuator had to move a certain distance from the initial configuration. Subsequent points in the path required significantly less movement from the previous point.

For the middle point, additional iterations were necessary because the actuator was at a singular position, as clearly indicated in the manipulability index plot in Figure 3.10(a2). The concept of minimum bending angle, discussed in Section 3.1.3, proved effective in avoiding a singular Jacobian and allowing the **JIV** method to converge even at this singular configuration. The **JPI** and **SVD** algorithms produced results consistent with those of **JIV**, as they also provided the Jacobian inverse.

The **ADD** damping failed to provide a solution for the **JIV** algorithm because the damping factor λ_d was found to exceed one. This led to overshooting due to a larger step size and caused the Jacobian determinant to become zero, resulting in a mathematical error. In contrast, the **SVD** method, known for its robustness, avoided mathematical errors but failed to achieve convergence at any point along the path.

Both **DLS** and **PDL** exhibited identical results, as **PDL** is an extension of **DLS**. Nevertheless, both methods demonstrated a lower convergence rate compared to **JIV**, attributable to the structure of these algorithms in implementing damping. Additionally, the **JTR** algorithm showed the lowest convergence rate among all the algorithms, a result consistent with its known slow convergence characteristics. This slow performance was fairly mitigated by using **ADD** damping, which significantly improved the **JTR** algorithm's performance. Such behaviour is consistent with the literature, where **ADD** is recognised as the optimal option for **JTR** [100].

For the **CVX** constraints, both the **JPI** and **SVD** algorithms exhibited similar results, though they were slower compared to the **JIV** algorithm with **NCX** constraints. The cause of this behaviour has not been fully understood and could be a focus for future studies. Likewise, the **GRD** algorithm showed a slower convergence rate compared to the **JIV** algorithm with **NCX** constraints, as well as both **JPI** and **SVD**, due to the generally slower nature of gradient descent-based methods compared to Newton's method.

Lastly, the damping factor was manually decreased until **JTR** algorithm successfully achieved convergence at all points along the path. Specifically, convergence was attained when the damping factor reached $\lambda_c = 0.44$. All the Jacobian algorithms were tested, and the results indicate that both **DLS** and **PDL** exhibited a better convergence rate compared to the other algorithms. This improved performance is believed to be related to the structure of these two algorithms, where constant damping provides superior results.

Table 3.1: Results of **IK** for the first path (horizontal line) without enforcement of physical limits.

Hor. Line	Jacobian Algorithm								
Damping	JIV	JTR	JPI	DLS	SVD	PDL	GRD (CVX)	JPI (CVX)	SVD (CVX)
CND (0.44)	1,018	2,187	1,018	482	1,018	482	920	619	619
CND (1.00)	220	52,775	220	1,381	220	1,381	376	252	252
SFD	220	24,268	220	1,381	220	1,381	376	252	252
LFD	224	25,423	224	1,374	224	1,374	376	252	252
HFD	220	2,980	220	1,381	220	1,381	376	252	252
QFD	220	36,102	220	1,381	220	1,381	376	252	252
FRD	220	6,064	220	1,329	220	1,329	376	258	258
ADD	NaN	356	NaN	4,133	202,000	4,133	11,909	202,000	202,000

3.6.2.2 Second Path: Helical Curve

With higher path complexity, some algorithms failed to provide **IK** solutions due to numerical instability, as seen in the results in Table 3.2. For both **JIV** and **JPI** algorithms, the Jacobian values decrease with iterations due to overshooting, which causes the determinant to approach zero and eventually leads to mathematical errors. The decreasing Jacobian values correspond to diminishing effectiveness in capturing the rate of change of the configuration space q variables, which impacts the stability of the **IK** solution.

In contrast, the **SVD** algorithm successfully converged, owing to its robustness. The choice of damping profile for **SVD** appears to be less critical, as most points along the path converge with just two iterations, similar to the results observed for the first path. Therefore, **SVD** with no damping, $\lambda_c = 1$, emerges as the optimum choice.

Again, both **DLS** and **PDL** showed identical results and exhibited a lower convergence rate compared to **SVD**. This performance difference can be attributed to the structure of these algorithms in implementing damping, which explains why they were unable to converge at one of the points in the path, particularly near the actuator's singularity, when no damping was applied. However, with the use of any damping profile except **ADD**, both **DLS** and **PDL** were able to converge, despite the numerical instability near the singular configuration, as illustrated in Figure 3.11(b1).

On the other hand, the **JTR** algorithm demonstrated very poor performance despite its robustness to mathematical errors, given that no Jacobian inversion is required. As observed in the first path, **ADD** damping significantly improved the stability and convergence rate of **JTR**, while negatively impacting the performance of other algorithms.

In regards to the **CVX** constraints, both the **JPI** and **SVD** algorithms exhibited nearly identical results, though they were slower compared to the **SVD** algorithm with **NCX** constraints. A slight improvement was observed in both **JPI** and **SVD** algorithms when the **FRD** profile was applied. This behaviour suggests potential advantages of this damping profile and warrants further investigation in future studies. Conversely, the convergence rate of the **GRD** algorithm deteriorated significantly with increased path complexity, highlighting the benefits of using Newton's method for solving the **IK** problem in **SPAs**.

Lastly, the damping factor was manually reduced until most algorithms successfully converged at all points along the path. This outcome was achieved with a damping factor of $\lambda_c = 0.26$, which was determined through iterative adjustment to optimise convergence. As observed with the first path, both **DLS** and **PDL** showed improved convergence rates compared to other algorithms, attributed to their effective performance with constant damping. The **JTR** algorithm, however, failed to converge again, reinforcing its unsuitability for this application.

Table 3.2: Results of **IK** for the second path (helical curve) without enforcement of physical limits.

Helical	Jacobian Algorithm								
Damping	JIV	JTR	JPI	DLS	SVD	PDL	GRD (CVX)	JPI (CVX)	SVD (CVX)
CND (0.26)	2,314	14,356	2,314	764	2,314	764	5,051	1,670	1,672
CND (1.00)	NaN	70,838	NaN	7,044	225	7,044	1,658	508	514
SFD	NaN	20,611	NaN	5,226	225	5,226	1,687	508	514
LFD	NaN	20,151	NaN	5,216	227	5,216	1,689	503	503
HFD	NaN	13,016	NaN	3,285	225	3,285	3,700	508	514
QFD	NaN	35,242	NaN	5,983	225	5,983	1,653	517	517
FRD	NaN	9,878	NaN	3,414	232	3,414	2,167	496	496
ADD	NaN	3,350	NaN	43,089	202,000	43,089	11,037	202,000	202,000

3.6.2.3 Third Path: 3D Curve

Although this path introduces challenges related to variable path curvature in the **IK** process, it does not exhibit significantly higher complexity compared to the previous paths. In comparison to the second path, the manipulability index ICI appears to be slightly improved. This improvement may be attributed to the path geometry and its positioning relative to the actuator base. This difference is evident in the results presented in Table 3.3 when compared with the results for the second path shown in Table 3.2, particularly for the same combinations of Jacobian algorithms and damping profiles.

The **JIV**, **JPI**, and **SVD** algorithms demonstrated performance comparable to that observed in the second path. However, the choice of damping profile notably affected the **SVD** algorithm. The **FRD** profile emerged as the most effective damping option, underscoring its potential. Nevertheless, the difference between the damped and undamped **SVD** results was minimal, suggesting that the undamped **SVD** may be the preferable choice for this path.

Both **DLS** and **PDL** algorithms once again yielded identical results, demonstrating lower convergence rates compared to the **SVD** algorithm. Analysing the results of these two algorithms for solving the **IK** for the second and third paths confirms the observed reduced complexity of the third path. Consequently, the **JTR** algorithm exhibited marginally improved performance, although it failed to converge with most damping profiles due to oscillations near singular configurations, as illustrated in Figure 3.12(b1).

Notably, the algorithm was able to achieve convergence at all points along the path and with all damping profiles when the convergence iteration limit was increased to $\mathbb{K}_{max} = 10,000$ iterations. This increment had two significant implications: it permitted additional iterations to facilitate convergence, and it affected the damping factor value during the iterations, given that $\mathbb{K}_{max}$ is utilised to calculate $\lambda_d^{\mathbb{K}}$ in most damping profiles.

Regarding the **CVX** constraints, both the **JPI** and **SVD** algorithms exhibited nearly identical results, consistent with their behaviour observed in the second path, though slower than the **SVD** algorithm with **NCX** constraints. However, the **FRD** profile did not yield significant enhancement for either of these algorithms.

The **GRD** algorithm, in contrast, demonstrated distinct performance differences in comparison to the second path. On one hand, it achieved faster convergence with several damping profiles due to the reduced complexity of the path. On the other hand, it failed to converge when the **HFD** damping was applied. The slower convergence rate of this algorithm is believed to be the underlying reason for this failure. This hypothesis was substantiated by extending the iteration limit beyond the $\mathbb{K}_{max} = 2,000$ iterations, which subsequently resulted in successful convergence.

Table 3.3: Results of **IK** for the third path (3D curve) without enforcement of physical limits.

3D Curve	Jacobian Algorithm								
Damping	JIV	JTR	JPI	DLS	SVD	PDL	GRD (CVX)	JPI (CVX)	SVD (CVX)
CND (0.65)	657	7,242	657	2,338	657	2,338	1,511	601	606
CND (1.00)	NaN	5,437	NaN	5,140	228	5,140	1,257	453	447
SFD	NaN	6,721	NaN	4,417	228	4,417	1,300	453	447
LFD	NaN	6,528	NaN	4,361	219	4,361	1,292	448	445
HFD	NaN	10,823	NaN	3,081	228	3,081	3,369	453	447
QFD	NaN	5,651	NaN	4,877	226	4,877	1,263	442	451
FRD	NaN	8,662	NaN	3,076	218	3,076	2,037	446	446
ADD	NaN	2,381	NaN	43,115	202,000	43,115	202,000	202,000	202,000

Lastly, the damping factor was manually reduced until all algorithms successfully achieved convergence at every point along the path. This result was attained with a damping factor of $\lambda_c = 0.65$, determined through repeated adjustments to optimise convergence. The **CVX** constraints function was observed to achieve faster convergence compared to the **NCX** constraints, despite the latter's lower computational complexity.

Additionally, both **DLS** and **PDL** demonstrated significantly slower convergence rates in the current scenario (manual reduction of λ_d) relative to their performance on the second path. This reduction in performance is attributed to the relatively smaller decrease in the damping factor λ_d for the current path, in contrast to the more substantial reduction to 0.26 applied in the second path. The larger reduction in damping factor in the second path facilitated more substantial updates by the algorithms, provided stability was maintained, which contributed to faster convergence.

3.6.2.4 Fourth Path: Figure-Eight Curve

This path represents the highest level of complexity among the examined paths, attributable to its intricate geometry and spatial positioning, as evidenced by the data in Table 3.4. Although the **SVD** algorithm was previously validated as the optimal choice for the first three paths, it encountered convergence issues on this path due to numerical instability. This occurred even when the iteration limit for convergence was extended to $\mathbb{K}_{max} = 10,000$ iterations. The suboptimal performance of the **SVD** algorithm is attributed to its convergence towards a saddle point rather than a local or global minimum.

In this context, the utilisation of **CVX** constraints demonstrated superior efficacy, achieving convergence across all points along the path, irrespective of the algorithm or damping profile employed, with the sole exception of the **ADD**, as illustrated in Figure 3.13(b1). Both the **JPI** and **SVD** algorithms exhibited nearly identical results, outperforming all other algorithms. The **SVD** algorithm continues to be the preferred choice, given its enhanced robustness in addressing Jacobian-related issues compared to the **JPI** algorithm.

Returning to the **NCX** constraints, the **JTR** algorithm demonstrated superior performance compared to all other paths, despite the increased path complexity. This behaviour is attributed to the inherent characteristics of this algorithm, wherein the updates are typically smaller and more incremental. Consequently, this gradual approach enables the algorithm to smoothly follow the path without making abrupt jumps. Such a smooth progression is essential for navigating complex paths with sharp turns or loops, exemplified by the figure-eight curve.

Conversely, for paths characterised by frequent changes in direction and curvature, the direct least-squares solution provided by the **SVD** algorithm may not necessarily be the most suitable choice at each step. The algorithm might either overshoot or undershoot the desired position, particularly when the path's complexity induces rapid variations in the Jacobian. Nevertheless, both the **DLS** and **PDL** algorithms managed to achieve convergence, albeit with significantly slower convergence rates. The selection of the damping profile also has a substantial influence on performance.

Lastly, focusing on the **CVX** constraints, the damping factor was manually adjusted to achieve the minimum number of iterations required for the **GRD** algorithm, thus attaining the fastest convergence rate. All algorithms achieved convergence at all points along the path. This outcome was attained with a damping factor of $\lambda_c = 0.50$, determined via iterative adjustments to optimise convergence. This investigation aimed to assess the efficacy of Newton's method, which once again demonstrated its superiority.

Both the **JPI** and **SVD** algorithms achieved convergence with the fewest number of iterations compared to all other algorithms. Therefore, for the linear approximation of the Taylor series, a **CVX** constraints function solved using Newton's method via the **SVD** algorithm is deemed the optimal choice for solving the **IK** of **SPAs** across all four paths examined. The impact of redundancy, multi-segment actuators, and the enforcement of physical limits will be discussed in the subsequent sections.

Table 3.4: Results of **IK** for the fourth path (figure-eight curve) without enforcement of physical limits.

Eight	Jacobian Algorithm								
Damping	JIV	JTR	JPI	DLS	SVD	PDL	GRD (CVX)	JPI (CVX)	SVD (CVX)
CND (0.50)	945	10,366	945	1,602	945	1,602	775	650	650
CND (1.00)	NaN	5,082	NaN	5,580	22,217	5,580	955	363	364
SFD	NaN	5,204	NaN	5,381	22,217	5,381	955	363	364
LFD	NaN	5,280	NaN	5,259	22,326	5,259	919	361	361
HFD	NaN	12,857	NaN	4,427	22,217	4,427	955	363	364
QFD	NaN	5,095	NaN	5,551	22,233	5,551	964	366	366
FRD	NaN	7,773	NaN	4,042	22,281	4,042	784	360	360
ADD	NaN	1,356	NaN	95,714	202,000	95,714	39,610	202,000	202,000

3.6.3 Single-Segment Actuator Subjected to Physical Limits

Thus far, all results presented in Section 3.6.2 were obtained without enforcing physical limits. However, real-life applications involve actuators with design-imposed physical constraints. The **IK** solver must operate within these limits, potentially leading to convergence issues if the target pose lies outside the actuator's workspace.

Reconsidering the first path from Section 3.6.2.1, let us focus on algorithms with slower convergence rates for comparative purposes. The **JTR** algorithm, combined with the **FRD** damping profile, was selected randomly. The minimum and maximum limits for both the segment length L_s and bending angle θ_s were enforced, and the **IK** solver results are depicted in Figure 3.14(a1 to c2).

Based on the actuator workspace projections shown in Figure 3.14(a1), 20 points at each end of the path lie outside the workspace. Consequently, convergence was not achieved at these points, as depicted Figure 3.14(b1), as the actuator length reached its maximum limit, as shown in Figure 3.14(c1). The iterations continued at each

point until reaching the termination limit $\mathbb{K}_{max} = 2,000$, as the convergence error remained greater than SAR as can be seen in Figure 3.14(b2).

The **IK** solver is incapable of further reducing this error due to the imposed physical limits. Nevertheless, the solver's performance can be improved by incorporating a mechanism to terminate iterations once no additional reduction in error is observed. Implementing this enhancement may result in a decrease in the total number of iterations required to track all points along the path, including those situated outside the workspace.

The endpoint location p_{cs} can be systematically monitored during the iterations by comparing it with the previous iterations over a predefined window, such as two iterations. The iterative process can be terminated when the error difference falls below a specified threshold, which can be set as a percentage of SAR for simplicity. A recommended threshold of 0.1% SAR can be utilised to ensure sufficiently precise results before halting the iterations.

Moreover, depending on the selected algorithm, convergence may be impeded by oscillations, presenting a significant challenge to the implementation of the error monitoring option. This issue can be resolved by employing the moving average technique [335], widely utilised in statistics to smooth noisy data. This technique can be implemented in MATLAB using the `movmean()` function.

The **IK** results utilising the error monitoring option are depicted in Figure 3.15(a1 to c2). The reduction in total iterations is substantial, as all unnecessary iterations were eliminated. Moreover, the point with the highest iteration count along the path, identified as point ID = 0, is selected for an in-depth examination.

Figure 3.16(a1) presents a 3D map illustrating the convergence error at the selected point. The error Euclidean norm map is calculated via **FK** across the entire range of segment length L_s and bending angle θ_s . Additionally, a vertical red-coloured line is positioned at the L_s and θ_s values obtained from the **IK** results, as shown in Figure 3.15(c1, c2) at point ID = 0.

Additionally, the 3D map is represented as a contour plot in Figure 3.16(a2) to facilitate improved visualisation. Both the 3D map and the contour plot confirm that the **IK** results correspond to the minimum achievable error at that point, thereby demonstrating the efficacy of the error monitoring option in eliminating redundant iterations upon reaching the minimal error threshold.

The convergence of the endpoint coordinates p_{cs} is depicted in Figure 3.16(b1), where the coordinate values stabilised at approximately 150 iterations and continued to converge until the 0.1% SAR threshold was satisfied. The oscillation in the results was significantly mitigated, as shown in Figure 3.16(b2), through the application of the moving average technique with a window of two iterations, as illustrated in Figure 3.16(c). The size of the error monitoring window can be increased to ensure greater stability in the **IK** solver if the convergence oscillation is substantial. However, a larger window may introduce additional iterations, dependent on the magnitude of the oscillation.

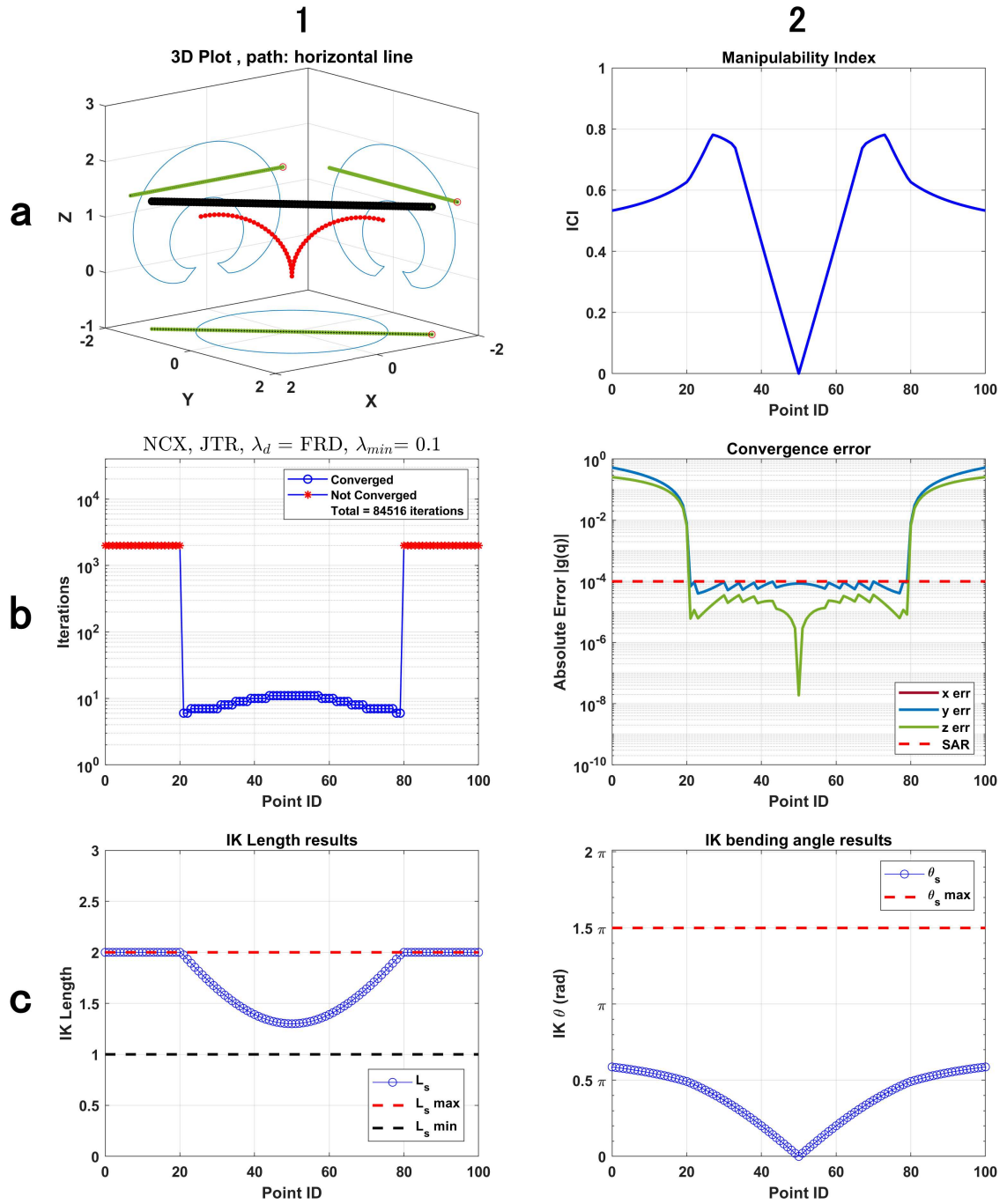


Figure 3.14: IK results for a single-segment actuator following a horizontal line (first path) using linear approximation with the enforcement of the actuator physical limits. (a1) The 3D path and the actuator workspace are projected onto Cartesian planes. Actuator arcs at the first and last points of the path are highlighted in red, reflecting their convergence status. (a2) The manipulability index. (b1) Convergence iterations for each point along the path using NCX constraints, JTR method, and FRD profile. Points outside the actuator workspace do not achieve convergence. (b2) Components of the convergence error vector at each point, with non-converging points exhibiting error exceeding SAR. Results for the length L_s and bending angle θ_s are depicted in (c1) and (c2), respectively, indicating that points with no convergence reached the maximum length limit.

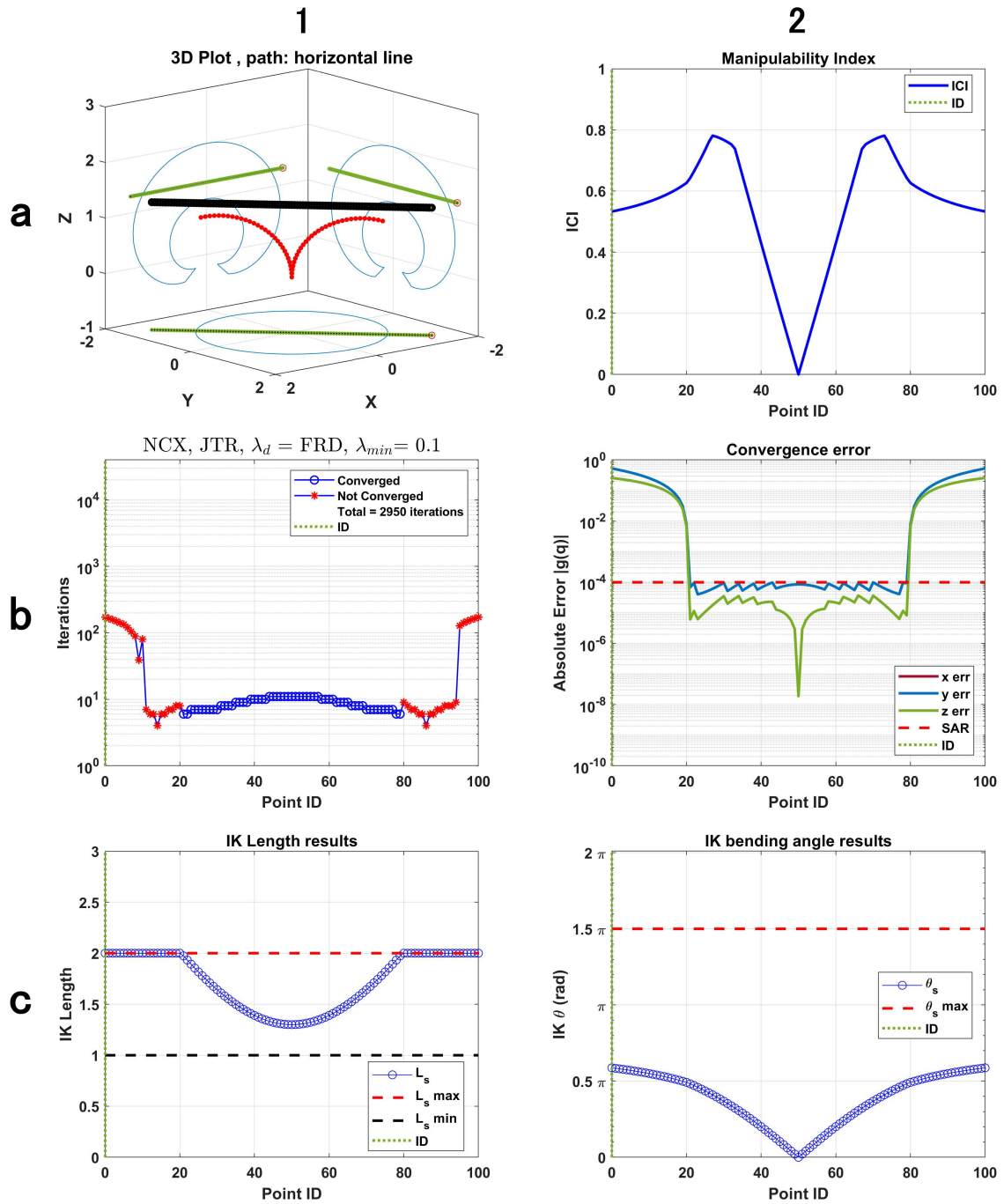


Figure 3.15: IK results for a single-segment actuator following a horizontal line (first path) using linear approximation with the enforcement of the actuator physical limits. (a1) 3D path and the actuator workspace are projected onto Cartesian planes. Actuator arcs at the first and last points of the path are highlighted in red, indicating their convergence status. (a2) The manipulability index. (b1) Convergence iterations for each point along the path using NCX constraints, JTR method, and FRD profile. The implementation of the error monitoring option significantly reduced the number of iterations required for points with no convergence. (b2) Components of the convergence error vector at each point, with non-converging points exhibiting errors exceeding SAR. Results for the length L_s and bending angle θ_s are depicted in (c1) and (c2), respectively, indicating that points with no convergence reached the maximum length limit.

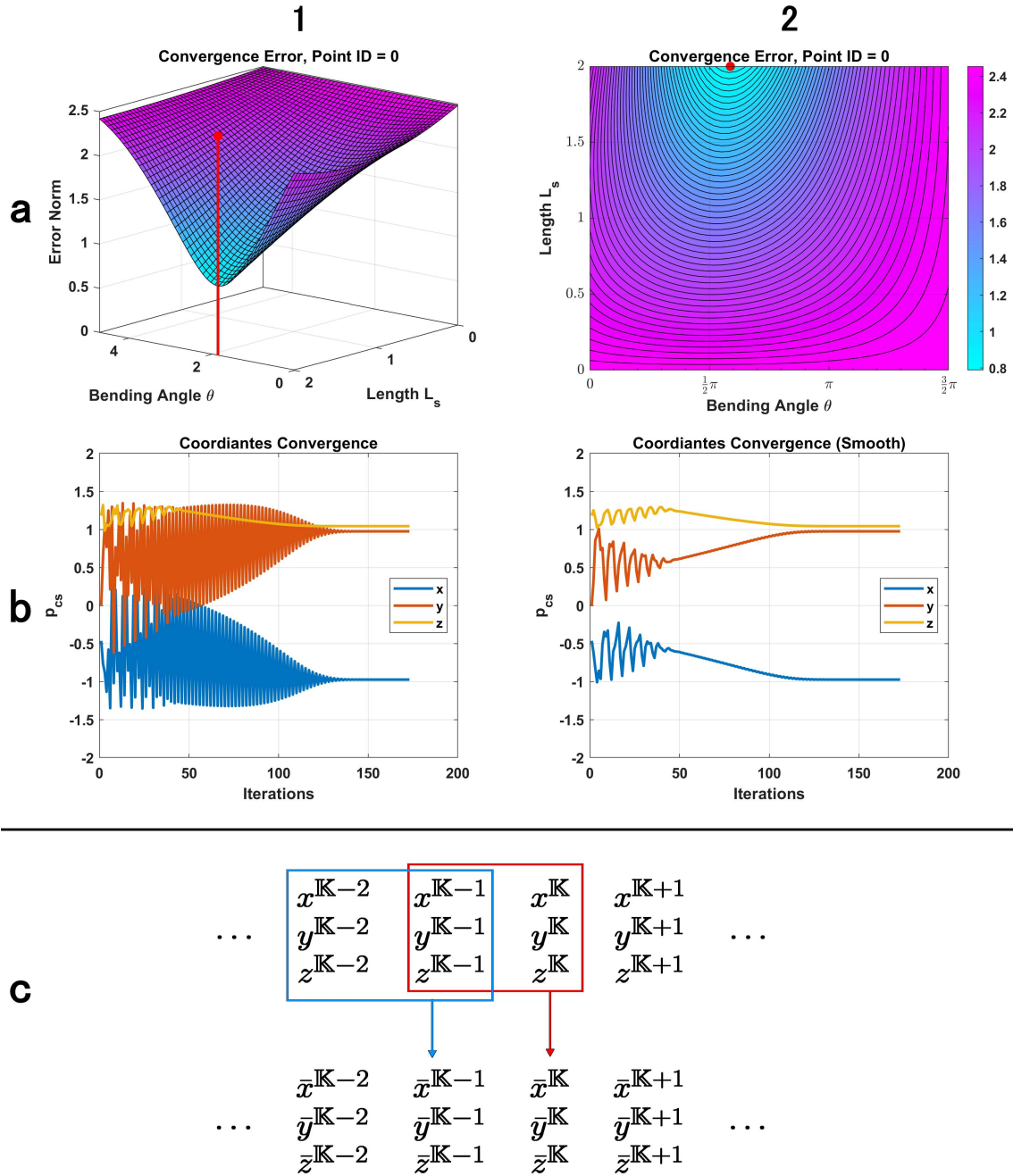


Figure 3.16: Convergence analysis of the IK results for a single-segment actuator following a horizontal line (first path) using linear approximation with the enforcement of the actuator physical limits. (a1) The 3D plot depicting convergence error at point ID = 0. The length L_s and bending angle θ_s values obtained from the IK results are indicated by a vertical red line. (a2) Contour plot of the convergence error for the same point ID, with IK results marked by a red dot at the minimum convergence error. Raw and smoothed results for the convergence of the end point p_{cs} coordinates at the same point ID are shown in (b1) and (b2), respectively. (c) Illustration of the moving average technique with a window of two iterations to smooth oscillations in the end point p_{cs} coordinates.

3.6.4 Multi-Segment Actuator Subjected to Physical Limits

Despite the effectiveness of the [IK](#) solver in minimising the error, the desired pose p_d remains unreachable when an actuator reaches its physical limits. To address this limitation, a single-segment actuator may either be redesigned with an increased length, if feasible, or additional segments can be incorporated to augment the total length and the number of DOF_k in the kinematic chain.

Consider the first path as an example. A single-segment actuator required a length of 3 units to reach the outer points of the path when physical limits were not enforced, as illustrated in Figure 3.10(c1). This length is unattainable when physical limits are enforced, as clearly demonstrated in Figure 3.15(c1). By adding one more segment, the actuator gains additional physical length and redundancy, thereby enabling it to reach the outer points if only the position error is considered, with $DOF_k = 6$ compared to $DOF_m = 3$. The results presented in Table 3.1 were replicated using a multi-segment actuator comprising two segments, $\mathfrak{s} = 2$, with physical limits enforced. The error monitoring option was employed during the convergence process to eliminate unnecessary iterations. The updated results are detailed in Table 3.5 below.

Table 3.5: [IK](#) results for a multi-segment actuator (two segments) following the first path (horizontal line) with the enforcement of physical limits. The error monitoring option was utilised during the convergence process.

Hor. Line	Jacobian Algorithm								
Damping	JIV	JTR	JPI	DLS	SVD	PDL	GRD (CVX)	JPI (CVX)	SVD (CVX)
CND (0.40)	NaN	28,275	36,703	5,366	36,703	5,366	1,150	12,973	12,960
CND (1.00)	NaN	42,545	1,147	26,413	1,147	26,115	22,149	743	743
SFD	NaN	27,309	1,147	23,707	1,147	23,707	21,805	743	743
LFD	NaN	29,888	1,138	34,723	1,138	34,723	21,723	747	747
HFD	NaN	29,150	1,147	18,656	1,147	18,656	25,693	743	743
QFD	NaN	38,674	1,147	27,359	1,147	27,359	22,040	743	743
FRD	NaN	8,696	1,084	31,193	1,084	4,232	23,015	785	785
ADD	NaN	2,853	2,764	39,107	2,764	39,107	14,776	192,730	190,667

The performance of various Jacobian algorithms demonstrated significant differences when using [NCX](#) constraints to solve the [IK](#) problem of a multi-segment actuator, compared to a single-segment actuator. Given the size of the error vector $g(q)$ and the number of segments used to generate the results in Table 3.5, the optimisation Jacobian $J_g(q)$ is no longer a square matrix. Consequently, the [JIV](#) method failed to produce any results. In contrast, both the [JPI](#) and [SVD](#) methods successfully managed the non-square nature of the optimisation Jacobian $J_g(q)$. The convergence outcomes for both methods were identical, demonstrating their robustness in solving the [IK](#) problem irrespective of the damping profile, with the exception of the [ADD](#) profile.

Additionally, the [JTR](#), [DLS](#), and [PDL](#) methods effectively handled the non-square size of the optimisation Jacobian $J_g(q)$ because of their mathematical structure.

Nevertheless, they failed to converge with most damping profiles, as the convergence reached a plateau in the error function, where further iterations did not yield any improvements. Despite this, some exceptions to this lack of convergence were observed, such as the **JTR** method with the **ADD** profile and the **PDL** method with the **FRD** profile. These findings present opportunities for future investigations.

Using **CVX** constraints, the behaviour observed was markedly different from that of the single-segment actuator. Specifically, the **GRD** method failed to converge across all damping profiles, with the exception of **ADD**. Further examination revealed that convergence did not occur at several intermediate points along the path, as shown in Figure 3.17(b1). Notably, a point with no convergence, point ID = 50 and located at $x = y = 0$ in the middle of the path, was selected for further analysis. Figure 3.17(a1) displays three actuator arcs, with the middle arc failing to reach the designated point. Despite having adequate room for manipulability, as shown in Figure 3.17(a2), the convergence error at this point was substantial, as depicted in Figure 3.17(b2).

The second segment was constrained to its minimum length while its bending angle was at the maximum, as depicted in Figure 3.17(c1, c2). This configuration suggests that the solver has approached a region where no further improvements are feasible, even when using **CVX** constraints. Both the convergence error mesh and contour plots for the second segment, shown in Figure 3.18(a1, a2), indicate that a segment length smaller than the physical minimum limit would be required to achieve the optimal actuator configuration.

This behaviour is attributed to the **GRD** method rather than the actuator itself, as the singularity issue was addressed through the minimum bending limit concept discussed earlier in Section 3.1.3. The **IK** solver attempted to push the first segment towards a near-singular position, as shown in Figure 3.17(a1). At this state, the **GRD** method exhibited oscillations near the singularity, which hindered convergence to an optimal solution. The rapid oscillation in the end point p_c coordinates is evident in Figure 3.18(b1), where the smoothing technique failed to reduce oscillations below the error monitoring threshold of 0.1% *SAR*, resulting in continued iterations until being terminated at $\mathbb{K}_{max}$.

Regardless, both the **JPI** and **SVD** methods demonstrated rapid convergence due to their proficiency in managing non-square Jacobians. However, compared to the single-segment actuator results presented in Table 3.1, the number of iterations was approximately three times greater for the multi-segment actuator, rather than the anticipated two-fold increase. Further investigation is required to enhance their convergence speed, especially for applications involving real-time control. Nonetheless, the **CVX** constraints have proven to be substantially more effective than **NCX** constraints in solving the multi-segment actuator **IK** problem.

Regarding the oscillations observed with the **GRD** method, several attempts were made to manually adjust the damping profile, ultimately reducing it to $\lambda_c = 0.4$, at which instance the **GRD** method successfully achieved convergence. At this damping value, both the **DLS** and **PDL** methods also achieved convergence. However, the **JPI** and **SVD** methods continued to experience convergence issues at some points in the middle of the path, regardless of whether **NCX** or **CVX** constraints were used. This was attributed to aggressive damping, which led to oscillations rather than facilitating rapid convergence to an optimal solution.

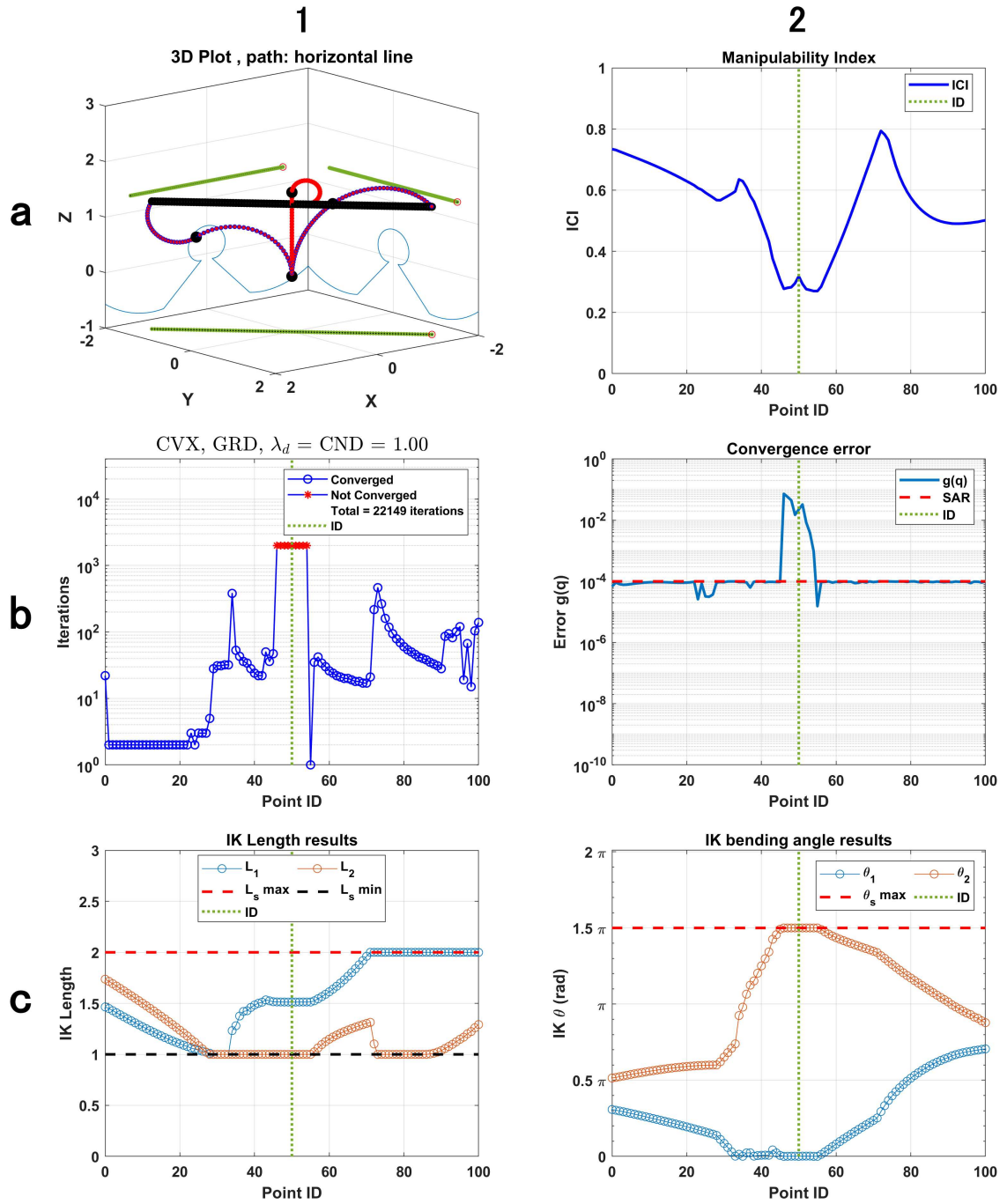


Figure 3.17: IK results for a multi-segment actuator following a horizontal line (first path) using linear approximation with the enforcement of the actuator physical limits. (a1) The 3D path and the actuator workspace are projected onto Cartesian planes. The actuator arcs for the first, middle, and last points in the path are highlighted in purple or red, indicating the convergence status at these points. (a2) The manipulability index. (b1) Convergence iterations for each point along the path using CVX constraints, GRD method, and CND profile. (b2) Components of the convergence error vector at each point, with non-converging points exhibiting errors exceeding SAR. Results for the length L_s and bending angle θ_s for each segment are shown in (c1) and (c2), respectively, indicating that points with no convergence reached the maximum length limit.

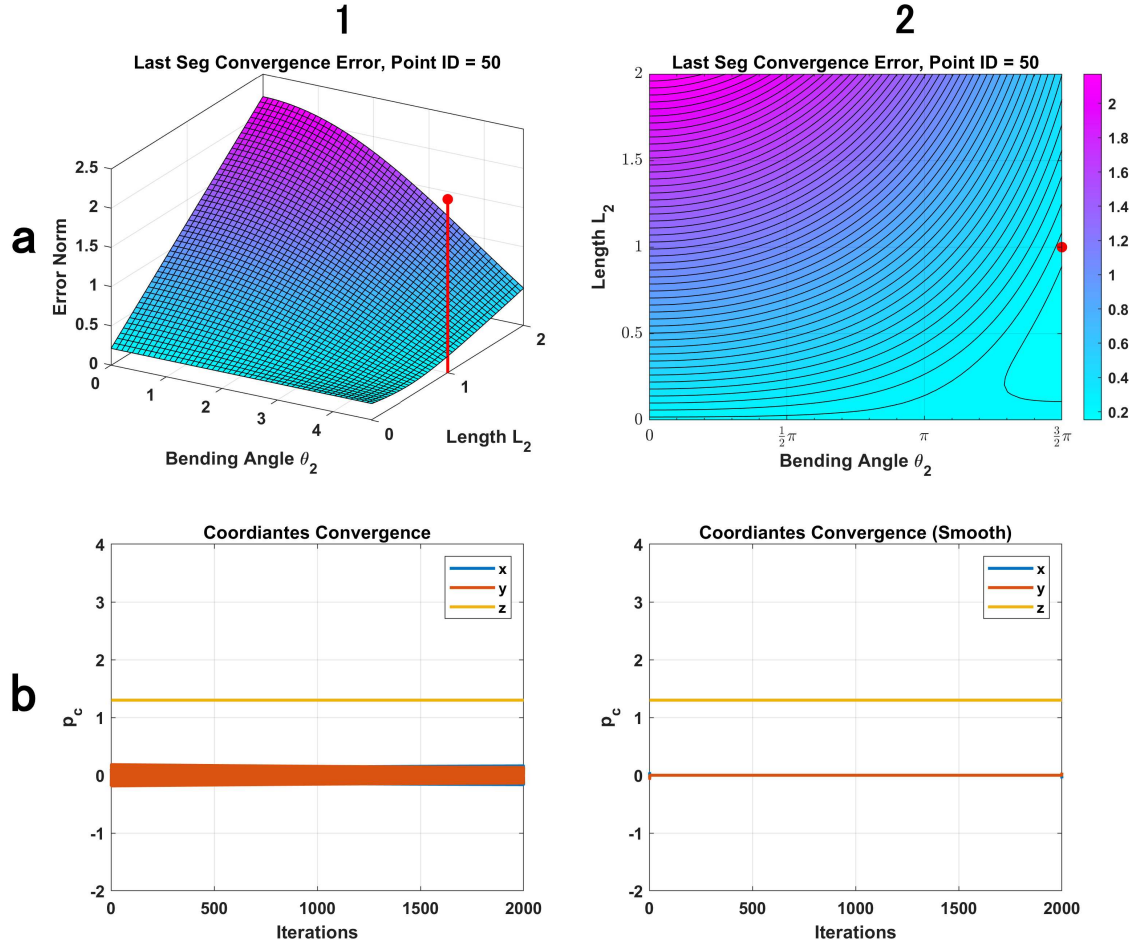


Figure 3.18: Convergence analysis of the IK results for a multi-segment actuator following the first path (horizontal line) using linear approximation with the enforcement of the actuator physical limits. (a1) 3D plot illustrating the convergence error for the second segment at point ID = 50. The length L_s and bending angle θ_s values obtained from the IK results for the second segment are indicated with a vertical red line. (a2) Contour plot of the convergence error for the same point ID, with the IK results marked by a red dot, denoting the minimum convergence error. The raw and smoothed results for the convergence of the actuator endpoint p_c coordinates at the same point ID are presented in (b1) and (b2), respectively.

3.6.5 Quadratic Approximation

This section focuses exclusively on a single-segment actuator, $\mathfrak{s} = 1$, as the derivation of the optimisation Hessian $H_g(q)$ for a multi-segment actuator remains incomplete. The application of the quadratic approximation iteration scheme, as defined in (3.23), yielded unexpected results. Specifically, none of the paths achieved convergence, with iterations ceasing when no further reduction in the location error, defined by the CVX constraints, was observed.

The Hessian $H_g(q)$ of the composite constraints function, as defined in (3.34), was found to have a negative or zero determinant. This characteristic may direct the convergence towards a saddle point rather than a local or global minimum, even with the employment of a CVX constraints function.

As stated earlier in Section 3.2.2, Newton's method converges to a global minimum if $g(q)$ is a strongly convex function with a Lipschitz continuous Hessian $H_g(q)$. When employing a CVX constraints function, the linear approximation of $g(q)$ can indeed lead to convergence towards a global minimum, as demonstrated by the results listed earlier in Table 3.4. However, the quadratic approximation of $g(q)$ lacks a guarantee of convergence. This is because it relies on a linear approximation of the gradient $\nabla g(q)$, and it is not necessarily a convex function.

The quadratic approximation may converge to either a local or global minimum, or to saddle points, depending on the properties of the optimisation Hessian $H_g(q)$. Specifically, a positive definite Hessian generally leads to convergence at a local or global minimum. In contrast, a positive semi-definite Hessian may result in convergence to a saddle point or a plateau within $g(q)$. Conversely, a negative definite Hessian can hinder convergence due to the concavity of the function in that region.

Further analysis revealed that the second term of $H_g(q)$ significantly impacts the convergence performance. A common approach to address this issue is to introduce a weighting factor to stabilise and improve convergence [336]. Accordingly, the optimisation Hessian $H_g(q)$ is modified as follows:

$$H_g(q) = H_{g1}(q) + hsf * H_{g2}(q) \quad \in \mathbb{R}^{ns \times ns}, \quad (3.100)$$

where hsf is a scalar representing the Hessian stabilisation factor. The optimisation Hessian $H_g(q)$ is partitioned into two separate terms, which are outlined as follows:

$$\begin{aligned} H_{g1}(q) &= J_f(q)^T J_f(q), \\ H_{g2}(q) &= \sum_{\mu=1}^m (f[\mu] - p_d[\mu]) H_{f_\mu}(q). \end{aligned} \quad (3.101)$$

In addition, to address potential issues with non-invertible Hessians, the SVD inverse, denoted as $H_g(q)^\dagger$, is employed in (3.23) in place of standard inversion. Additional details regarding the SVD inverse can be found in Section 3.4.4. Several values for hsf within the range $[0, 1]$ were manually selected to evaluate the performance of the

IK solver. The convergence results across all four paths are presented in Table 3.6, with results highlighted in red indicating points where convergence was not achieved. Additionally, the results for $hsf = 1$ reflect the convergence performance without any stabilisation, as discussed earlier in this section.

Table 3.6: IK convergence results for all four paths using quadratic approximation with Hessian stabilisation factor hsf , without the enforcement of actuator physical limits.

Quadratic	hsf										
Path	1	0.5	0.2	0.1	0.05	0.02	0.01	0.005	0.004	0.001	0
Hor. Line	54,222	305	314	116	309	111	116	110	111	306	112
Helical	18,630	14,508	114	307	313	306	308	307	308	305	118
3D Curve	16,534	10,321	308	113	305	305	308	304	304	304	123
Eight	24,079	328	311	106	107	115	309	306	109	106	161

As previously mentioned in Section 3.2.2, certain methods reported in the literature, such as Gauss-Newton and BFGS [119], approximate the optimisation Hessian $H_g(q)$ by omitting the second term $H_{g2}(q)$. This approach produces a Hessian matrix that remains positive semi-definite. Nonetheless, the determinant of this Hessian may occasionally be zero or nearly zero. This indicates the presence of at least one direction with zero curvature, suggesting a plateau in the error function $g(q)$. Consequently, error reduction becomes minimal, and iterations may persist until reaching the termination limit $\mathbb{K}_{max}$ or halt if error monitoring is employed and the change falls below 0.1% SAR .

To evaluate the performance of the Hessian approximation, the second term, $H_{g2}(q)$, can be omitted by setting hsf to zero, as indicated in Table 3.6. This approximation proved effective for achieving convergence in the first, second, and third paths. However, it was insufficient for the fourth path, which presents the highest complexity among the paths. Excluding the second term, $H_{g2}(q)$, simplifies the computational load, as the calculation of the actuator Hessian, $H_f(q)$, is no longer necessary for less complex paths.

Several hsf values were manually tested to identify an appropriate value, as shown in Table 3.6. Despite this manual search across a broad range of values, convergence was not achieved for all four paths. A thorough examination of the results indicated that the second term, $H_{g2}(q)$, occasionally had one or more zero or negative eigenvalues, potentially leading to zero or negative eigenvalues for the Hessian $H_g(q)$. Consequently, the determinant of $H_g(q)$ could be zero or negative, depending on the influence of the second term.

For instance, Figure 3.19(a1 to c2) illustrates the IK results for the second path (helical) using a stabilisation factor of $hsf = 0.1$. The related convergence analysis for the point with the maximum number of iterations along this path, point ID = 0 (which is the starting point of the path), is depicted in Figure 3.20(a1 to c2). Error monitoring option was activated to avoid unnecessary iterations.

The **IK** solver was unable to achieve convergence at any point along the path, as depicted in Figure 3.19(b1). This resulted in a zero manipulability index across the entire path, as shown in Figure 3.19(a2). The convergence error was substantial, occasionally exceeding the segment unit length, as illustrated in Figure 3.19(b2). Despite employing an exaggerated mathematical length limit of $L_s = 10$ units in the solver, no convergence was achieved. The persistent flatness of the error function $g(q)$ in this region is believed to be the primary reason for the lack of convergence. The solver was unable to identify optimal solutions, even when pushing the length L_s and bending angle θ_s values to their maximum theoretical limits, as shown in Figure 3.19(c1, c2), respectively.

To address the issue of zero or negative eigenvalues, a straightforward approach is to enforce the positive definiteness of the second term $H_{g2}(q)$. This can be achieved by substituting $H_{g2}(q)$ with an identity matrix of the same dimensions during the iterations [315, 337]. Consequently, the second term in (3.101) is modified as follows:

$$H_{g2}(q) = \begin{cases} I_{ns \times ns}, & \text{if any } \lambda_g < \lambda_{tol}, \\ \sum_{\mu=1}^m (f[\mu] - p_d[\mu]) H_{f_\mu}(q), & \text{otherwise.} \end{cases} \quad (3.102)$$

where λ_g denotes the eigenvalues of the second term $H_{g2}(q)$, and $\lambda_{tol} = 10^{-6}$ is a sufficiently small threshold used to differentiate between zero and positive values. The convergence results for the four paths, employing the same hsf values and including the positive definiteness enforcement, are listed in Table 3.7.

The results demonstrate a significant improvement in convergence performance. A substantial number of hsf values achieved successful convergence for the given paths, with nearly one iteration per point. The improved quadratic solver demonstrates greater efficiency compared to linear solvers. This is evident from the results listed in Tables 3.1, 3.2, 3.3, and 3.4, which show performance across the first through fourth paths.

Table 3.7: **IK** convergence results for all four paths using quadratic approximation, stabilisation factor hsf , and Hessian modification to enforce positive definiteness. Actuator physical limits are not enforced.

Quadratic	hsf + Enforcement										
Path	1	0.5	0.2	0.1	0.05	0.02	0.01	0.005	0.004	0.001	0
Hor. Line	352	227	141	106	106	110	108	115	107	308	112
Helical	1,044	580	293	188	139	114	110	116	118	113	118
3D Curve	1,278	685	333	205	150	124	128	161	116	128	123
Eight	1,067	585	290	183	140	114	111	111	109	107	161

For comparison purposes, the example illustrated in Figures 3.19(a1 to c2) and 3.20(a1 to c2) was re-evaluated with the positive definiteness enforcement applied. The updated results are presented in Figures 3.21(a1 to c2) and 3.22(a1 to c2), respectively.

With the positive definiteness enforcement, the IK solver successfully converged at all points, as shown in Figure 3.21(b1). This convergence led to a positive manipulability index for the entire path, depicted in Figure 3.21(a2). Additionally, the length L_s and bending angle θ_s results were within acceptable ranges, as illustrated in Figure 3.21(c1, c2), and the convergence error remained within the SAR limit, as shown in Figure 3.21(b2).

The convergence error analysis for the point with the maximum number of iterations, point ID = 49, demonstrates optimal convergence, as illustrated in Figure 3.22(a1, a2). The coordinates of the end point p_{cs} remained stable from the first iteration, as shown in Figure 3.22(b1, b2) indicating rapid convergence. The enforcement of positive definiteness for the second term $H_{g2}(q)$ proved effective in producing a positive definite Hessian $H_g(q)$ at this point, as seen in Figure 3.22(c1), and across the entire path, as depicted in Figure 3.22(c2).

Furthermore, the same example was re-evaluated with the enforcement of actuator physical limits. This additional step aimed to assess the effectiveness of the positive definiteness enforcement in scenarios where some points in the path might exceed physical limits, which is likely to hinder convergence. The results of this evaluation are presented in Figures 3.23(a1 to c2) and 3.24(a1 to c2).

The 3D plot and the projections of the path shown in Figure 3.23(a1) illustrate the points that fall outside the actuator's physical limits. Convergence was not achieved at these points, as depicted in Figure 3.23(b1). Despite this, the solver successfully maintained stable actuator configurations and adhered to the path throughout, as evidenced by the positive manipulability index in Figure 3.23(a2), even with some errors exceeding the SAR limit, as shown in Figure 3.23(b2).

The results for the length L_s and bending angle θ_s are consistent with physical limits, demonstrating the solver's capability to maintain stability while following the path. The convergence error remained optimal even at points outside the physical limits, such as the point with ID = 47, as depicted in Figure 3.24(a1, a2). Notably, the coordinates of the end point p_{cs} were stable from the initial iteration, as shown in Figure 3.24(b1, b2). The eigenvalues of the Hessian $H_g(q)$ were all positive at this point, as demonstrated in Figure 3.24(c1), indicating a positive definite Hessian for this point and the entire path, as further confirmed in Figure 3.24(c2).

Analysis of the results presented in Table 3.7 indicates that $hsf = 0.004$ appears to be the optimal value for the four paths examined. However, this does not imply that it is universally optimal for other paths. Identifying the ideal value for hsf remains an open question and will necessitate further investigation to explore its relationship with the actuator Hessian $H_f(q)$ and the complexity of the path.

Moreover, a detailed evaluation of the computational demands associated with linear versus quadratic inverse kinematics solvers was not undertaken. The variability in benchmarking results is influenced by numerous factors that are often beyond the direct control of researchers. These factors encompass variations in hardware configurations, the effects of background processes, and changes in system load. Each of these elements can significantly affect the consistency and reliability of performance comparisons.

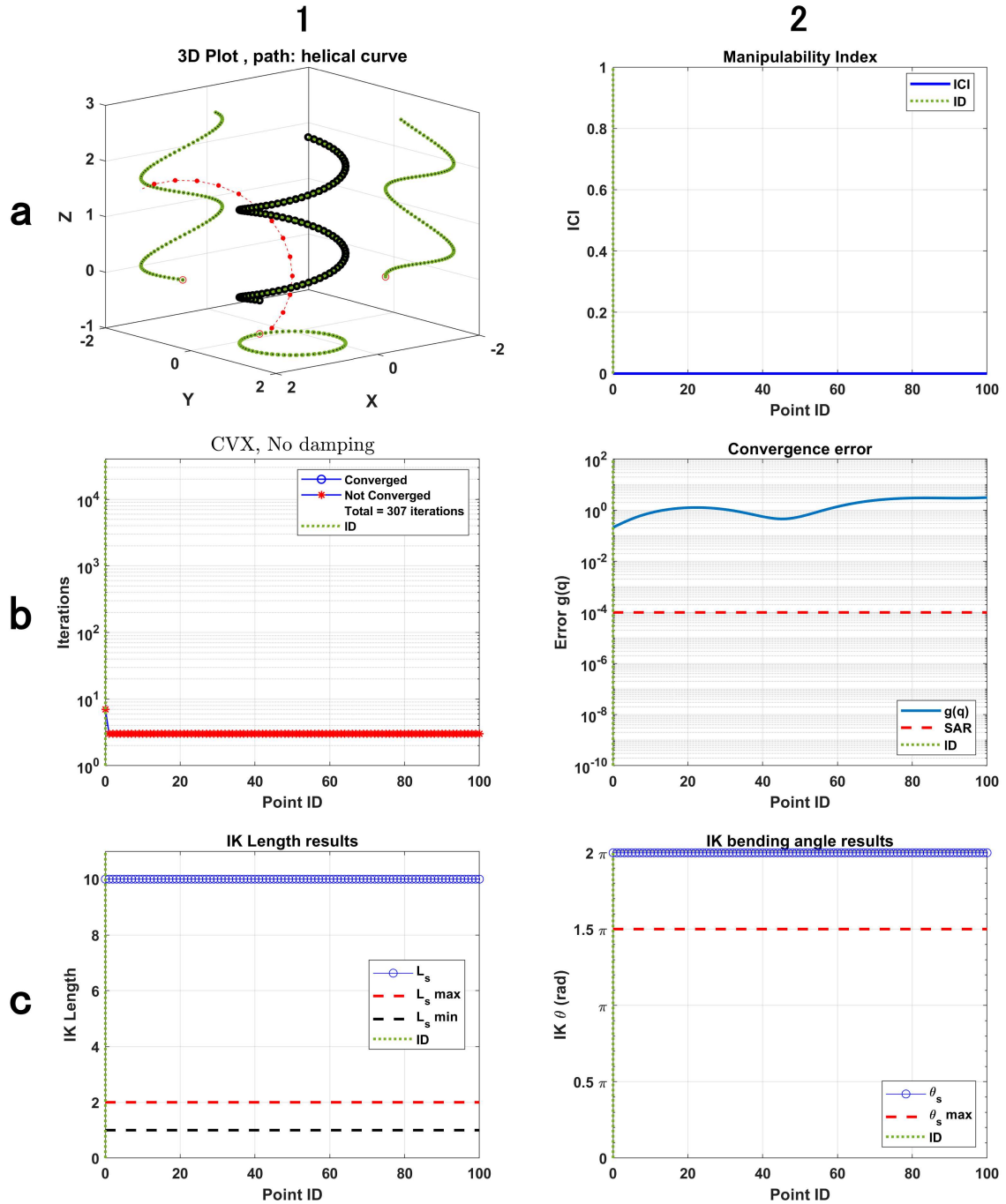


Figure 3.19: IK results for a single-segment actuator following a helical curve (second path) using the quadratic approximation and a Hessian stabilisation factor of $hsf = 0.1$ without enforcing actuator physical limits. (a1) 3D plot of the path and its projections, with the actuator arc for the first point in the path highlighted in red to indicate no convergence. The point with the maximum iterations, ID = 0, is marked with a dotted green line in the other plots. (a2) The manipulability index. (b1) Convergence iterations for each point along the path, showing that no points converged. (b2) Convergence error for each point, with all points exhibiting errors greater than the SAR limit. Results for the length L_s and bending angle θ_s are presented in (c1) and (c2), respectively.

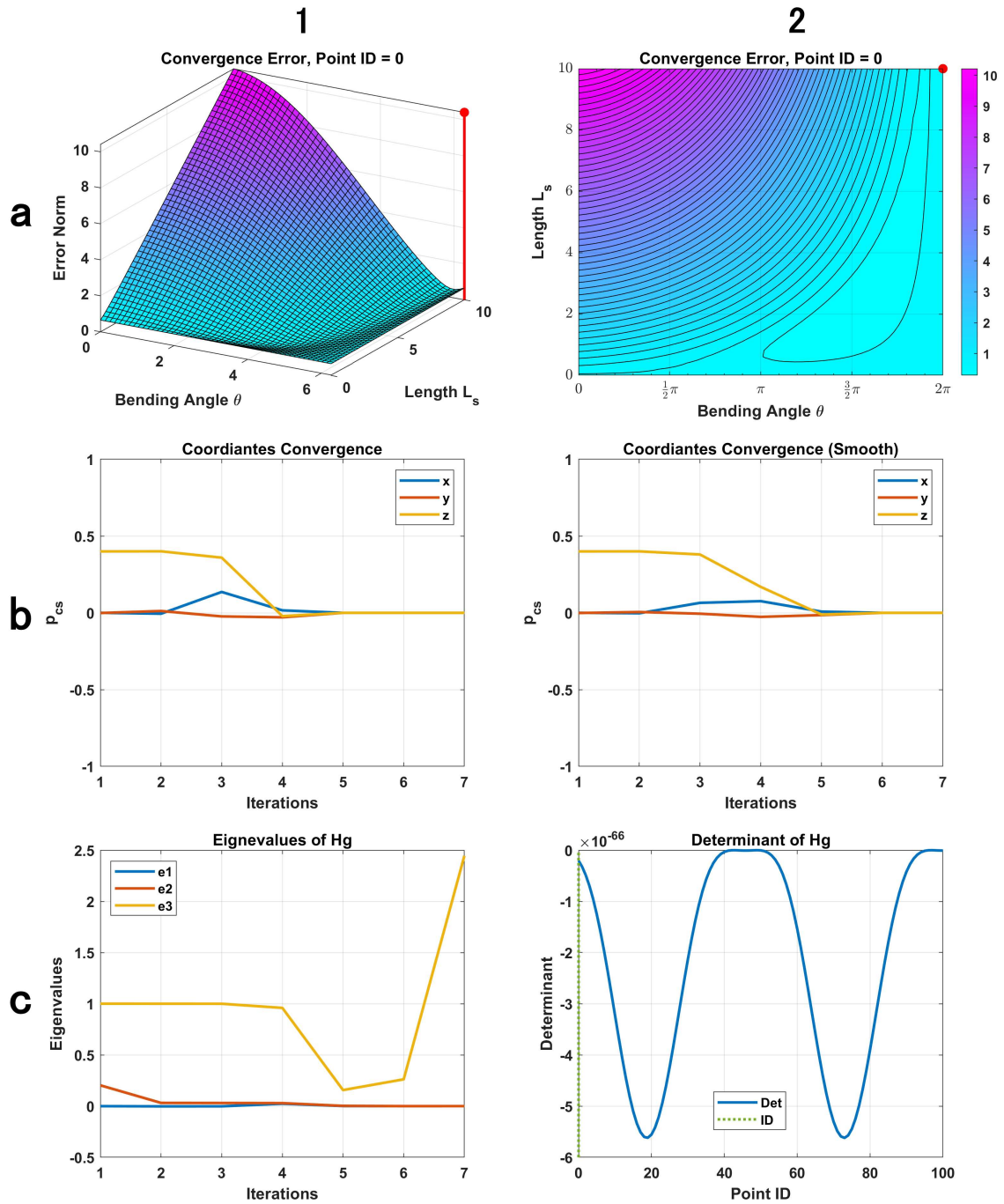


Figure 3.20: Convergence analysis of the IK results for a single-segment actuator following a helical curve (second path) using the quadratic approximation with a Hessian stabilisation factor of $hsf = 0.1$, without enforcing actuator physical limits. (a1) 3D plot showing the convergence error at point ID = 0, with IK results for the length L_s and bending angle θ_s indicated by a vertical red line. (a2) Contour plot of the convergence error for the same point ID, where the IK results are marked with a red dot, indicating that the convergence error is not at the minimum. The raw and smoothed results for the convergence of the end point p_{cs} coordinates at the same point ID are shown in (b1) and (b2), respectively. (c1) Eigenvalues of the Hessian $H_g(q)$ at the same point ID, showing that the Hessian is not positive definite for all iterations. (c2) Determinant of the Hessian $H_g(q)$ at the same point ID, indicating it is negative, hence not positive definite, for all points in the path.

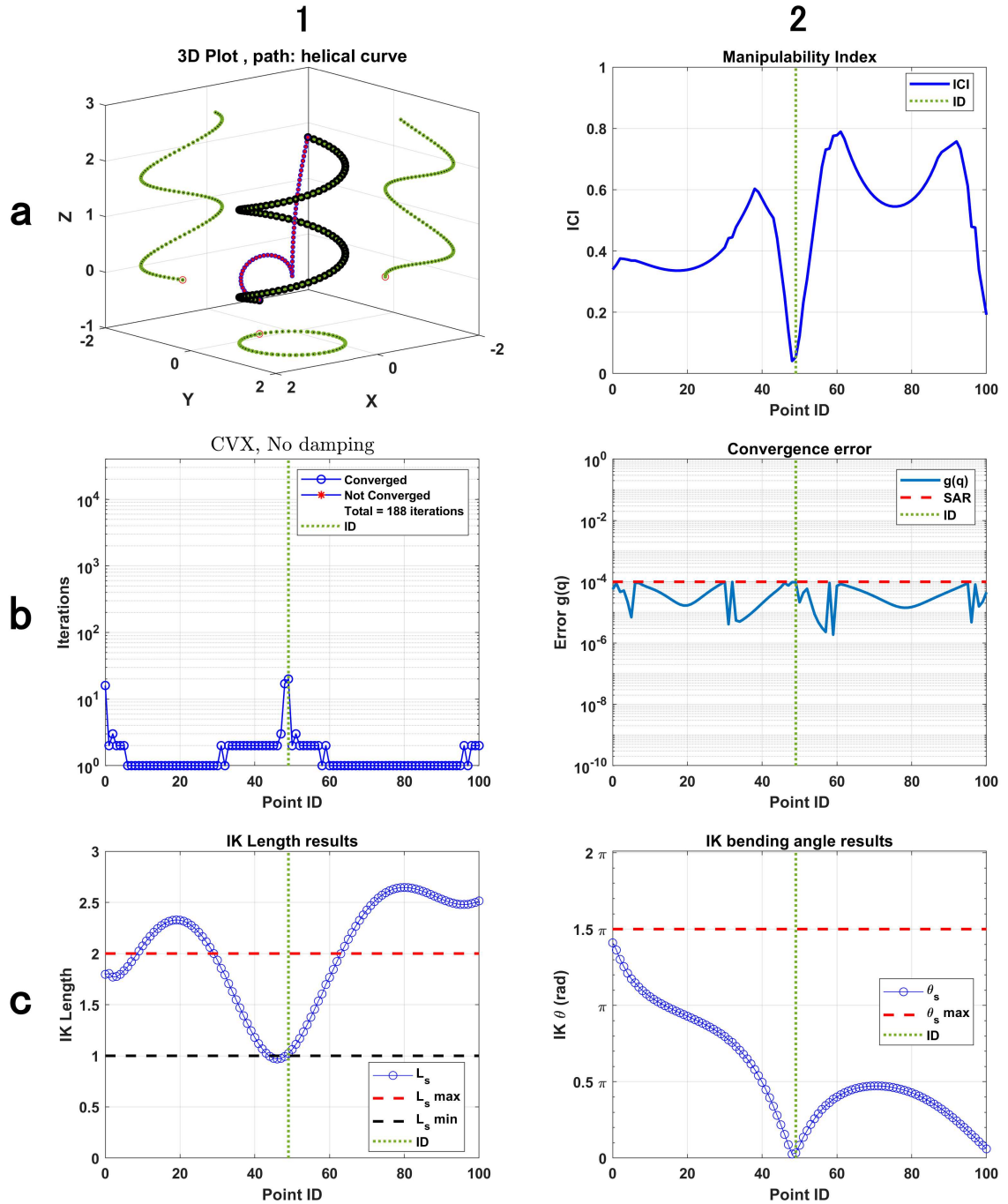


Figure 3.21: IK results for a single-segment actuator following a helical curve (second path) using the quadratic approximation with a Hessian stabilisation factor of $hsf = 0.1$, and positive definiteness enforcement, without enforcing actuator physical limits. (a1) 3D path and its projections, with actuator arcs for the first and last points marked in purple. The point with the maximum iterations, $ID = 49$, is indicated by a dotted green line in the other plots. (a2) The manipulability index. (b1) Convergence iterations for each point in the path, showing that all points have converged. (b2) Convergence error at each point, with errors less than or equal to SAR. Results for the length L_s and bending angle θ_s are shown in (c1) and (c2), respectively.

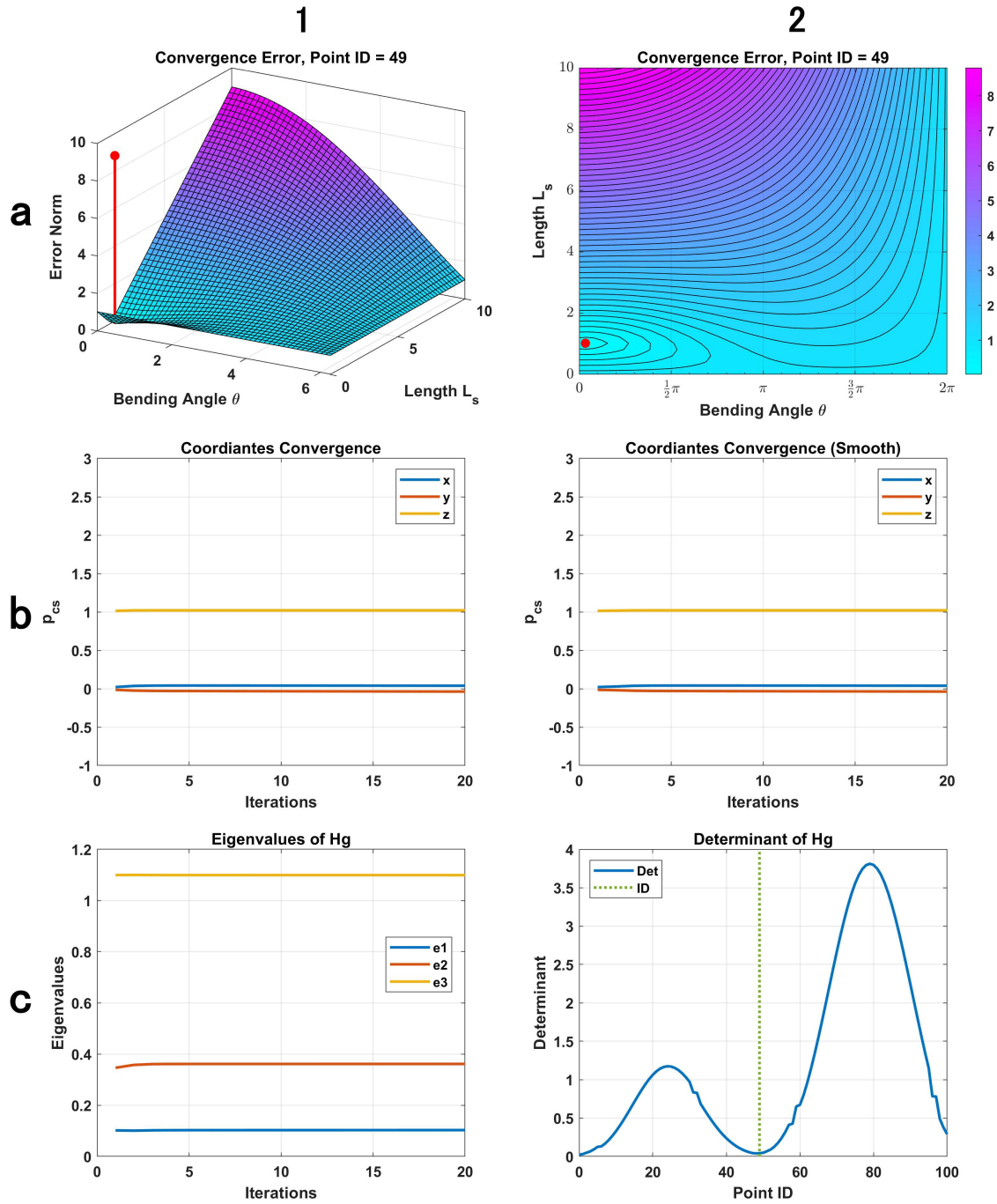


Figure 3.22: Convergence analysis of the IK results for a single-segment actuator following a helical curve (second path) using the quadratic approximation with a Hessian stabilisation factor of $hsf = 0.1$ and positive definiteness enforcement, without enforcing actuator physical limits. (a1) 3D plot of the convergence error at point ID = 49, with the IK results for the length L_s and bending angle θ_s marked by a vertical red line. The convergence error is at its minimum, as confirmed by the contour plot in (a2). The raw and smoothed results for the convergence of the end point p_{cs} coordinates at the same point ID are shown in (b1) and (b2), respectively. (c1) Hessian $H_g(q)$ eigenvalues at the same point ID, indicating positive definiteness in all iterations. (c2) Hessian $H_g(q)$ determinant, which remains positive definite for all points in the path.

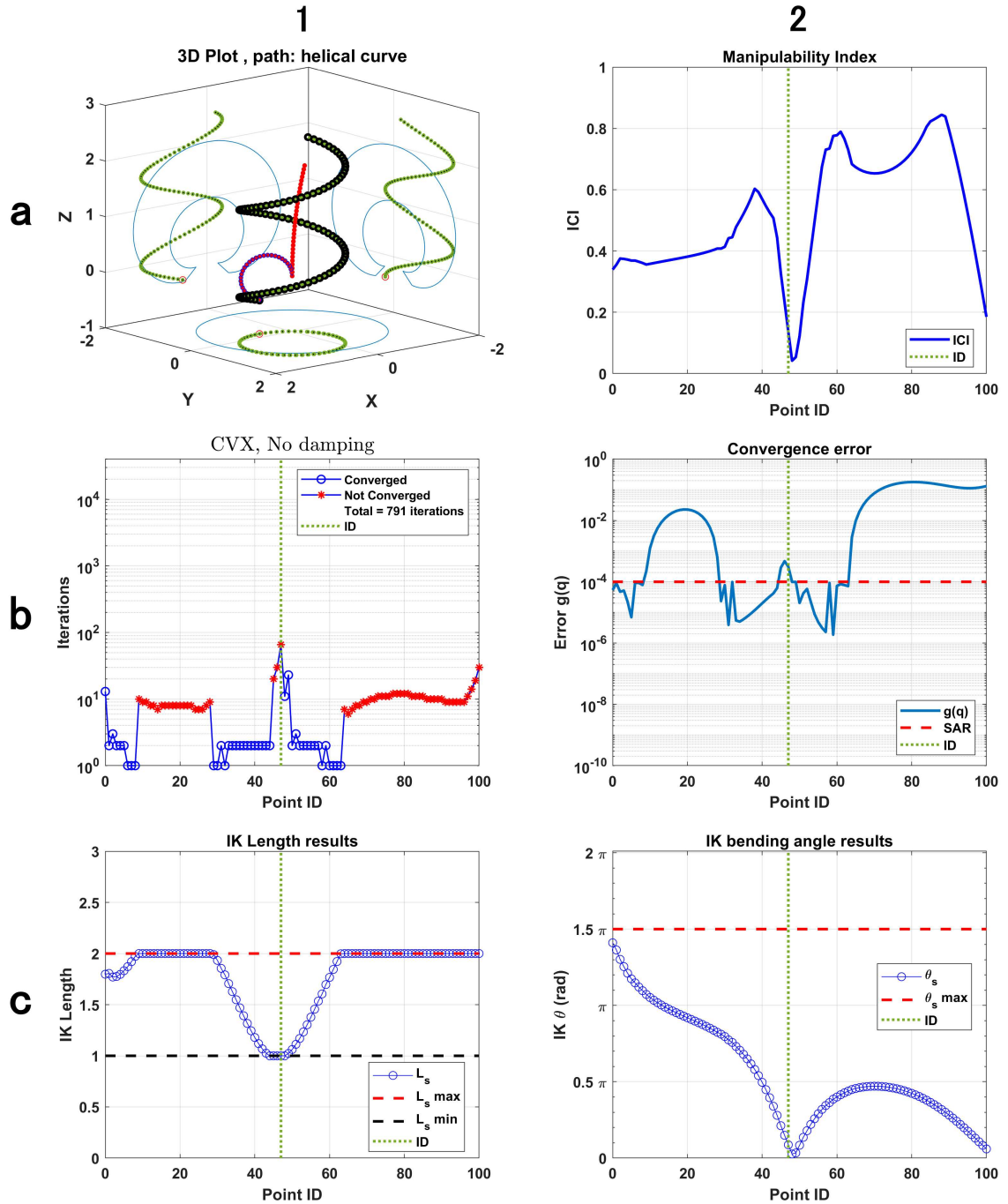


Figure 3.23: IK results for a single-segment actuator following a helical curve (second path) using the quadratic approximation with a Hessian stabilisation factor of $hsf = 0.1$ coupled with enforcement of both positive definiteness and actuator physical limits. (a1) 3D path with projections of the actuator workspace on Cartesian planes. Actuator arcs for the first and last points in the path are marked in purple and red, respectively, indicating their targeted positions relative to the limits. The point with the maximum iterations, $ID = 47$, is highlighted with a dotted green line in the other plots. (a2) The manipulability index. (b1) Convergence iterations for each point in the path. (b2) Convergence error at each point. Results for the length L_s and bending angle θ_s are shown in (c1) and (c2), respectively.

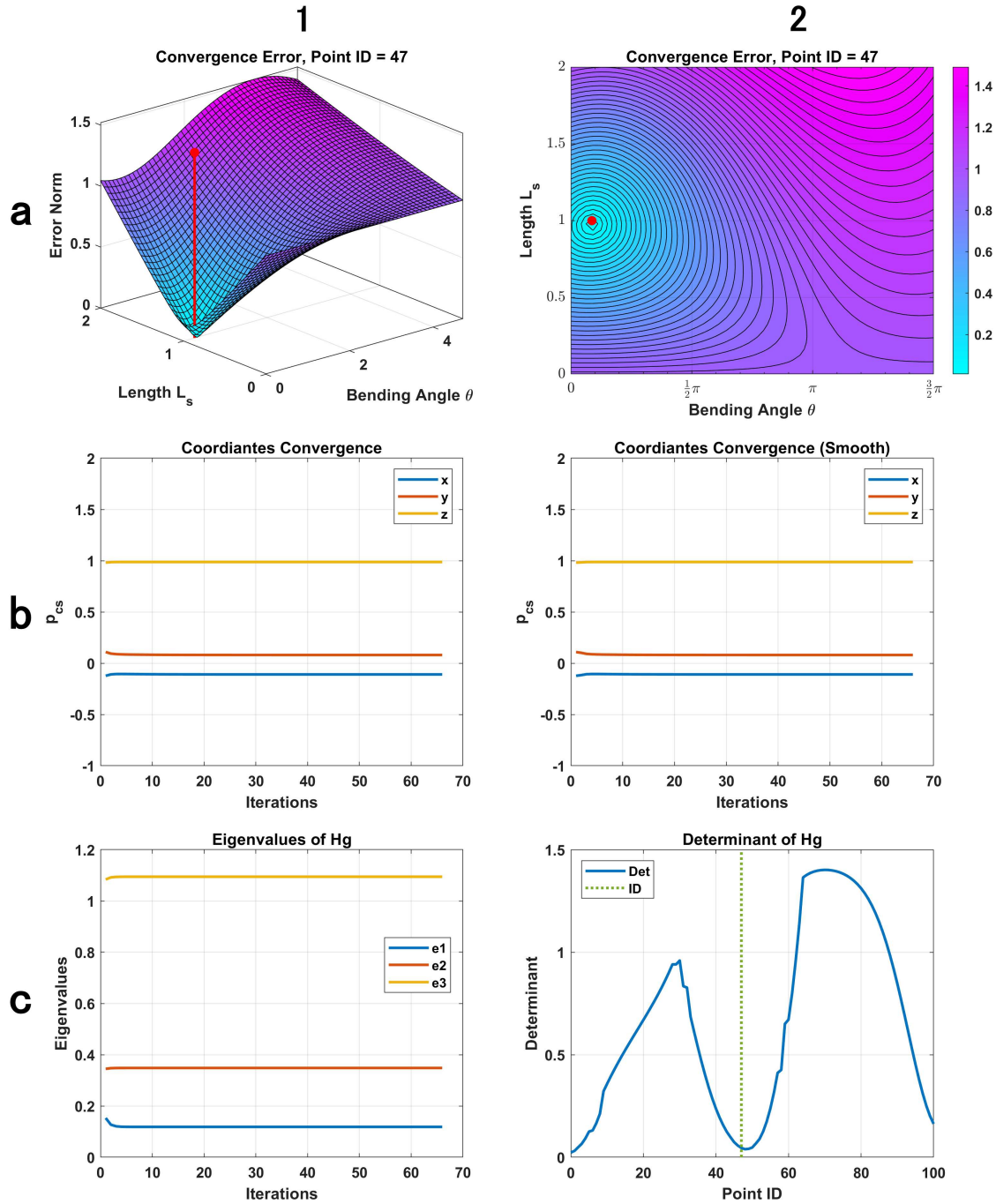


Figure 3.24: Convergence analysis of the IK results for a single-segment actuator following a helical curve (second path) using the quadratic approximation with a Hessian stabilisation factor of $hsf = 0.1$ coupled with enforcement of both positive definiteness and actuator physical limits. (a1) 3D plot of convergence error at point ID = 47, with IK results for the length L_s and bending angle θ_s marked by a vertical red line. The minimum convergence error is confirmed in the contour plot in (a2), despite being outside the physical limits. The raw and smoothed results for the convergence of the end point p_{cs} coordinates at the same point ID are shown in (b1) and (b2), respectively. (c1) Hessian $H_g(q)$ eigenvalues at the same point ID, indicating positive definiteness in all iterations. (c2) Hessian $H_g(q)$ determinant, which remains positive definite for all points in the path, irrespective of the physical limits.

3.7 Conclusion

The **PCC** model streamlines the modelling process of **SPAs** by dividing the actuator into one or more constant curvature segments. Each segment is defined by three parameters: the rotation angle ϕ_s , the bending angle θ_s , and the length L_s . Together, these parameters form the configuration space vector q_s for each segment.

The **PCC** model encounters an inherent singularity at the straight position, $\theta_s = 0$. While previous solutions relied on alternative analytical representations to address singularity, this study proposes a numerical solution by implementing a minimum bending limit, ensuring the bending angle remains above zero.

A limit of $\theta_s = 2 \times 10^{-4}$ was found to be suitable for maintaining the stability of the **IK** solver and achieving the required spatial accuracy ratio *SAR*. The numerical solution not only effectively overcame the singularity but also ensured the use of the most common representation, arc parameterisation, instead of the other analytical representations mentioned earlier in Chapter 2, Section 2.1.3.

Additionally, the **PCC** model faces a limitation concerning the discontinuity in the rotation angle. To address this issue, a two-step correction for Euler angles was implemented. This correction proved effective in minimising the discontinuity issue, thereby ensuring smooth performance for the **IK** solver.

Newton's method is employed to address the **IK** optimisation problem using both linear and quadratic Taylor series approximations. The actuator Jacobian $J_f(q)$ and its Hessian $H_f(q)$ were derived, essential components for implementing the **IK** solver. Specifically, the Jacobian includes both linear and rotational components, while only the linear part of the Hessian was derived, as per the scope defined in this chapter. The derivation of the Hessian $H_f(q)$ for multi-segment actuators was not pursued due to the time constraints specific to this study.

The actuator's classification as non-holonomic, holonomic, or redundant depends on the number of segments and movement constraints. Two types of constraints were investigated in this study. The first type of constraints, **NCX** constraints, represents the error vector between the current and desired pose of the actuator. The **NCX** constraints encompass six total constraints: three for Cartesian coordinates and three for rotations about Cartesian axes.

The second type of constraints, **CVX** constraints, which represents the Euclidean norm of the error vector, encompassing three constraints for Cartesian coordinates. Rotation angles were excluded due to their distinct measurement units. For both **NCX** and **CVX** constraints, optimisation Jacobian $J_g(q)$ and Hessian $H_g(q)$ were derived.

The concept of actuator manipulability is introduced, and an appropriate index is proposed based on previous studies. This manipulability index serves to indicate proximity to singularity while traversing specific paths. On top of that, various Jacobian algorithms from the literature are outlined in this chapter, providing a necessary context for their later analysis in this study. Several damping profiles are also suggested to evaluate their efficacy in mitigating large motions and oscillations near singularities.

To assess the [IK](#) solver performance, four distinct paths were selected: a horizontal line, a helical curve, a 3D curve, and a figure-eight curve. These paths were chosen and ordered based on their varying degrees of complexity. Each path is discretised into 100 points to precisely monitor all changes in the actuator's state throughout traversal. Initiating from a straight position, the actuator progresses sequentially through each point along the path. The total number of iterations required to converge at each point is meticulously recorded, and the outcomes are tabulated separately for each path.

The study is structured into four distinct categories based on the type of [IK](#) solver employed. Initially, the focus lies on the linear [IK](#) solver, where comprehensive analyses are conducted on a single-segment actuator operating without imposed physical limits. This foundational exploration sets the stage for subsequent investigations into the impact of enforcing these limits, both on single-segment actuators and on more complex multi-segment configurations.

Transitioning to the quadratic [IK](#) solver, the study returns to the single-segment actuator scenario, this time under the lens of the Hessian matrix's performance within [CVX](#) constraints. Here, particular attention is given to identifying potential enhancements to ensure robust convergence and optimal performance.

Throughout the linear-based analyses, the study meticulously evaluates multiple Jacobian algorithms across the suggested damping profiles for both [NCX](#) and [CVX](#) constraints. The findings from these analyses provide detailed insights into solver performance across different scenarios. The performance of the Jacobian algorithms was observed to be influenced by the complexity of the path and the damping profile.

For example, the [JIV](#) method without damping was found to be the suitable choice for the first path, given its simplicity, especially since the optimisation Jacobian $J_g(q)$ is a square matrix and invertible. In contrast, a solver based on [CVX](#) constraints and the [SVD](#) method without damping proved to be optimal for the fourth path, which is the most complex.

In the case of a multi-segment actuator, the [JIV](#) method is no longer suitable as the optimisation Jacobian $J_g(q)$ becomes a non-square matrix. Here, a solver utilising [CVX](#) constraints and the [SVD](#) method was again found to be effective.

The damping profile was observed to have minimal impact, primarily due to the rapid convergence facilitated by [CVX](#) constraints. However, further investigation is needed to determine the specific relationship between each damping profile and Jacobian algorithm, particularly in scenarios involving more complex paths.

In the fourth category of this study, a quadratic [IK](#) solver based on the exact Hessian exhibited faster convergence rates compared to both the quadratic solver with an approximated Hessian and the linear solver utilised in the first category. However, it is important to note that the quadratic solver is not inherently a [CVX](#) function, as it approximates the gradient $\nabla g(q)$ rather than the [CVX](#) constraints function itself.

Consequently, the original form of the quadratic solver failed to converge across all paths, necessitating the introduction of a Hessian stabilisation factor hsf to enhance convergence. Multiple values of hsf were manually selected, with many proving ineffective until the positive definiteness of the Hessian was ensured using

a replacement identity matrix. This technique proved highly effective, leading to rapid convergence with generally only one iteration per point across various path and hsf combinations.

It is critical to acknowledge that all these analyses were conducted in a simulated environment. Nevertheless, in an experimental setup, a suitable sensing method—whether proprioceptive or alternative—is necessary to provide feedback on the actuator configuration while following a given path. Several factors impact the accuracy of the feedback and, consequently, the IK solver, including measurement noise, state estimation efficiency, and the validity of the PCC assumption relative to the actuator design.

In conclusion, this study presents an opportunity for additional investigation and enhancement, particularly in exploring the Hessian stabilisation factor hsf relationship with the actuator Hessian $H_g(q)$ and path complexity. Future investigations could extend these analyses to include multi-segment actuators after deriving the complete actuator Hessian. Additionally, enhancing the non-convex quadratic solver could involve implementing convex optimisation techniques such as linearisation and relaxation [97, 127, 128].

Chapter 4

Spiral Sensors Modelling

This chapter presents a robust mathematical model to accurately measure the twist exhibited by SPAs, leveraging fundamental principles from the concepts of *geodesics* and *covering spaces*. Then, four illustrative case studies are presented to elucidate the diverse applications of the proposed mathematical framework for assessing twist across varying deformations.

Furthermore, a simplified FEA model, implemented within Abaqus, is unveiled. This model incorporates an orthotropic layer as a means of reinforcing the SPA structure. The validity of this approach is subsequently assessed through a comparative analysis, comparing simulation outcomes obtained from a single muscle SPA with previously documented findings.

Subsequently, a multi-muscle FEA model is developed to further investigate the mathematical model's performance and reliability. Utilising the multi-muscle FEA model, the mathematical model's accuracy and robustness are verified. After presenting the simulation outcomes, a thorough analysis of the mathematical model is conducted, complemented by discussions involving experimental validation against previously published results and addressing inherent limitations.

4.1 The Torus Parameterisation

The implicit definition of a torus T radially symmetric about z -axis in Euclidean space $\mathbb{R}^3$, Figure 4.1(a), is [338]:

$$T := \left\{ (x, y, z) \in \mathbb{R}^3 \mid \left(\sqrt{x^2 + y^2} - c \right)^2 + z^2 = R^2 \right\}, \quad (4.1)$$

where R is the radius of the tube or the minor radius of the torus, and c is the distance from the centre of the tube to the centre of the torus or the major radius of the torus. Only the ring torus, i.e., $c > R$, is considered. This torus is a two-dimensional *Riemannian manifold* embedded in the Euclidean space $\mathbb{R}^3$.

A PCC soft actuator with a circular cross-section can be represented as a toric segment, Figure 4.1(b). The base is fixed at the origin, and the imaginary torus

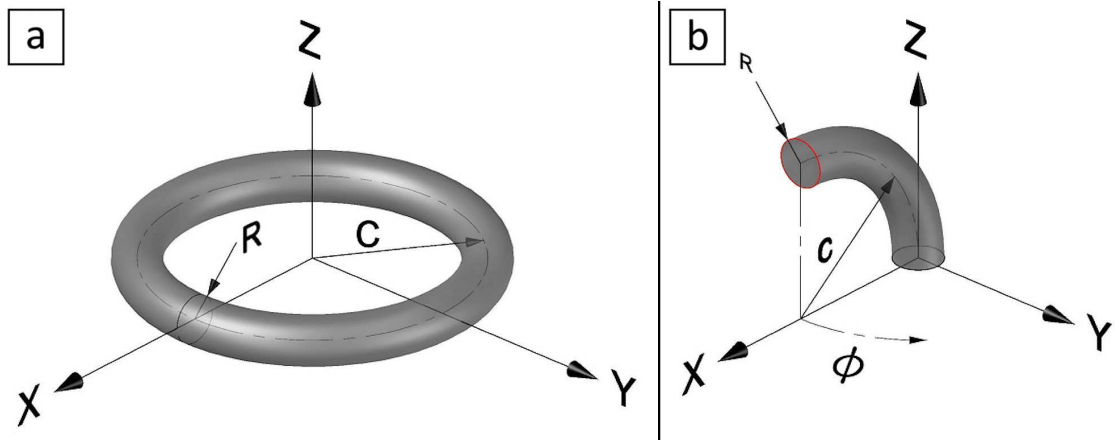


Figure 4.1: Torus geometry in the Euclidean space $\mathbb{R}^3$ where (a) shows a full torus radially symmetric about z -axis, and (b) shows a toric segment moved and rotated to represent a PCC soft actuator.

is symmetric about a centreline parallel to the y -axis when rotation angle $\phi = 0$. Drawing from (4.1), the definition of the toric segment T_{sm} in $\mathbb{R}^3$ is:

$$T_{sm} := \left\{ (x, y, z) \in \mathbb{R}^3 \mid \left(\sqrt{(c-x)^2 + z^2} - c \right)^2 + y^2 = R^2 \right\}. \quad (4.2)$$

A coordinate system $(u, v) \in [0, 2\pi]$ is used such that the Euclidean plane $\mathbb{R}^2$ covers the torus [339] as shown in Figure 4.2(a,b). The torus T is parameterised as a surface of revolution $F_t(u, v)$ by:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = F_t(u, v) = \begin{bmatrix} (c + R \cos v) \cos u \\ (c + R \cos v) \sin u \\ R \sin v \end{bmatrix}. \quad (4.3)$$

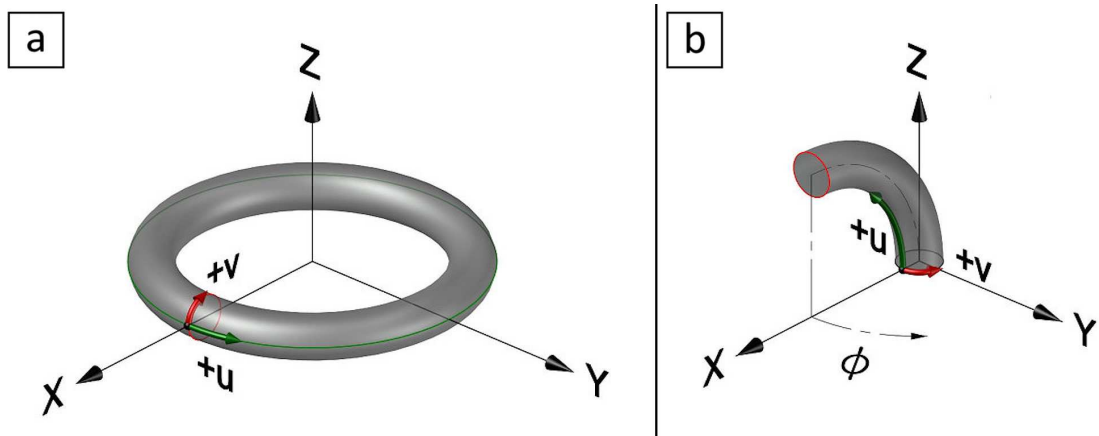


Figure 4.2: Torus notation where (a) shows a full torus radially symmetric about z -axis, and (b) shows a toric segment moved and rotated to represent a PCC soft actuator.

However, there is a difference between T and T_{sm} in the location and direction of u, v coordinates with respect to the Euclidean origin $(0, 0, 0)$. The torus T is transformed by rotating around x and z -axes then shifting by $[c, 0, 0]^T$. Also, transforming v to start from $+x$ and rotating anticlockwise around the centreline, i.e., around $+z$ -axis at the base, as shown in Figure 4.2(b) by substituting $v = \pi - v$ in (4.3). The result is a parameterisation equation of the toric segment T_{sm} as a surface of revolution $F_{sm}(u, v)$ at $\phi = 0$ as follows:

$$F_{sm}(u, v) = R_z(\pi)R_x\left(\frac{\pi}{2}\right)F_t(u, \pi - v) + [c, 0, 0]^T = \begin{bmatrix} c - (c - R \cos v) \cos u \\ R \sin v \\ (c - R \cos v) \sin u \end{bmatrix}. \quad (4.4)$$

To rotate the soft actuator around z -axis by ϕ , another rotation matrix $R_z(\phi)$ is multiplied by (4.4) after modifying the v coordinate to keep the u, v origin at $+x$ -axis. This can be achieved by substituting $v = v - \phi$ in (4.4) which gives:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = F_{sm}(u, v, \phi) = R_z(\phi) \begin{bmatrix} c - (c - R \cos(v - \phi)) \cos u \\ R \sin(v - \phi) \\ (c - R \cos(v - \phi)) \sin u \end{bmatrix}. \quad (4.5)$$

Substituting $c = \frac{1}{\kappa}$ and $u = h \kappa$ where h and κ are the centreline length (or height) and curvature of the toric segment respectively, gives:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = F_{sm}(u, v, \phi) = R_z(\phi) \begin{bmatrix} \frac{1}{\kappa} - \frac{1}{\kappa} \cos(h \kappa) + R \cos(h \kappa) \cos(v - \phi) \\ R \sin(v - \phi) \\ \frac{1}{\kappa} \sin(h \kappa) + R \sin(h \kappa) \cos(v - \phi) \end{bmatrix}. \quad (4.6)$$

When the soft actuator is at a straight position, i.e., $\kappa = 0$, Eq. (4.6) can be evaluated using L'Hôpital's rule at the limit of the curvature $\kappa \rightarrow 0$. Since there is no bending, the parametrisation of the torus converges to the standard parametrisation of a cylinder as shown in Figure 4.3(b). Also, the rotation angle ϕ is neglected since the cylinder, i.e., the soft actuator, is symmetric around z -axis which yields:

$$F_{sm}(u, v, \phi) \xrightarrow[\kappa \rightarrow 0]{limit} \begin{bmatrix} R \cos v \\ R \sin v \\ h \end{bmatrix} = F_{cyl}(h, v), \quad (4.7)$$

where h is the height of the cylinder as well as the centreline length of the toric segment (equivalent to l), and R is the radius of the cylinder as well as the minor radius of the torus.

4.1.1 Soft Sensor Modelling

A soft sensor can be attached to the surface of the soft actuator [140], as illustrated in Figure 4.3. As described previously, when the curvature κ of a toric segment tends to zero, it becomes a vertical cylinder, which can be represented as a plane in a planar space, as shown in Figure 4.3(a). In this representation, the horizontal axis is denoted by v , while u (or h) serves as the vertical axis. It is worth noting that h is a special case of u when κ tends to zero.

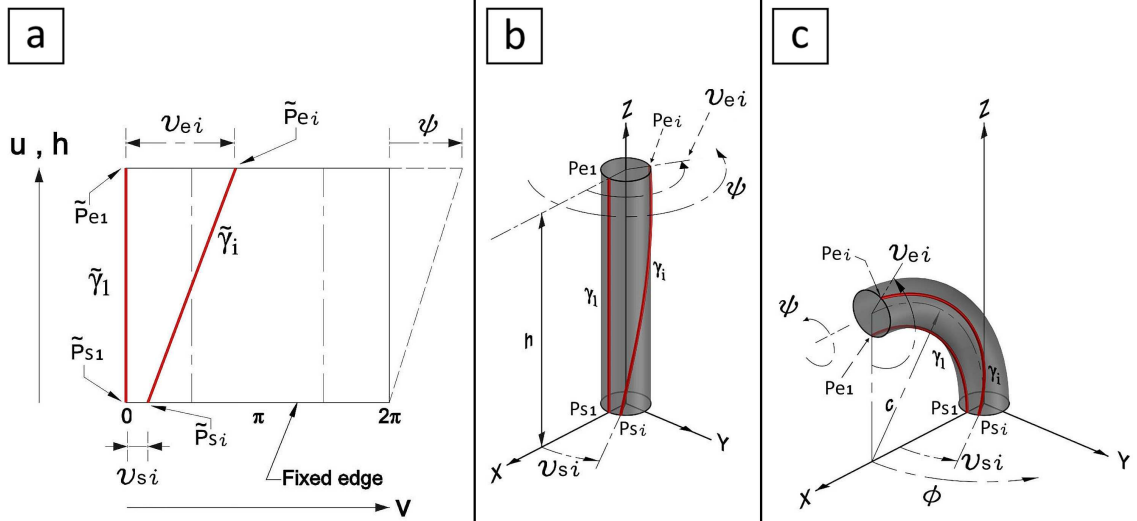


Figure 4.3: (a) A planar cover of a cylinder showing straight lines that bend as (b) geodesic curves on a cylinder, and (c) non-geodesic curves on a toric segment.

To understand the difference between planar and Euclidean spaces' geodesics, consider a point $P_m(u, v) = (\frac{3}{4}\pi, \frac{1}{2}\pi)$ marked on a torus surface, as depicted in Figure 4.4(a). A geodesic curve [340] starting from the origin $(u, v) = (0, 0)$ and passing through P_m is calculated. This curve is then *lifted* to a planar space, Figure 4.4(b), where it manifests as a straight line connecting the origin to a *lifted* point $\tilde{P}_m$.

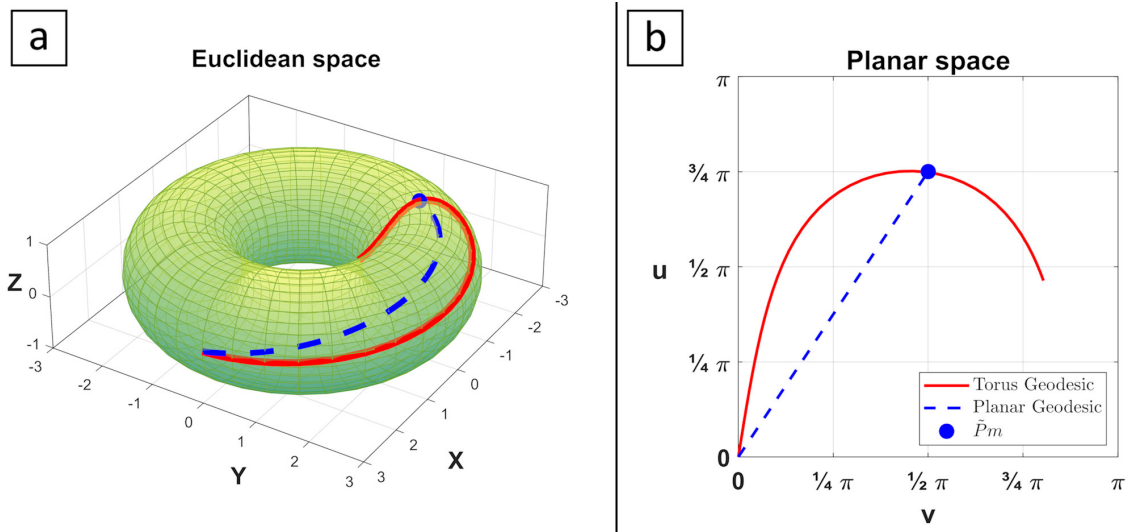


Figure 4.4: (a) Euclidean versus (b) planar geodesics.

This straight line is also *mapped* back to the torus surface in the Euclidean space.

The disparity between the two paths demonstrates that a straight line in the planar space does not necessarily correspond to a geodesic curve on the torus surface. Consequently, relying solely on the geodesic theory is insufficient for deriving a mathematical model for the sensor path in Euclidean space. Given that the torus is considered a Riemannian manifold, it necessitates the integration of concepts from both the geodesic theory and covering spaces to map the planar space onto the Euclidean space.

The sensor path, Figure 4.3(a), can be conceptualised as a straight line $\tilde{\gamma}_i$ between two points $\tilde{P}_{si}$ and $\tilde{P}_{ei}$ in the planar space, which is then *mapped* as a curve γ_i between the corresponding points P_{si} and P_{ei} in the Euclidean space $\mathbb{R}^3$. The start and end horizontal angles are denoted by v_{si} and v_{ei} , respectively. When $v_{si} = v_{ei} = 0$, the sensor path is designated as $\tilde{\gamma}_1$ in the planar space and γ_1 in the Euclidean space.

The following assumptions underpin the modelling:

- The start and end points are situated at the bottom and top edges of the soft actuator.
- The base remains fixed, and a twist angle ψ , Figure 4.3(b), is applied to the top edge.
- The cross-sectional radius of the soft actuator, i.e., the torus inner radius R , remains constant.

The start and end points of the $\tilde{\gamma}_1$ and $\tilde{\gamma}_i$ paths can be defined in terms of $[u, v]^T$ coordinates as follows:

$$\tilde{P}_{s1} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \tilde{P}_{e1} = \begin{bmatrix} h \\ \psi \end{bmatrix}, \tilde{P}_{si} = \begin{bmatrix} 0 \\ v_{si} \end{bmatrix}, \tilde{P}_{ei} = \begin{bmatrix} h \\ v_{ei} + \psi \end{bmatrix}. \quad (4.8)$$

From Figure 4.3(a), the straight line $\tilde{\gamma}_i$ can be parameterised as:

$$\tilde{\gamma}_i(s) = \begin{bmatrix} u(s) \\ v(s) \end{bmatrix} = \begin{bmatrix} sh \\ v_{si} + s(v_{ei} + \psi - v_{si}) \end{bmatrix}, \quad s \in [0, 1], \quad (4.9)$$

where s represents the unit time, with $s = 1$ corresponding to the full length of $\tilde{\gamma}_i(s)$. The parameter s can be interpreted as the length ratio, given that $\tilde{\gamma}_i$ is a geodesic line characterized by unit speed ($|\dot{\tilde{\gamma}}_i(s)| = 1$) and zero curvature. Similarly, starting from Equation (4.5), the curved path $\gamma_i(s) = [x(s), y(s), z(s)]^T$ in Figure 4.3(b,c) is parameterised as:

$$\gamma_i(s) = \begin{bmatrix} x(s) \\ y(s) \\ z(s) \end{bmatrix} = R_z(\phi) \begin{bmatrix} c - (c - R \cos(v(s) - \phi)) \cos u(s) \\ R \sin(v(s) - \phi) \\ (c - R \cos(v(s) - \phi)) \sin u(s) \end{bmatrix}. \quad (4.10)$$

4.1.2 Length modelling

To determine the length of the soft sensor, denoted as l_i , the length of a unit speed curve, γ_i , is calculated which is defined as [149]:

$$l_i := l(\gamma_i) := \int_0^1 \|\dot{\gamma}_i(s)\| \, ds, \quad (4.11)$$

where $|\dot{\gamma}_i(s)|$ represents the norm of the derivative of γ_i with respect to the parameter s , given by:

$$\|\dot{\gamma}_i(s)\| = \sqrt{\dot{x}^2 + \dot{y}^2 + \dot{z}^2}, \quad (4.12)$$

with $\dot{x}$, $\dot{y}$, and $\dot{z}$ denoting the derivatives with respect to s . For brevity, the parameter (s) will be omitted from x , y , z , u , and v operators. Eq. (4.10) can be expanded and simplified as:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} c(1 - c_u)c_\phi + Rc_{v\phi}c_uc_\phi - Rs_{v\phi}s_\phi \\ c(1 - c_u)s_\phi + Rc_{v\phi}c_us_\phi - Rs_{v\phi}c_\phi \\ c s_u - Rc_{v\phi}s_u \end{bmatrix}, \quad (4.13)$$

where $c_{v\phi} = \cos(v - \phi)$, $s_{v\phi} = \sin(v - \phi)$, $c_u = \cos u$, $s_u = \sin u$, $c_\phi = \cos \phi$, and $s_\phi = \sin \phi$. Using the chain rule to find the derivative of (4.13) with respect to the parameter s . The derivatives are denoted by $(\cdot)$. Simplifying gives:

$$\begin{aligned} \dot{x} &= \left(\dot{u}(c - Rc_{v\phi})s_u - \dot{v}Rs_{v\phi}c_u \right) c_\phi - \dot{v}Rc_{v\phi}s_\phi, \\ \dot{y} &= \left(\dot{u}(c - Rc_{v\phi})s_u - \dot{v}Rs_{v\phi}c_u \right) s_\phi + \dot{v}Rc_{v\phi}c_\phi, \\ \dot{z} &= \dot{u}(c - Rc_{v\phi})c_u + \dot{v}Rs_{v\phi}s_u. \end{aligned} \quad (4.14)$$

Substituting (4.14) into (4.12) and simplifying gives:

$$\|\dot{\gamma}_i(s)\| = \sqrt{\dot{v}^2 R^2 + \dot{u}^2 \left(c - R \cos(v - \phi) \right)^2}. \quad (4.15)$$

To make a connection between the planar and Euclidean spaces, the concepts of *covering spaces* [341] is used to *map* $\tilde{\gamma}_i$, defined in (4.9), to γ_i , defined in (4.10), by inserting $\tilde{\gamma}_i$ into F_{sm} after converting $u(s)$ from a unit length in planar space to a unit angle in Euclidean space, i.e., $u(s) \cdot \kappa$, as follows:

$$\gamma_i \hat{=} F_{sm} \circ \tilde{\gamma}_i = F_{sm}(u(s), v(s), \phi), \quad (4.16)$$

which leads to the parameters $u(s)$ and $v(s)$, along with their derivatives as follow:

$$\begin{aligned}
u(s) &= sh\kappa \\
v(s) &= v_{si} + s(\Delta v_i + \psi) \\
\dot{u} &= h\kappa \\
\dot{v} &= \Delta v_i + \psi,
\end{aligned} \tag{4.17}$$

where $\Delta v_i = v_{ei} - v_{si}$.

Finally, substituting these expressions into (4.11) and using the actuator curvature, $c = \frac{1}{\kappa}$, yields the full expression of the soft sensor length in the Euclidean space as:

$$l_i = \int_0^1 \sqrt{R^2(\Delta v_i + \psi)^2 + h^2 \left(1 - R\kappa \cos(v_{si} + s(\Delta v_i + \psi) - \phi)\right)^2} ds, \tag{4.18}$$

which is an integral equation that requires a numerical solution. This equation is applicable only to a ring torus, where $c > R$, as previously noted.

Alternatively, geodesic theory can be applied to model the kinetic energy E_k of a massless particle moving along the sensor path. The resulting model is a fully defined equation, making it simpler to calculate. However, this approach is beyond the scope of this thesis; interested readers can refer to Appendix D for further details.

4.2 Case Studies

To gain deeper insights into the length mathematical model (4.18), four case studies are examined, each involving a soft actuator with a single soft sensor attached to its surface. The first two cases involve the actuator in its initial configuration, while the third and fourth cases explore other configurations where the actuator bends without rotation and with rotation, respectively. For each case, the sensor length is calculated, and the results are further discussed to provide a thorough analysis.

4.2.1 Case 1: A Straight Sensor at the Initial Configuration

The soft actuator is in its initial configuration, characterised by $\kappa = \phi = \psi = 0$, and a sensor is vertically attached to its surface, implying $\Delta v_i = 0$, as illustrated in Figure 4.5(a). Substituting these parameters into (4.18) yields:

$$l_i = \int_0^1 \sqrt{h^2(1 - R\kappa \cos v_{si})^2} ds = h. \tag{4.19}$$

This expression signifies that the sensor, for all possible values of v_{si} , possesses a length equivalent to the centreline length of the soft actuator.

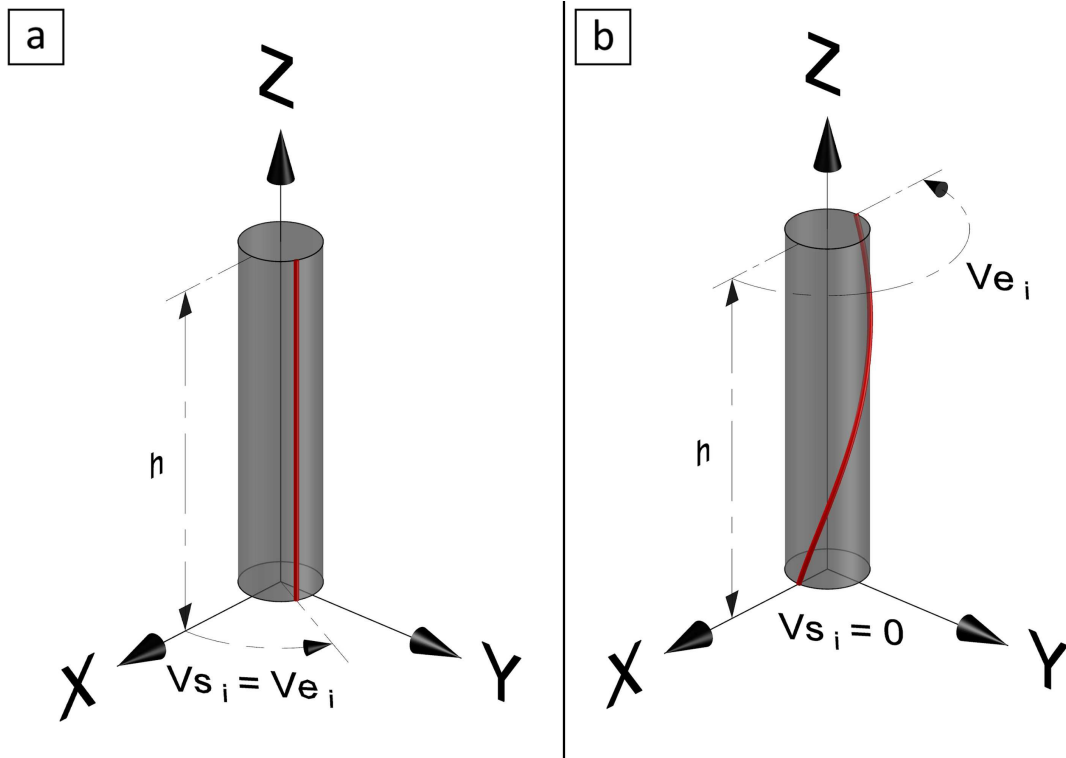


Figure 4.5: Case studies 1 and 2 at the initial position shown in (a) and (b) , respectively.

4.2.2 Case 2: A Spiral Sensor at the Initial Configuration

In this case, the soft actuator remains in its initial configuration, characterised by $\kappa = \phi = \psi = 0$, while the sensor is attached in a spiral fashion along the surface, denoted by $\Delta v_i \neq 0$, as depicted in Figure 4.5(b). Substituting these parameters into Equation (4.18) yields:

$$\begin{aligned}
 l_i &= \int_0^1 \sqrt{\Delta v_i^2 R^2 + h^2 \left(1 - R\kappa \cos(v_{si} + s\Delta v_i)\right)^2} ds, \\
 &= \int_0^1 \sqrt{\Delta v_i^2 R^2 + h^2} ds, \\
 &= \sqrt{\Delta v_i^2 R^2 + h^2}.
 \end{aligned} \tag{4.20}$$

This expression corresponds to the arc length equation of a cylindrical helix [113], capturing the length of the spiral sensor along the soft actuator's surface.

4.2.3 Case 3: A Straight Sensor Under Bending

In this case, the soft actuator experiences bending within the XZ plane without twist, characterised by $\kappa > 0$, $\phi = \psi = 0$, while the soft sensor remains vertically attached to the surface, denoted by $\Delta v_i = 0$, as illustrated in Figure 4.6(a,b). Substituting these conditions into (4.18) yields:

$$l_i = \int_0^1 \sqrt{h^2(1 - R\kappa \cos v_{si})^2} ds = h(1 - R\kappa \cos v_{si}). \quad (4.21)$$

The term $(1 - R\kappa \cos v_{si})$ denotes the curvature ratio, which varies depending on the position of the sensor v_{si} along the surface, as depicted in Figure 4.6(b). Consequently, when the sensor is situated in a plane perpendicular to the bending plane, i.e., $v_{si} = v_{ei} = \frac{\pi}{2}$ or $\frac{3}{2}\pi$ in Case 3, its length equals the centreline length h of the soft actuator. Eq. (4.21) holds within the range $c > R$, as previously noted.

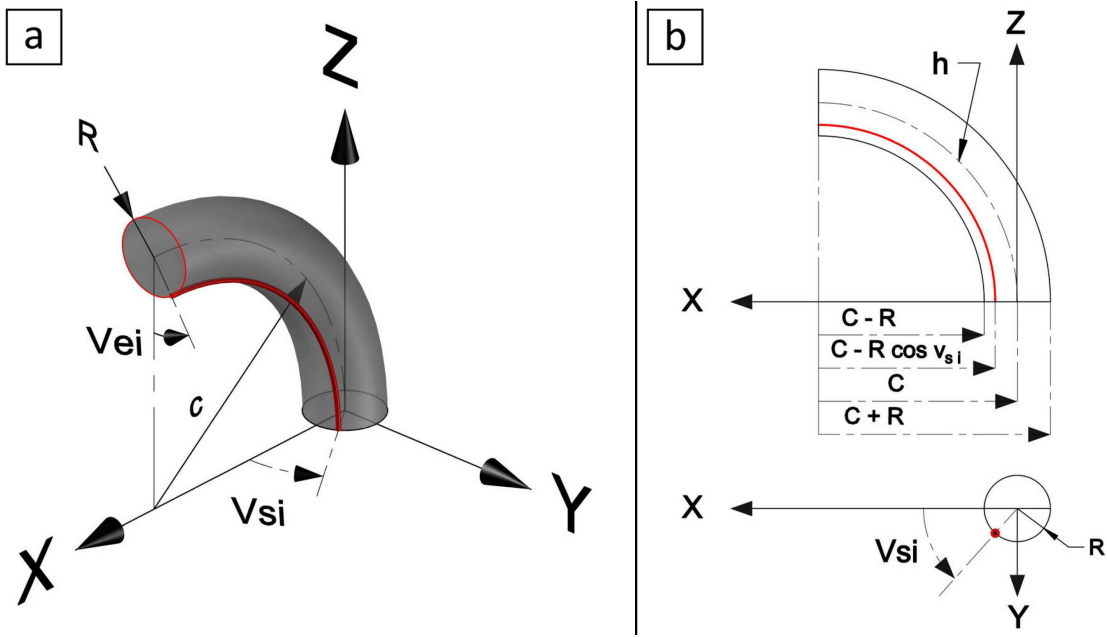


Figure 4.6: Case study 3, bending in XZ plane, where (a) shows a 3D view, and (b) shows the front and top views.

4.2.4 Case 4: A Spiral Sensor Under Bending and Rotation

The soft actuator initiates from its initial configuration with $\kappa = \phi = \psi = 0$ and gradually bends until the top face forms a right angle with the base plane. Simultaneously, the actuator rotates around the z -axis until $\phi = \frac{\pi}{2}$, as depicted in Figure 4.7(a,b). In this fourth case, there is no twisting ($\psi = 0$), and the actuator dimensions are arbitrarily chosen to have a straight position length of $h_o = 250 \text{ mm}$ and an actuator radius of $R = 15 \text{ mm}$.

Two design options for the soft actuator exhibit different behaviours under this bending condition: firstly, the Constant centreline Length Design (CLD) [36, 201, 342], where $h = h_o$ and the inner surface of the toric segment undergoes compression, i.e., $h_{in} \leq h_o$. Secondly, the Non-Compressible inner surface Design (NCD) [129, 189, 197], where there is no compression in the inner surface, $h_{in} = h_o$, and the centreline length extends to accommodate the difference, i.e., $h \geq h_o$.

The extension of the centreline length for NCD depends on the bending angle θ , as illustrated in Figure 4.7(b), and is calculated as follows:

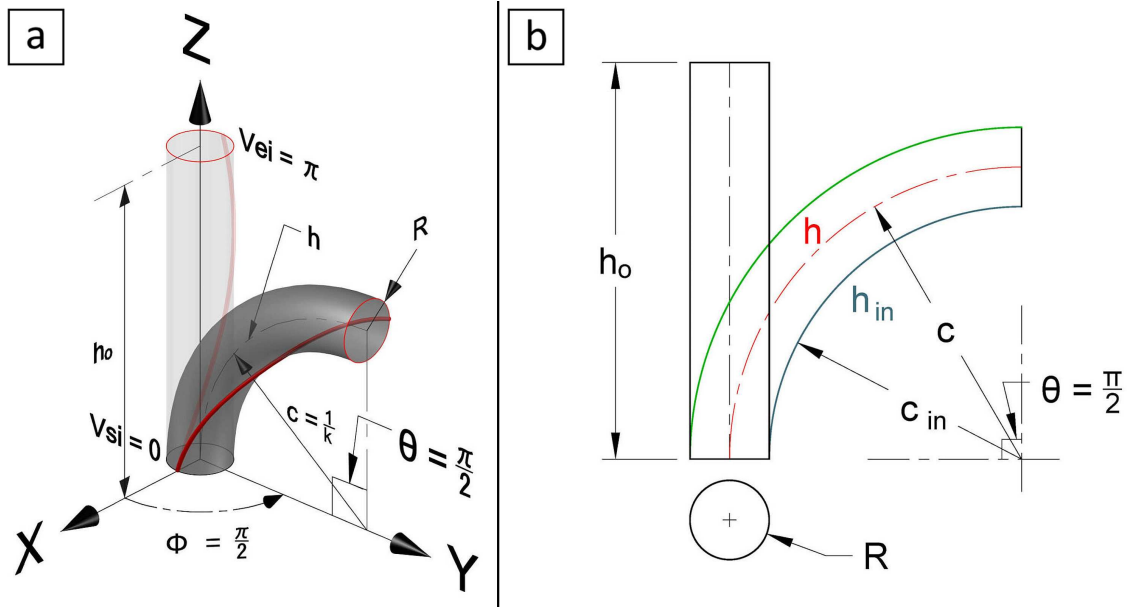


Figure 4.7: Case study 4, bending and rotation, where (a) shows a 3D view, and (b) shows the initial h_o , the centreline h , and the inner surface h_{in} lengths.

$$\theta = \frac{h_{in}}{c_{in}} = \frac{h}{c} \Rightarrow h = \left(1 + \frac{\theta R}{h_o}\right) h_o, \quad (4.22)$$

where c_{in} represents the inner surface radius and is calculated as $c_{in} = c - R$. The sensor is positioned on the actuator surface with $v_{si} = 0$ and $v_{ei} = \pi$, as shown in Figure 4.7(a). Both bending θ and rotation ϕ angles are increased simultaneously. The results of the sensor to actuator length ratio against the change in bending angle θ and rotation angle ϕ are illustrated in Figure 4.8(a). Numerical solutions of (4.18) are obtained using the Runge-Kutta method to calculate the sensor length at each step.

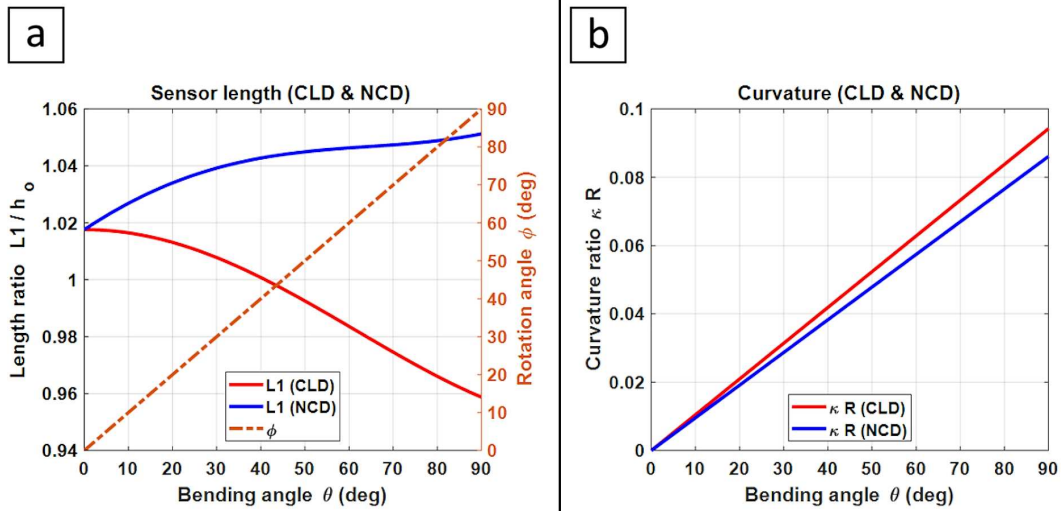


Figure 4.8: Case study 4: (a) Shows both the sensor length and rotation angle ϕ . (b) Shows the actuator curvature. Both figures are plotted against bending angle θ and for CLD and NCD.

The sensor length is approximately 1.8% longer than the actuator's length in the straight position due to the spiral path. For **CLD**, the sensor length decreases by approximately 6% as it passes through the compressed inner radius of the actuator. However, strain sensors cannot measure negative strain, and thus, sensors need to be pre-stretched during actuator fabrication to accommodate compression sensing. For **NCD**, the sensor length increases by around 3%. Pre-stretching is not necessary, but it can enhance sensor accuracy near the straight position. Regarding curvature results, the **CLD** design requires a larger curvature than **NCD** to achieve the same bending angle θ , which is expected due to design differences.

4.3 Finite Element Analysis (FEA)

In Chapter 2, Section 2.2.6, various **FEA** techniques employed to model braided sleeves, whether accounting for contact effects or not, were reviewed. A common challenge noted among these techniques is the complexity arising from the need to incorporate thousands of elements into the **FEA** models.

In this section, an alternative simplified **FEA** model is introduced. Initially, a detailed examination of a prior study concerning the **FEA** modelling of braided sleeves is presented to analyse its outcomes and gather valuable insights.

Additionally, a mathematical formulation for the orthotropic layer concept is provided, followed by an in-depth explanation of the proposed multi-muscle actuator model. Subsequently, mathematical formulations are presented for extracting the lengths of each strain sensor, as well as determining the actuator curvature and its state in 3D space at each time frame.

4.3.1 Braided Sleeves as Reinforcements

T. Hassan *et al.* [27] investigated the relationship between length and diameter changes in a commercially available braided sleeve by building an **FEA** model and comparing the results with experimental tests. The braided sleeve examined had a nominal diameter of 25 mm at rest.

It was observed that the sleeve could attain a maximum diameter of 30 mm when compressed to a length of 58 mm. Conversely, when extended to a length of 110 mm, it reached a minimum diameter of 16.7 mm. This behaviour highlights the sleeve's ability to undergo significant changes in diameter in response to variations in length.

To facilitate the fabrication of an extensible pneumatic muscle, a silicone tube can be inserted into the braided sleeve to restrict radial expansion, as previously mentioned. The diameter of the tube at zero pressure must match the maximum diameter of the braided sleeve, which is 30 mm in this instance.

To analyse the effects of diameter changes, data points were extracted from these results, and a mathematical model was recalibrated accordingly. Using a reference length of 58 mm for the pneumatic muscle at zero pressure, see Figure 4.9(a), it was observed that extending the muscle by 20% (to a length of 69.6 mm) resulted in a decrease in diameter by 1.33% (to 29.6 mm dia.) according to experimental findings,

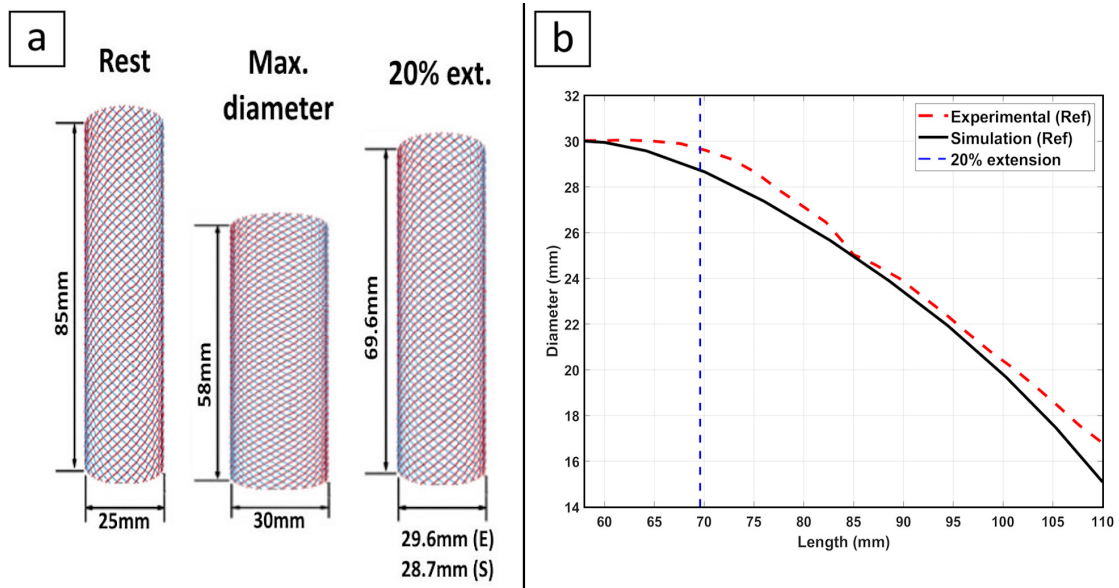


Figure 4.9: Length vs diameter changing of a commercially available braided sleeve.(a) Length and diameter at rest, fully compressed, and 20% extension. (b) Experimental vs simulation results adapted from [27]. Reproduced with permission. Copyright 2019, IEEE

and by 4.3% (to 28.7 mm dia.) according to simulation results as illustrated in Figure 4.9(b).

The change in diameter was found to be minimal within the 20% extension range in the experimental results, allowing it to be disregarded for simplicity. Moreover, the diameter remained consistent across the length at any given moment, as the ends were free to adjust radially. However, in the fabrication of a pneumatic muscle, the diameters at the ends will be fixed to match the size of the chosen attachments or nozzles. The reduction in diameter will occur gradually, reaching its minimum value at the midpoint of the braided sleeve, resulting in a hyperboloid shape.

4.3.2 Orthotropic Layer

To address the intricacies associated with FEA design and computational complexity of braided sleeves, particularly concerning contact interactions, an alternative approach is proposed based on employing a reinforcement layer encircling the surface of each pneumatic muscle. This approach suggests utilising a plane stress orthotropic material model instead of an isotropic one. A similar methodology was adopted in a study to simulate a fibre-reinforcement layer as an anisotropic homogeneous layer [212], where the authors examined their proposal by modelling strain energy and comparing the results with experimental tests.

However, this approach expands upon the aforementioned study by incorporating the modelling of braided sleeves and analysing their impact on pneumatic muscle diameter. Common FEA approaches often entail modelling individual fibres of the reinforcement layer [27, 212, 343]. Instead, the suggestion is to model the entire mesh as a single orthotropic "skin" layer in Abaqus, thereby eliminating the need for individual fibre representations and their associated *contact interactions*, conse-

quently reducing simulation time significantly.

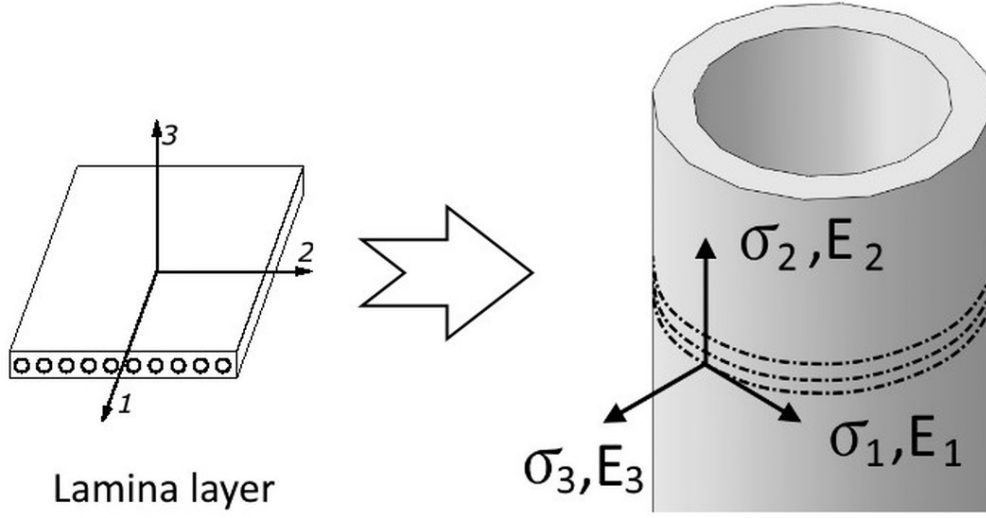


Figure 4.10: Proposed reinforcement orthotropic layer in Abaqus

The proposed reinforcement model enables longitudinal expansion while restricting radial expansion of the pneumatic muscle due to differences in Young's moduli, Figure 4.10. The general stress-strain relation for a material possessing orthotropic symmetry is given by [344]:

$$\begin{Bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \gamma_{12} \\ \gamma_{13} \\ \gamma_{23} \end{Bmatrix} = \begin{bmatrix} 1/E_1 & -\nu_{21}/E_2 & -\nu_{31}/E_3 & 0 & 0 & 0 \\ -\nu_{12}/E_1 & 1/E_2 & -\nu_{32}/E_3 & 0 & 0 & 0 \\ -\nu_{13}/E_1 & -\nu_{323}/E_2 & 1/E_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/G_{12} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/G_{13} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/G_{23} \end{bmatrix} \begin{Bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \tau_{12} \\ \tau_{13} \\ \tau_{23} \end{Bmatrix}, \quad (4.23)$$

where $\{\varepsilon_j\}$ and $\{\gamma_{jk}\}$ represent normal and shear strain components, respectively, $\{\sigma_j\}$ and $\{\tau_{jk}\}$ denote normal and shear stress components, respectively, and $\{E_j\}$, $\{\nu_{jk}\}$, and $\{G_{jk}\}$ are Young's modulus, Poisson's ratio, and shear modulus in the (jk) -plane, respectively. Under the plane stress state, $\sigma_3 = \tau_{13} = \tau_{23} = 0$. Substituting in (4.23) yields:

$$\begin{Bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \gamma_{12} \end{Bmatrix} = \begin{bmatrix} 1/E_1 & -\nu_{21}/E_2 & 0 \\ -\nu_{12}/E_1 & 1/E_2 & 0 \\ 0 & 0 & 1/G_{12} \end{bmatrix} \begin{Bmatrix} \sigma_1 \\ \sigma_2 \\ \tau_{12} \end{Bmatrix}. \quad (4.24)$$

Assuming linear material behaviour, the elastic properties E_j , ν_{jk} , and G_{jk} are constants. Due to the symmetry of the compliance matrix in (4.24), only four independent elastic properties are necessary, namely E_1 , E_2 , ν_{12} , and G_{12} . However, shear moduli G_{13} and G_{23} are also required for modelling transverse shear deformation in Abaqus. The elastic properties of the reinforcement layer are selected based on PET [27], with a density of 1350 kg/m^3 , a Young's modulus $E_1 = 6 \text{ GPa}$, and a Poisson's ratio $\nu_{12} = 0.37$. The remaining elastic properties are discussed below.

To simulate the reinforcement layer as orthotropic in Abaqus, a *Lamina* elastic model is utilised, necessitating the following conditions for material stability in plane stress:

$$E_1, E_2, G_{12}, G_{13}, G_{23} > 0, \quad \text{and} \quad |\nu_{12}| < (E_1/E_2)^{1/2} \quad (4.25)$$

The longitudinal elastic strength of braided sleeves is negligible when its diameter exceeds the minimum limit, equivalent to $E_2 = 0$ in the orthotropic layer. However, the stability condition (4.25) mandates $E_2 > 0$. After testing several assumptions, it was found that any value less than $E_2 = 1 \text{ KPa}$ has a negligible impact on longitudinal extension in the FEA model.

Regarding the shear moduli, G_{13} and G_{23} are irrelevant since the material model is for the plane stress state. Furthermore, the braided sleeve exhibits no shear strength if extended longitudinally before reaching the minimum diameter due to the threads' ability to freely slide on top of one another. Again, the stability condition (4.25) requires $G_{12} > 0$.

After testing several values in Abaqus, it was found that $G_{12} = 0.6 \text{ MPa}$ is the minimum value to reach a stable solution in the FEA model. For all the above, $G_{12} = G_{13} = G_{23} = 1 \text{ MPa}$ was used for simplicity. To simulate the silicone muscle, the Elastosil M4601 properties are utilised, represented by Yoeh [345] hyper-elastic model with coefficients $C_1 = 0.11$ and $C_2 = 0.02$.

4.3.3 Multi-Muscle Actuator

Inspired by prior work [17], The multi-muscle SPA comprises three pneumatic muscles. The actuator's circular cross-section is positioned in the XY plane, with its geometric centre situated at the origin. The actuator extends along the $+z\text{-axis}$, and its three pneumatic muscles are evenly spaced at intervals of 120° , as depicted in Figure 4.11(a). The first muscle is placed in the 1st quadrant, while the second muscle is positioned on the $-x\text{-axis}$. Lastly, the third muscle is placed in the 4th quadrant.

This arrangement enables the generation of bending movement in the XZ plane, i.e., $\phi = 0$, by activating the minimum number of muscles, specifically the second muscle alone. For the representation of each muscle, a silicone tube model is utilised, characterised by a diameter of $d = 8 \text{ mm}$ and a thickness of 2 mm .

An orthotropic layer reinforces the external surface of each muscle. The SPA body, coloured green for differentiation from the muscles, is also simulated as a silicone structure with a diameter of $D = 22 \text{ mm}$ and is divided into three sectors to delineate its centreline, as explained in Section 4.3.4 below.

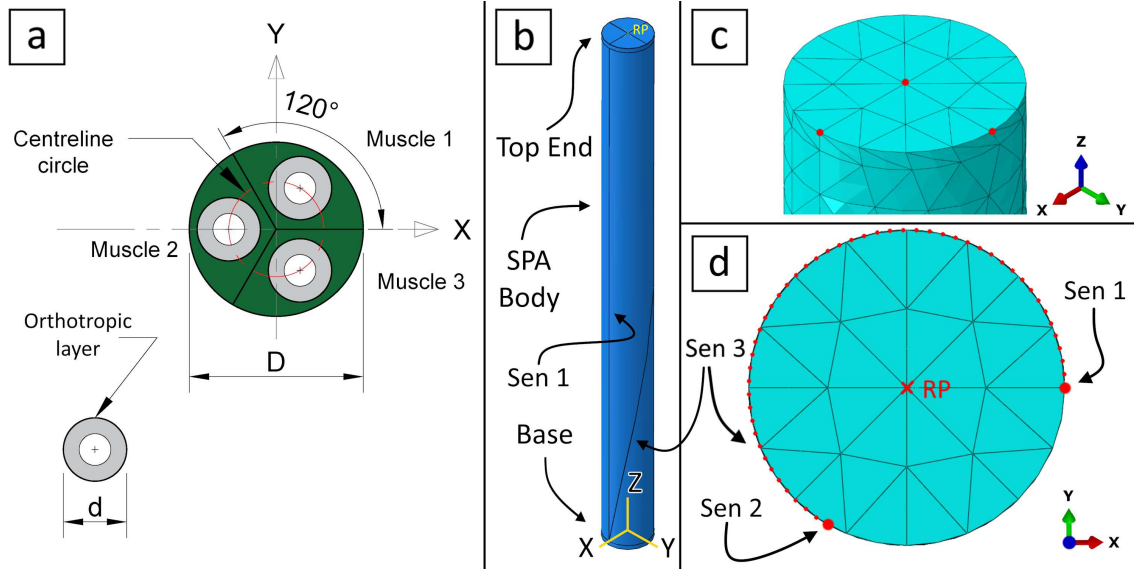


Figure 4.11: FEA model of the multi-muscle SPA where (a) depicts the cross-section details with $D = 22\text{ mm}$ and $d = 8\text{ mm}$, (b) illustrates the SPA surface partitions for the first and third sensors; total actuator length is 254 mm , (c) shows the tracking nodes of the top end, and (d) is a top view showing the three soft sensors and the reference point. The first and second sensors are depicted as dots as they are perpendicular to the XY plane.

The SPA comprises three parts: two Stainless Steel 18/8 discs (base and top end), each 2 mm thick, and a silicone body with a vertical length of 250 mm , as illustrated in Figure 4.11(b). The Stainless Steel 18/8 material has a density of 7930 kg/m^3 , a Young's modulus of $E_1 = 193\text{ GPa}$, and a Poisson's ratio of $\nu_{12} = 0.305$. While the base remains fixed, the top end has freedom of movement and rotation in all directions.

In Abaqus, the STRI65 element type is employed to model the orthotropic layer, which is a thin shell triangular element consisting of 6 nodes and having five DOF per node. Similarly, the C3D10HS element type is employed to simulate the silicone and stainless steel parts. This C3D10HS element is a 10-node general purpose quadratic tetrahedron, offers improved surface stress visualisation.

To induce extension, bending, and rotation movements, pressure loads (p_1 , p_2 , and p_3) can be independently applied to each muscle. For twisting movement, a moment M is applied at the SPA's free end. Since the C3D10HS element only encompasses translation DOF, the moment M cannot be directly applied to the top surface. Instead, a reference point (RP) is defined in space, as illustrated in Figure 4.11(b,d), and connected to the edge of the top end using a *Continuum distributing coupling* constraint. Application of moment M at RP generates the required twist. Anti-clockwise moments are deemed positive, as all rotations in Abaqus adhere to the right-handed rule.

4.3.4 Centreline and Soft Sensors Length

The SPA body is segmented into three sectors, see Figures 4.11(a) and 4.12(c), to delineate the central edge marking the centreline. Additionally, specific locations on the SPA's surface were partitioned to create edges representing three soft sensors, see Figure 4.11(b). The strategic selection of sensor locations ensures sufficient data collection to cover the actuator's entire range of motion, including extension, bending, rotation, and twist.

The first sensor follows a straight path, with $v_{s1} = v_{e1} = 0$. Similarly, the second sensor also follows a straight path, with $v_{s2} = v_{e2} = \frac{4}{3}\pi$. In contrast, the third sensor traces a spiral path, with $v_{s3} = 0$ and $v_{e3} = \frac{4}{3}\pi$, as depicted in Figure 4.11(d). These sensors span the entire length of the SPA body, starting from the base and terminating at the edge of the top end. After the SPA meshing process, four node sets were created along the centreline and the path of each soft sensor. Simulation results provided Cartesian coordinates for each node within these sets at every time frame, utilising a Python code. This Python code exports results to MATLAB for further analysis and comparison with the results obtained from the length model (4.18).

The Python code (`MultiMuscles_data_extract.py`) responsible for data extraction to MATLAB is provided in the supplementary materials of [74].

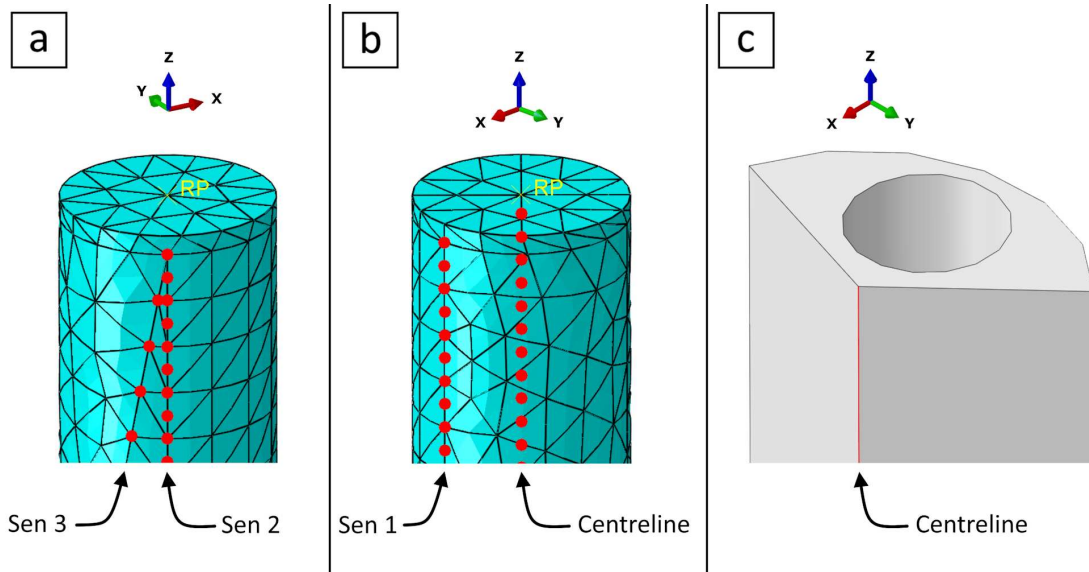


Figure 4.12: Centreline and soft sensors of the multi-muscle SPA where (a) illustrates the nodal path of the second and third soft sensors, (b) shows the centreline and first soft sensor nodal paths, and (c) provides an isometric view of a single sector from the SPA body, highlighting the centreline edge. The second pneumatic muscle, not depicted, occupies the cylindrical hole of this sector.

To calculate the centreline and soft sensors lengths in MATLAB, each dataset was represented as B-spline curves [346]. B-splines are piecewise polynomial functions adept at describing complex trajectories with minimal parameters. This representation facilitates path approximation, yielding smooth curves rather than piecewise polygonal curves with sharp transitions. Notably, B-splines offer the advantage of independence from FEA mesh size.

A B-spline curve of degree p is formed through a linear combination of B-spline basis functions, defined as [347]:

$$B(g) = \mathbf{P}_0 N_{0,p}(g) + \mathbf{P}_1 N_{1,p}(g) + \cdots + \mathbf{P}_n N_{n,p}(g) = \sum_{i=0}^n \mathbf{P}_i N_{i,p}(g). \quad (4.26)$$

where $\mathbf{P}_i$ are the control points, and $N_{i,p}(g)$ are B-spline basis functions of degree p , defined as:

$$N_{i,p}(g) = \frac{g - g_i}{g_{i+p} - g_i} N_{i,p-1}(g) + \frac{g_{i+p+1} - g}{g_{i+p+1} - g_{i+1}} N_{i+1,p-1}(g), \quad (4.27)$$

where g_i are the B-spline *knots* in which $i = 0, 1, 2, \dots, n + p + 1$ and $g_i \leq g_{i+1}$. Similar to (4.11), the length of $B(g)$ is calculated by integrating the norm of its derivative:

$$l_b = \int_0^{g_{n+p+1}} \|B'(g)\|^2 dg, \quad (4.28)$$

where $B'(g)$ is the derivative of a B-spline curve, of degree $p - 1$, defined as:

$$B'(g) = \frac{d}{dg} B(g) = \sum_{i=0}^{n-1} \frac{p}{g_{i+p+1} - g_{i+1}} (\mathbf{P}_{i+1} - \mathbf{P}_i) N_{i+1,p-1}(g). \quad (4.29)$$

Eq. (4.28) can be numerically integrated to calculate the length of the centreline and the three soft sensors at each time frame.

4.3.5 Centreline Curvature

The model (4.18) used to calculate the length of the soft sensors is derived for a constant curvature actuator, as previously stated. Additionally, the curvature value is necessary for comparing the mathematical model with Abaqus results. Hence, the SPA curvature was computed at each time frame.

Let $\gamma_c(t) = (x_1(t), y_1(t), z_1(t))$ be a parametric equation defining the centreline curve in Cartesian coordinates at any given time t . Assuming $\dot{\gamma}_c$ is defined, differentiable, and nowhere equal to the zero vector, the curvature can be derived [348, 349] as:

$$\kappa = \frac{\|\dot{\gamma}_c \times \ddot{\gamma}_c\|}{\|\dot{\gamma}_c\|^3}, \quad (4.30)$$

where $\times$ denotes the vector cross product. Simplifying gives the curvature as:

$$\kappa = \frac{\sqrt{(\ddot{z}_1 \dot{y}_1 - \ddot{y}_1 \dot{z}_1)^2 + (\ddot{x}_1 \dot{z}_1 - \ddot{z}_1 \dot{x}_1)^2 + (\ddot{y}_1 \dot{x}_1 - \ddot{x}_1 \dot{y}_1)^2}}{(\dot{x}_1^2 + \dot{y}_1^2 + \dot{z}_1^2)^{3/2}}. \quad (4.31)$$

Assuming a position tolerance of 0.001 mm for the top end, the minimum curvature to satisfy this tolerance can be calculated using simple trigonometric equations, refer to Chapter 3, Section 3.1.3 for more details. For an actuator with a flexible length of 250 mm , the minimum curvature is $\kappa_{\min} = 3.2 \times 10^{-8} \text{ mm}^{-1}$. The actuator is considered straight for any curvature below this limit. In MATLAB, representing γ_c as a C^2 class B-spline ensures a smooth curve for calculating first and second derivatives numerically. The required Cartesian coordinates were extracted using the procedure described in Section 4.3.4 above.

4.3.6 Bending, Rotation, and Twist Angles

The bending θ , rotation ϕ , and twist ψ angles of the actuator were calculated using the Kabsch algorithm [350]. This algorithm determines an optimal rotation matrix R_k that minimises the RMSE e_k between two paired sets of 3D points, A_k and B_k . In the SPA described in Section 4.3.3 above, these two sets share the same points (nodes), with the first set A_k representing the SPA's initial position and the other set B_k representing any subsequent time frame. The first set A_k comprises three nodes located on the SPA's top edge surface, as depicted in Figure 4.11(c). The nodes are strategically positioned with one at the centre of the surface, another along the positive x -axis, and the last one along the positive y -axis. Tracking the coordinates of these nodes at any given time provides the necessary data sets A_k and B_k for the Kabsch algorithm.

Given the two sets of 3D point $A_k, B_k \in \mathbb{R}^{3 \times 3}$, the transformation is given by:

$$B_k = R_k A_k + t_k + N_k, \quad (4.32)$$

where t_k represents the translation matrix, and N_k denotes the noise matrix. The RMSE is calculated as:

$$e_k = \sqrt{\sum_{i=1}^3 \|R_k A_k^i + t_k - B_k^i\|^2}, \quad (4.33)$$

where the vectors $A_k^i, B_k^i \in \mathbb{R}^3$. Both R_k and t_k can be determined by minimising e_k . To achieve this, the centroids of each data set are computed:

$$cen_A = \frac{1}{3} \sum_{i=1}^3 A_k^i, \quad cen_B = \frac{1}{3} \sum_{i=1}^3 B_k^i, \quad (4.34)$$

and both data sets are re-centred so that both centroids align at the origin, eliminating the translation component and leaving only the rotation. The RMSE then becomes:

$$e_k = \sqrt{\sum_{i=1}^3 \|R_k cen_A - cen_B\|^2}. \quad (4.35)$$

Optimal rotation and translation can be found by computing the [SVD](#) of a cross-covariance matrix H_k :

$$H_k = (A_k - cen_A)(B_k - cen_B)^T = U\Sigma V^T, \quad (4.36)$$

which yields:

$$R_k = V \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & d_k \end{bmatrix} U^T, \quad t_k = cen_B - R_k cen_A, \quad (4.37)$$

where $d_k = \text{sign}(\det(VU^T))$ serves as a correction factor to ensure a right-handed coordinate system. The translation matrix t_k is irrelevant to our current model but is included as a reference for future analysis.

Next, the rotation matrix R_k is converted to Euler rotation angles r_1, r_2, r_3 around the Cartesian axes, with the proper rotation order determined to ensure correct conversion from Euler to [SPA](#) angles. From the [SPA](#)'s initial (straight) configuration, both rotation ϕ and twist ψ angles rotate around the z -axis, while the bending angle θ rotates around the y -axis. This order is denoted as ZYZ . However, for twist alone, this order generates a singularity, and an alternative order, XYZ , is used, which eliminates rotation, $\phi = 0$, since it is irrelevant at the initial position.

When the [SPA](#) curvature $\kappa \geq \kappa_{min}$, which is ZYZ rotation order, Euler angles can be calculated [307] using R_k as follows:

$$\begin{aligned} r_1 &= \arctan\left(\frac{R_{k23}}{R_{k13}}\right), \\ r_2 &= \arctan\left(\frac{\sqrt{1 - R_{k33}^2}}{R_{k33}}\right), \\ r_3 &= \arctan\left(\frac{R_{k32}}{-R_{k31}}\right), \end{aligned} \quad (4.38)$$

and for $\kappa < \kappa_{min}$, which is XYZ rotation order, Euler angles are:

$$\begin{aligned} r_1 &= \arctan\left(\frac{-R_{k23}}{R_{k33}}\right), \\ r_2 &= \arctan\left(\frac{R_{k13}}{\sqrt{1 - R_{k13}^2}}\right), \\ r_3 &= \arctan\left(\frac{-R_{k12}}{R_{k11}}\right). \end{aligned} \quad (4.39)$$

These equations (4.38) and (4.39) can be directly implemented in MATLAB, or the direct cosine matrix to Euler (*dcm2angle*) function can be utilised after inverting the rotation angle R_k to change the rotation from extrinsic to intrinsic. Finally, SPA angles are calculated as:

$$\phi = r_1, \quad \theta = r_2, \quad \psi = r_1 + r_3. \quad (4.40)$$

It is worth noting that the twist angle ψ is the summation of the two Euler angles r_1 and r_3 . This is because rotating the actuator around the z -axis causes the top end face to rotate in the opposite direction [21], given that the base is fixed. Therefore, $\psi = 0$ when there is no external or additional twist.

4.4 Verification, Validation, and Discussion

This section presents the results of simulations conducted to verify the mathematical model (4.18), which describes the length of soft sensors when attached to the surface of a toric-shaped soft actuator, as discussed in Section 4.1.2. A novel approach to simulate the reinforcement layer is introduced in Section 4.3.2.

Initially, the new reinforcement approach is validated by comparing the results of a single-muscle SPA under extension testing with previously published data.

Subsequently, FEA simulations are conducted for the multi-muscle SPA detailed in Section 4.3.3. Four scenarios are explored to induce extension, bending, rotation, and twisting movements in various combinations.

The simulation results are then used to verify the outcomes obtained from the mathematical model (4.18) under identical scenarios. A comprehensive discussion follows, focusing on insights gleaned from the results, including curvature consistency, soft sensor length, twist in the length model, and the comparative sensitivity of spiral versus straight sensors.

Furthermore, a validation of the mathematical model (4.18) against previously published results is conducted to assess its accuracy. Finally, this section wraps up with a broader discussion about the limitations and potential applications of the mathematical model (4.18), incorporating examples from relevant literature.

4.4.1 Single-Muscle Extension Test

To validate the new reinforcement approach, an extension test was conducted in Abaqus, comparing the change in length with previously published results by [190]. The simulated sample, depicted in Figure 4.13(a), comprised a single pneumatic muscle with dimensions matching those of the sample used by [190]: a thickness, outside diameter d , and initial length L_o of 2 mm, 16.7 mm, and 165 mm, respectively.

The base was fixed, while the top end was free to move. Both the base and top surfaces were covered with a skin modelled as paper [345], with a density of 750 kg/m³, a Young's modulus $E_1 = 6.5$ GPa, and a Poisson's ratio $\nu_{12} = 0.2$. This paper skin

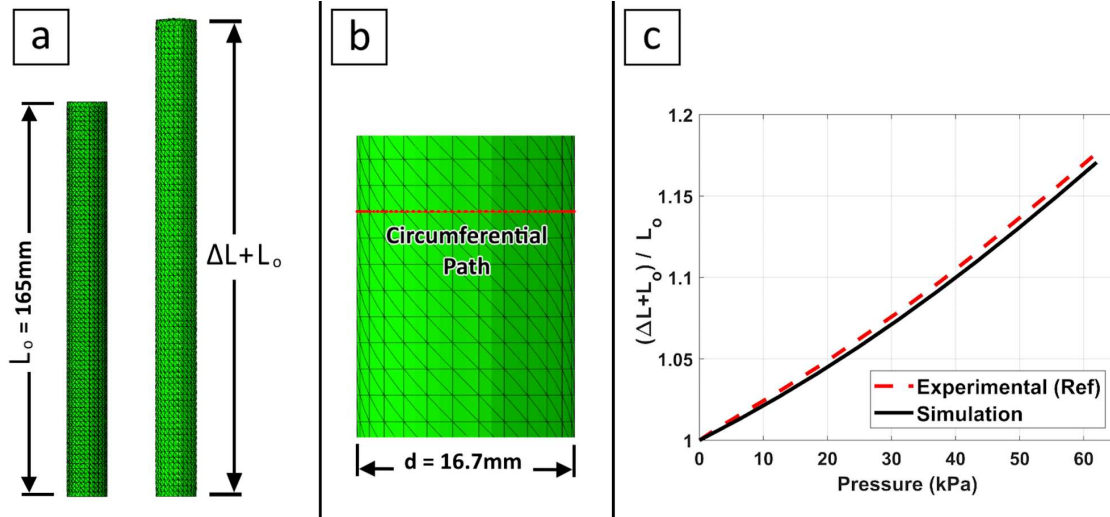


Figure 4.13: Extension test comparison, simulation in Abaqus versus results adapted from [190] experimental test where (a) depicts the initial length, (b) shows the circumferential path to measure change in diameter (if any), and (c) is the results comparison.

prevented radial deformation of the surfaces but maintained flexibility at connection edges. The same silicone material, Elastosil M4601, was used, and a pressure of 62.05 kPa was applied in the simulation.

For an optimal balance between accuracy and computation time, an approximate global mesh size of 2.0 was selected. A circumferential nodal path, shown in Figure 4.13(b), was created to measure its length at 0% and 100% pressure, ensuring no change in the pneumatic muscle diameter d . Data points were extracted from the results of [190, Figure 2a at 0° fibre angle] and plotted against the simulation results, as depicted in Figure 4.13(c).

The results demonstrated that the reinforcement layer closely matched the experimental data, with an error of less than 0.5%. Furthermore, the length of the circumferential path remained unchanged during extension, confirming the validity of the reinforcement approach. The muscle extended approximately 17% at full pressure, falling within the 20% extension limit with negligible diameter change, as discussed earlier in Section 4.3.1. Similarly, [190] reported an approximate diameter change of 1.5% in their test at full pressure, which could also be neglected. Finally, the Python code (`SingleMuscle.py`) used to generate the Abaqus model, shown in Figure 4.13(a,b), is provided in the supplementary materials of [74].

4.4.2 Multiple-Muscle Actuator

To strike a balance between accuracy and computation time, an approximate global mesh size of 5.0 was chosen. The simulation employed a *Static, General* analysis using the **Abaqus/Standard** solver, with a total simulation time of 1 second. Various tests were performed to induce extension, bending, rotation, and twisting movements in different combinations. The simulation values of pressures p_1 , p_2 , p_3 , and moment M are detailed in Table 4.1 for all scenarios. The initial lengths of the centreline, first, and second sensors were all set to 250 mm , while the third sensor

was set to 254.2 mm .

Additionally, the Python code (`MultiMuscle.py`) used to generate the Abaqus model, depicted in Figures 4.11 and 4.12, is provided in the supplementary materials of [74].

Table 4.1: Pressure values (p_1, p_2, p_3) in MPa and moment (M) values in $N.mm$ for different simulation scenarios involving bending (B), extension (E), rotation (R), and twist (T).

Simulation	Scenario	p_1	p_2	p_3	M
1	B,E	0.05	0.2	0.05	0
2	B,E,R	0.05	0.2	0.2	0
3	B,E,R,T	0.05	0.2	0.2	15,30,45
4	T,E	0.2	0.2	0.2	45

4.4.2.1 Simulation Scenarios

In the first simulation scenario involving bending and extension, the second muscle received a higher load. The SPA bent in the XZ plane, reaching a curvature of about 0.0052 mm^{-1} , Figure 4.14(a1,b1). The rotation ϕ and twist ψ angles were approximately zero, while the bending angle θ reached 76.3 deg , Figure 4.15(a1). The centreline length extended by 4.7%, Figure 4.15(b1).

For the second scenario, a higher pressure p_3 was applied, causing the SPA to rotate approximately 60 deg anticlockwise around the positive z -axis, in addition to bending and extension. This was observed in the projection of the SPA on the XY plane, Figure 4.14(a2). The bending angle θ and curvature were 82.5 deg and 0.0055 mm^{-1} , respectively. The centreline length extended by 7.3%, which is greater than the extension in the first scenario, reflecting the increased total pressure $p_1 + p_2 + p_3$. Similar to the first scenario, the twist angle ψ was approximately zero due to the absence of any twisting moment, Figure 4.15(a2).

In the third scenario, a twisting moment M was added to the configuration of the second scenario. This scenario was repeated three times with different moment M values: 15, 30, and 45 N.mm . The results of the latter is shown in Figures 4.14 and 4.15. The centreline length extended by an average of 7.4% in each simulation. The twisting moment caused an arching in the SPA centreline, deviating the bending from a plane, Figure 2.3(b). This arching is clearly visible via the SPA projection on the XY plane, Figure 4.14(a3).

The amount of arching was directly proportional to the moment M value. The dissimilar pressure loads in the muscles are believed to be the main cause of this arching. This implies that arching is expected whenever there is a twisting moment along with bending, generally speaking. However, different boundary conditions at the top end may generate different behaviours under moment loading. Investigating this arching behaviour is beyond the scope of this thesis. The twist ψ angle reached $38.7, 80.2, \text{ and } 121.7\text{ deg}$ for $M = 15, 30, \text{ and } 45\text{ N.mm}$, respectively.

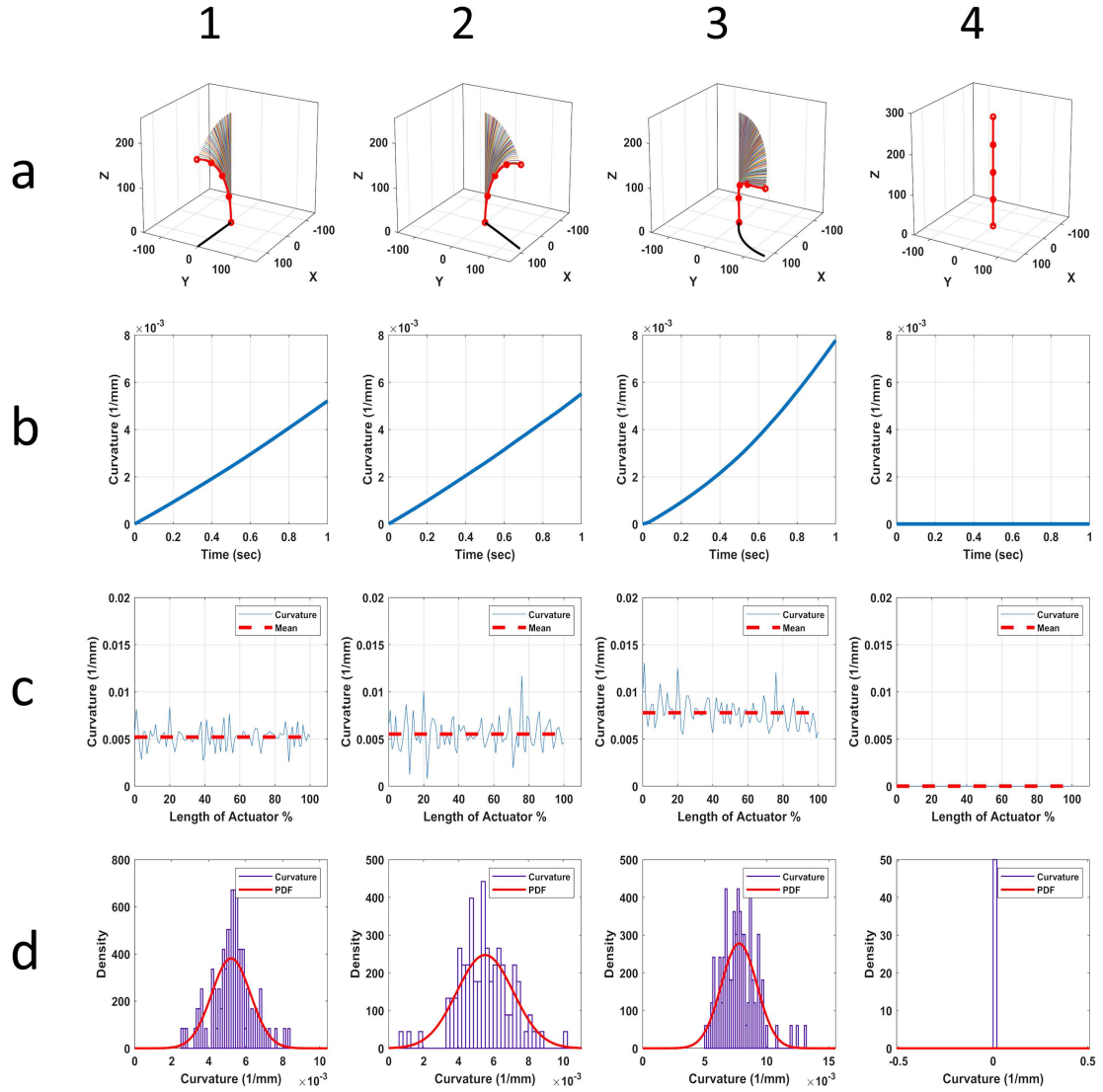


Figure 4.14: Simulation results for four scenarios: (a1 to a4) depict 3D plots of SPA positions and their XY plane projections as solid black lines, (b1 to b4) show curvature vs. time, (c1 to c4) display curvature across the SPA length at the last time frame along with its mean value, and (d1 to d4) present the PDF results for curvatures in (c1 to c4).

Additionally, the rotation angle ϕ increased noticeably from its initial value, from $60.6 \deg$ to $76.4 \deg$ for $M = 45 \text{ MPa}$, due to the arching effect, Figure 4.15(a3). This increase is anticipated because the angles were calculated based on the positions of the three tracking nodes at the top end surface, as shown in Figure 4.11(c).

The fourth and final scenario involved twist and extension to study the effect of twist on sensor sensitivity [351], i.e., the advantage of spiral sensors model. Translation of the centreline in both x - and y -axes was restricted. The SPA extended by 10% along the positive z -axis with equal pressure loads on its three pneumatic muscles, $p_1 = p_2 = p_3 = 0.2 \text{ MPa}$. Additionally, the SPA rotated freely anticlockwise around the positive z -axis by applying a moment of $M = 45 \text{ N.mm}$. The twist angle ψ reached $130.9 \deg$, while bending θ and rotation ϕ angles were zero, Figure 4.15(a4), as expected due to the absence of bending or the neglected curvature, $\kappa \leq \kappa_{min}$.

4.4.2.2 Curvature Constancy

In analysing the soft sensor length, an examination of the SPA curvature in the simulation results was conducted to determine its consistency across various scenarios. However, the fourth scenario, lacking bending, necessitated the neglect of curvature. Curvature was computed at each time frame using (4.31). Figures 4.14(c1 to c4) illustrate the curvature distribution across the SPA length at the final time frame for each scenario.

In analysing the soft sensor length, fluctuations in curvature across the SPA length were observed. These fluctuations are anticipated due to inherent variability, influenced by the first and second derivatives, even after employing B-spline smoothing on the discrete Abaqus model results. Noticeably, despite these fluctuations, the curvature maintains relative constancy around a certain value across the SPA length, satisfying the required constancy criterion. To further validate this constancy, the probability density function (PDF) of curvature was computed, revealing a normal distribution as an indication of the constancy, Figures 4.14(d1 to d4).

The mean value of the normal distribution, represented by the dashed red line in Figures 4.14(c1 to c4), provides a comparative benchmark for the curvature. In the scenario where the SPA remained straight, namely the fourth scenario where curvature (κ) is less than κ_{min} , the mean of the normal distribution is expected to be less than κ_{min} . The curvature values depicted in Figures 4.14(b1 to b4) represent the mean of the normal distribution at each time frame, serving as input for the length model (4.18) to compute the length of the three soft sensors at each time frame for comparison with Abaqus results.

4.4.2.3 Length of Soft Sensors

The length of each soft sensor may either extend or reduce based on the actuation direction and the sensor's position, that is v_{si} and Δv_i . In the simulation, it was assumed that the sensors can measure both extension and reduction, achievable through methods such as using pre-stretched sensors. In the first scenario, see Figure 4.15(b1), the length of both the second and third sensors increased, while it decreased for the first sensor. Similarly, in the second scenario, Figure 4.15(b2), the second sensor extended more than the other two, as the SPA bending occurred

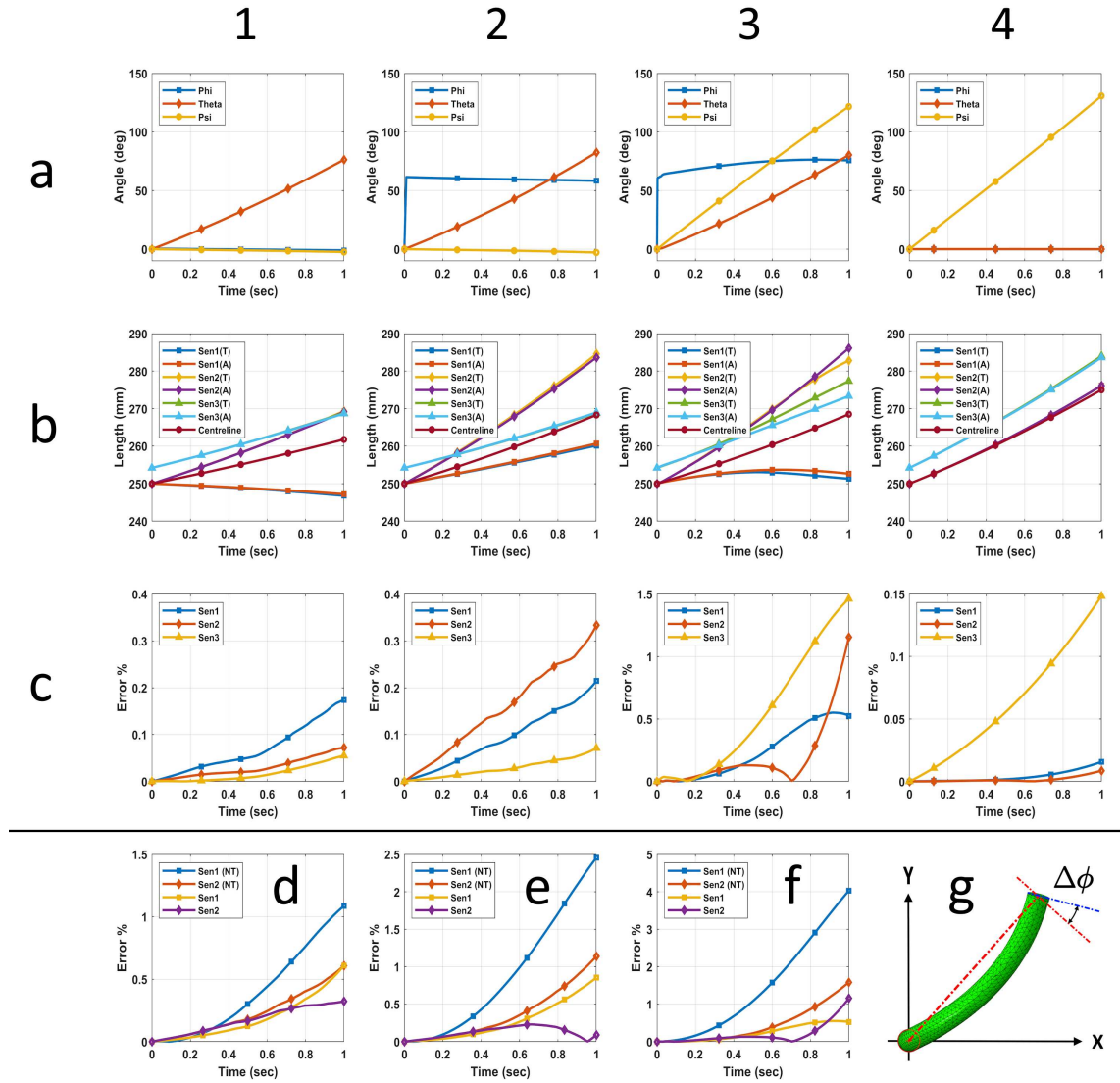


Figure 4.15: Simulation results: (a1 to a4) Rotation ϕ , bending θ , and twist ψ . (b1 to b4) Centreline length and comparison of soft sensor lengths between Abaqus (A) and theoretical (T) results. (c1 to c4) Percentage error of theoretical model compared to Abaqus results. The last set of figures (d, e, f) show error comparisons between models with and without twist (NT) for the third simulation scenario at $M = 15, 30$, and 45 MPa, respectively. (g) Top view of the SPA demonstrating the arching effect on rotation angle ϕ .

opposite to the direction of the second sensor in which $\phi \approx 60 \text{ deg}$.

In the third scenario, the arching caused a noticeable increase in error, especially for the third sensor, Figure 4.15(c3). The spiral geometry of the third sensor made it more sensitive to the arching compared to the first and second sensors. However, the length model (4.18) maintained high accuracy despite the unmodelled arching. The length errors of the third sensor with respect to Abaqus results were approximately 0.7%, 1.3%, and 1.5% at twisting moments $M = 15$, 30, and 45 $N.mm$, respectively. The latter is shown in Figure 4.15(c3).

Furthermore, the relation between curvature, centreline, and sensor length versus time were linear in both the first and second scenarios, Figures 4.14(b1, b2) and 4.15(b1, b2), where no arching occurred. Non-linear behaviours started to emerge in the third scenario, Figures 4.14(b3) and 4.15(b3), which could be further investigated in future works to gain more insights about arching. The advantage of including twist in the length model of soft sensors is discussed in Section 4.4.2.4 below.

A distinct trend for the length error is evident in the third scenario, Figure 4.15(c3), where the error rate, i.e., the slope of the error curve, initially increased and then decreased for all sensors. This trend is believed to be related to the arching effect and to have a direct relationship with the value of the twisting moment M . Further changes in the error rate are observed in the results of the second sensor, where the rate increased again as shown in Figures 4.15(d, e, f). Such further changes are speculated to be related to the loading scenario. The second sensor is positioned between the first and second muscles where higher pressure was applied, that is p_1 and p_2 . The combined effect of arching and the sensor's location led to this distinctive trend in the length error. Further investigation is required to study the potential maximum error values and the trends they may exhibit under different loading scenarios.

The fourth and final scenario aimed to demonstrate the relationship between the twist angle ψ and sensor length. The extension of the soft sensors was a combination of overall SPA extension and twist, Figure 4.15(b4). The third sensor exhibited more significant changes in length due to its spiral geometry, which is more sensitive to twist. While identical results were expected for both the first and second sensors, a minor difference in the length errors was observed, which is attributed to the accuracy of the Abaqus model, Figure 4.15(c4). Nonetheless, both errors are very small and likely negligible in experimental or real-life applications. For all scenarios except the third, the length error of any sensor did not exceed 0.33%, Figures 4.15(c1, c2, c4), confirming the validity and high accuracy of the length model (4.18).

4.4.2.4 Twist in the Length Model

To demonstrate the advantages of including SPA twist in the length model (4.18), the lengths of the first and second sensors were calculated by omitting the twist from the length model, i.e., $\psi = 0$ at all times, in both the third and fourth scenarios. The first and second sensors were selected since they are straight at the initial position. Omitting the twist for straight sensors simplifies the length model to resemble previous studies in literature. For example, the study of [21] can be referenced, in which the four arcs in their model can be treated as four soft sensors.

By comparing to Abaqus results, the percentage of length error when omitting twist in the third scenario at twisting moments $M = 15, 30$, and 45 N.mm is shown in Figures 4.15(d, e, f), respectively. The arching in the third scenario caused a deviation of $11.3, 21.2$, and 27.4 deg in rotation angle ϕ at twisting moments $M = 15, 30$, and 45 N.mm , respectively, Figure 4.15(g). Another deviation is observed in the bending angle θ , which is not shown, but explains the slight difference between the mean and the curvature trend after 80% of SPA length in Figure 4.14(c3). It is believed that arching is one of the reasons for the error difference between including or omitting twist. However, analysing the effects of arching falls outside the scope of this thesis..

The first sensor exhibited higher errors based on its location, the state of the SPA (i.e., bending, rotation, and extension), and the amount of twisting moment. In the third scenario, the error for the first sensor reached 4% when twist was omitted, compared to an actual error of 0.55% in the length model (4.18) with respect to Abaqus results, as shown in Figure 4.15(f). Furthermore, in the fourth scenario, the error when twist was omitted was 0.4% and 0.41% for the first and second sensors respectively, Figure 4.16(b), compared to negligible error values of $1.6 \times 10^{-2}\%$ and $8.5 \times 10^{-3}\%$, respectively, in the length model (4.18) with respect to Abaqus results. This difference is substantial and demonstrates the advantage of including twist in the length of soft sensors.

Additionally, including twist in the length model allows for the modelling of spiral strain sensors. Moreover, with twist in the model, the SPA state as well as the twisting moment M , if unknown, can be estimated using state estimation algorithms such as the Kalman Filter [352]. It is important to highlight that the first and second sensors should have similar error values in the fourth scenario since there is no bending or rotation. The difference in errors was insignificant compared to the error values when twist was omitted. However, it was significant compared to the negligible errors in the length model (4.18). Nonetheless, the error values are very small to be considered, and I believe the difference is due to the selected FEA mesh size.

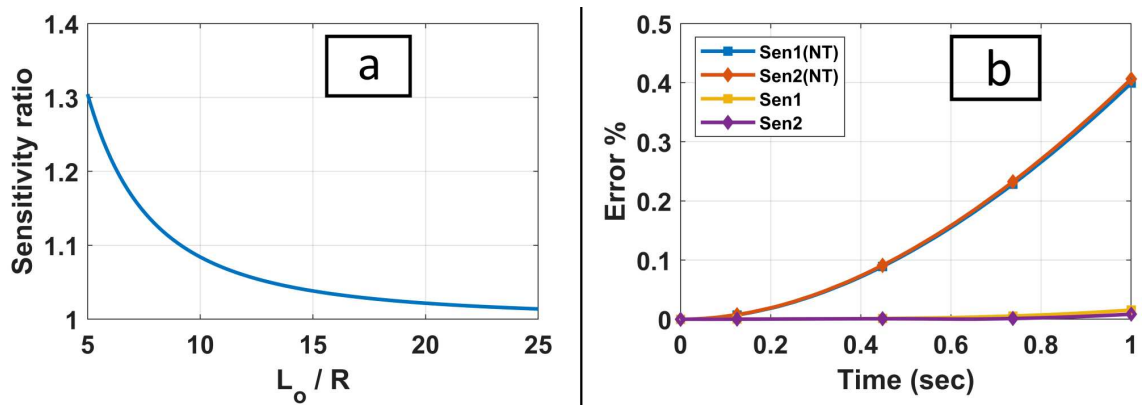


Figure 4.16: Comparison between spiral and straight sensor geometries: (a) depicts the sensitivity ratio vs. SPA's length-to-radius ratio, and (b) shows the error comparison between including and excluding twist (NT) in the theoretical model for the fourth simulation scenario.

4.4.2.5 Sensitivity

The sensitivity of a sensor denotes how its output varies with changes in the measured input quantity [351]. In extension simulations, both spiral and straight sensors measure changes in the SPA's centreline length, resulting in a sensitivity ratio that mirrors the length ratio. To put it simply, longer sensors exhibit higher responsiveness to fluctuations in centreline length within a given SPA. Moreover, the length of spiral sensors expands in conjunction with the SPA's cross-section radius.

To illustrate this principle, the sensitivity ratio of spiral to straight sensors is plotted against the SPA's length-to-radius, L_o/R , ratio at its initial position, Figure 4.16(a). The geometric characteristics of the straight sensor mirror those of either the first or second soft sensors in the FEA simulation. Meanwhile, the spiral sensor's geometry aligns with that of the second soft sensor in the FEA simulation, characterised by $\Delta v_i = \frac{4}{3}\pi$. With the SPA exhibiting an L_o/R ratio of approximately 22.7, the resulting sensitivity ratio stands at 1.017.

Despite this marginal difference in sensitivity, the spiral sensor demonstrates a significantly greater change in length compared to the straight sensors, Figure 4.15(b4). Consequently, employing spiral sensors may provide advantages dependent on the specific SPA and sensor configurations, as well as the nature of the testing scenarios. Notably, the sensitivity of spiral sensors escalates with the utilisation of sensors featuring a larger spiral span Δv_i or an SPA characterised by a smaller L_o/R ratio. Determining the optimal L_o/R ratio is based on the unique requirements of the application [353] and the preferences in design.

4.4.3 Model Validation

In addition to the previous validation outlined in Section 4.4.1, ensuring the accuracy of the model (4.18) requires further validation through experimental means. The process of experimental validation encompasses several crucial tasks, including the meticulous design of both the actuator and the sensors, the formulation of an effective controller, and the precise calibration of data acquisition and motion capture systems. The tasks' nature may vary depending on the specific test being conducted, which is important to note.

To showcase the versatility and accuracy of the length model (4.18), a comparative analysis was conducted between theoretical predictions and previously published experimental results. A comprehensive literature review undertaken as part of this thesis revealed a scarcity of relevant research pertaining to PCC-based soft actuators, with only a few exceptions such as [36, 144, 145], which were discussed in Chapter 2, Section 2.2.3.

It's worth noting that while there may be limited studies directly relevant to the investigation, other existing research can be leveraged for comparison, provided that the PCC model remains consistent across experimental tests. One such study, conducted by [143], presents an insightful examination involving three distinct tests: bending, twist, and deformation reconstruction. The experimental setup involved two multi-core optical fibres, one featuring straight cores and the other with spiral cores. Each sample comprised three peripheral optical fibre cores, labelled as $OF1, OF2, OF3$, positioned 120 deg apart, alongside a fourth central core OFC

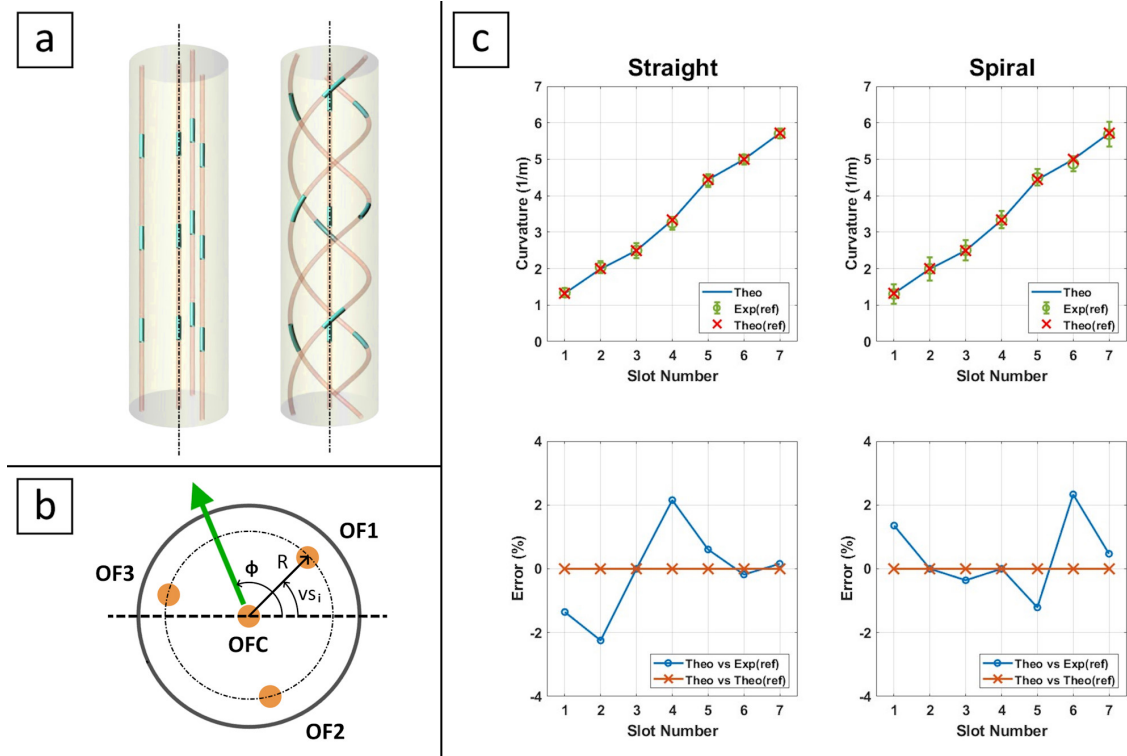


Figure 4.17: Bending test of the multi-core optical fibres, showing (a) samples of the straight and spiral cores, (b) a cross-sectional view illustrating the peripheral and core fibres along with an arbitrary bending direction marked with a green vector, and (c) comparisons between the theoretical and experimental curvature results for both the straight and spiral cores.

specifically designed for compensating the temperature variations, as illustrated in Figure 4.17(a, b). Both samples were equipped with multiple sets of FBG sensors inscribed within the cores, connected to an interrogator responsible for providing the light source and analysing the reflected spectrum to measure local strain in each FBG sensor.

The peripheral optical fibres are situated at a radius of $35 \mu m$. Both samples exhibit a sensorised length of 115.5 and 175 mm for the straight and spiral fibres, respectively. Notably, the spiral fibres feature 50 turns per unit length. In the modelling approach in this thesis, the optical fibres were represented as comprising three straight sensors and three spiral sensors, all positioned at a radius of $R = 0.035 mm$. The three individual sensors in both the straight and spiral samples begin at coordinates $v_{si} = 0, -\frac{2}{3}\pi, \frac{2}{3}\pi$, for the first, second, and third sensors, respectively. The straight sensors do not exhibit any coordinate difference, $\Delta v_i = 0$, whereas the spiral sensors have a coordinate difference of $\Delta v_i = \frac{35}{2}\pi$, determined by the number of turns per unit length.

For the bending test, the authors utilised a test bench constructed from acrylic board, comprising 7 slots with varying curvatures ranging from 1.33 to $5.71 m^{-1}$. Remarkably, this test bears resemblance to either the first or second simulation scenario discussed in Section 4.4.2.1, primarily focused on rotation ϕ angle. Curvature values and corresponding measurements were extracted from the published

results [143, Fig. 5(a)], and these values were used to calculate strain results for both straight and spiral sensors using the length model (4.18).

Subsequently, theoretical curvature results were calculated using [143, Eq.7], facilitating comparison with the authors' model and experimental findings. Notably, the rotation angle can be set to $\phi = 0$, as its value does not impact the results, given that [143, Eq.7] computes the magnitude of the curvature vector, coupled with the fact that both samples lack fixed ends.

The resultant comparison findings are visually depicted in Figure 4.17(c), while the MATLAB code (FBG_SNA2021.m) and the extracted data (FBG_SNA2021_data.m) utilised in generating the results are conveniently accessible in the supplementary materials of [74].

The comparison reveals proper alignment between the theoretical results of the length model (4.18) and the reference study [143]. It's important to note that the bending model in [143] is a simplified version compared to (4.18), as it excludes considerations for centreline length extension, rotation, and twist. Despite this difference, comparison with experimental results demonstrates a close match, with a mean error of less than $\pm 2.4\%$, taking into account manufacturing and measurement inaccuracies.

This comparison not only validates the accuracy of the length model (4.18) but also highlights its flexibility in representing various tests, provided the PCC model is maintained.

For the twist test, the authors of [143] employed a linear model [143, Eq.8] correlating twist angle with strain. This model includes a *material constant*, denoted as ' G ' in [143], that is calculated experimentally. However, this approach has limitations in accurately representing twist due to the non-linear relationship between strain and twist angle [354]. Additionally, conducting bending tests along with twist necessitates calculating the *material constant* for each curvature, introducing further complexity. The considerable error evident in the results of the twist test [143, Fig.6(a, b)] underscores these limitations. Lastly, the deformation reconstruction test falls beyond the scope of this thesis.

4.5 Conclusion

A PCC-based soft actuator can undergo various modes of deformation, including extension, bending, rotation, and twist. While existing models for straight strain sensors may adequately capture extension, bending, and rotation, they often fall short when it comes to accurately measuring twist, leading to potential errors in measurements. Despite the limitations inherent in the PCC model, an approach was developed in this study to represent spiral sensors and account for twist in SPAs. This method, grounded in the concepts of geodesics and covering spaces, provides a simplified and streamlined alternative compared to the approach presented in [151].

A simplified FEA model was built for the SPA, and various simulation scenarios were designed to validate the accuracy of the proposed length model (4.18). An orthotropic layer technique was introduced to model the reinforcement of pneumatic muscles, effectively reducing the computational burden of the FEA simulations by

eliminating the need for complex *contact interactions* in Abaqus. Additionally, a procedure was outlined for calculating the centreline length and soft sensor lengths from the [FEA](#) model using B-splines to smooth discrete results.

Furthermore, the Kabsch algorithm was implemented to compute the bending θ , rotation ϕ , and twist ψ angles from the [FEA](#) model data. This comprehensive [FEA](#) simulation methodology provides a valuable tool for researchers to develop customised models, explore [SPA](#) deformations in depth, streamline simulation efforts, and optimise resource utilisation in both pre-processing and post-processing phases. Moreover, the codes developed as part of this study facilitate efficient data extraction to MATLAB, further enhancing the accessibility and utility of the proposed techniques for the research community.

The results demonstrate the capability of the length model (4.18) to accurately represent both straight and spiral sensors across various testing scenarios. The error in length measurements for the soft sensors remained below 0.33% in both the first and second simulation scenarios. Introducing a twisting moment at the free end induced arching in the [SPA](#). Despite this arching, the error in length measurements did not exceed 1.5%, indicating that the [PCC](#) approximation remains applicable.

It's worth noting that the accuracy of the length model (4.18) hinges on the suitability of the [PCC](#) model for a given application. Higher twisting moments may lead to increased arching, potentially rendering the [PCC](#) model unsuitable. In comparison to the [CRM](#) and [PCS](#) models, the [PCC](#) model stands out as a preferable choice for real-time control, thanks to its simplicity in kinematic modelling [68]. However, assessing the suitability of the [PCC](#) model for specific applications falls beyond the scope of this thesis.

In comparing the exclusion versus inclusion of twist in the soft sensor modelling, it was observed that twisting the free end led to a 7.3 times larger error in the third scenario and a 48.2 times larger error in the fourth scenario. These significant errors underscore the advantage of utilising the length model (4.18) to represent both straight and spiral soft strain sensors across various configurations. Spiral sensors exhibit higher sensitivity compared to straight sensors within the same [SPA](#) due to their greater length. The model (4.18) offers the capability to leverage them in the development of soft sensing applications.

This sensitivity can be further enhanced by increasing the length of the spiral sensor through adjustments in the spiral span Δv_i or the [SPA](#)'s length-to-radius, L_o/R , ratio. Lastly, experimental validation was conducted on the length model (4.18) by comparing theoretical results with previously published data to demonstrate its accuracy and versatility. Subsequently, limitations of the length model (4.18) were discussed, along with potential applications.

Collectively, the length model (4.18), [FEA](#) model, and testing scenarios introduce a novel tool for enhancing proprioceptive sensing in various [PCC](#)-based soft actuators and may pave the way for future implementations, particularly in real-time control applications.

Chapter 5

Numerical Modelling of Uniaxial EGaIn-Based Soft Strain Sensors

In this chapter, the numerical modelling of uniaxial soft strain sensors utilising [EGaIn](#) liquid metal is explored. Soft sensors based on [EGaIn](#) exhibit remarkable sensitivity, stretchability, and stability, attributes that contribute to their reliability and adaptability across a spectrum of applications [272–274]. The primary focus of this chapter lies in the numerical modelling of these sensors, aimed at achieving high-precision strain estimation within sub-millimetre error margins.

After reviewing the complexity and manufacturing cost of several [EGaIn](#)-based soft sensors in the literature, one particular sensor design, as illustrated in Figure 5.1, was chosen for analysis in this study, fabricated as per the manufacturing technique presented by [242].

A comparative analysis between experimental data and the analytical model is then presented, shedding light on the limitations of the latter in accurately representing experimental outcomes. This prompts the adoption of regression analysis principles to construct a numerical model with sub-millimetre error tolerance, following the validation protocol elaborated on in Chapter 2, Section 2.3.3.

Subsequent to this, the chapter delves into exploring the influence of strain rate on hysteresis and proposes strategies to mitigate hysteresis-induced errors in strain estimation based on resistance measurements. The chapter features extensive discussions that appear at various points, contributing to a comprehensive examination of the topic, ultimately leading to a conclusive summary that includes a brief overview of limitations.

5.1 Geometrical Design of the Selected Sensor

The selected sensor, depicted in Figure 5.1, exhibits an active initial length of $L_{cc}^o = 76.3\text{ mm}$ marked between the connection centre points, where strain will be applied. The remaining length of the sensor remains inactive, contributing to the sensor's structural integrity and overall functionality. This active length is subdivided into two primary regions, each serving distinct purposes within the sensor's operation.

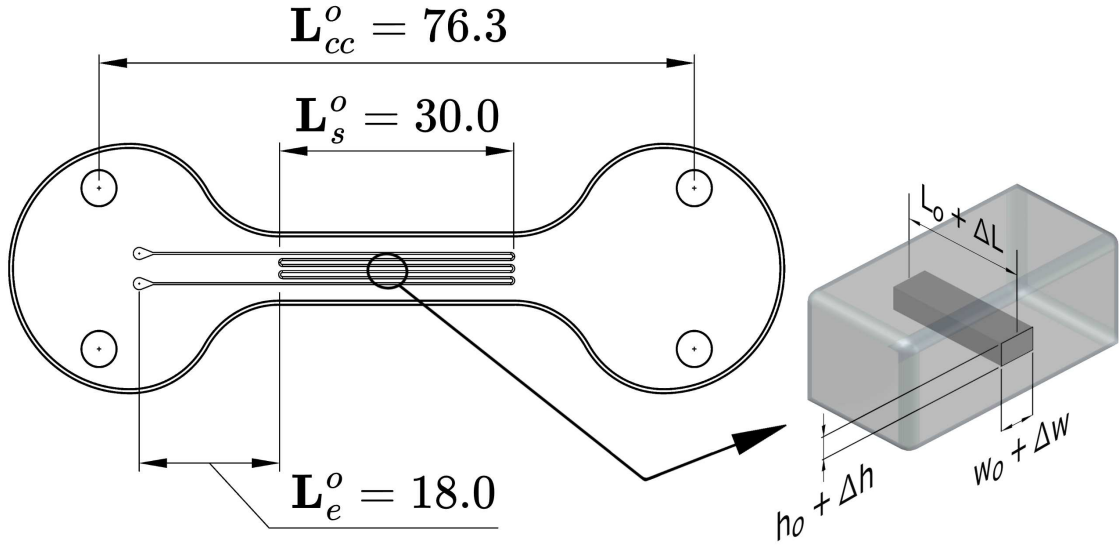


Figure 5.1: Geometrical design of the selected sensor, showing key dimensions: L_{cc} for the centre-to-centre distance, L_s for the length of the sensing element, and L_e for the remaining length of each connection end. All dimensions, denoted with the superscript " o ", are in millimetres and represent the initial values before any strain is applied. The 3D view displays a cross-section of an [EGaIn](#) microchannel during stretching, with h_o and w_o representing the initial height and width, and L_o representing the initial microchannel length.

The first region comprises two recognisable subregions: the sensing element and the connection microchannels. The sensing element, with an initial length of $L_s^o = 30.0\text{ mm}$, constitutes the portion of the sensor directly responsible for detecting and measuring applied strain. In contrast, the connection microchannels, spanning an initial length of $L_e^o = 18.0\text{ mm}$, facilitate the seamless integration of measuring devices for monitoring sensor resistance.

Moving beyond the sensor's active length, the second region consists of the remaining length within L_{cc} that lacks microchannels. This section of the sensor facilitates the transmission of applied strain from the centre points to the sensing element.

The microchannel, characterised by a rectangular cross-section, features initial theoretical dimensions of $w_o = 0.25\text{ mm}$ for width and $h_o = 0.15\text{ mm}$ for height. As [EGaIn](#) is an incompressible material, its volume remains constant throughout deformation. This fundamental property is encapsulated in the equation:

$$V = L_o A_o = LA, \quad (5.1)$$

where L and A represent the length and cross-sectional area of the microchannel, respectively, with the subscript " o " denoting initial values. The equation highlights the conservation of volume during deformation, underscoring [EGaIn](#)'s suitability for strain sensing applications.

The strain-induced changes in the sensor's dimensions directly impact its electrical resistance, necessitating a comprehensive understanding of the relationship between strain and resistance. While earlier models explained this relationship, a simplified

theoretical model proposed by [269] offers a pragmatic framework for analysing sensor behaviour. This model, expressed as:

$$\frac{\Delta R}{R_o} = \epsilon(2 + \epsilon), \quad (5.2)$$

quantifies the change in resistance ΔR relative to the initial resistance R_o as a function of applied strain ϵ . The sensor's resistance R itself is determined by the formula:

$$R = \rho \frac{L}{A}, \quad (5.3)$$

where ρ denotes the resistivity of [EGaIn](#) ($29.4 * 10^{-8} \Omega.m$). The initial resistance R_o is calculated based on the initial microchannel length, given by:

$$L_o = 6 * L_s^o + 2 * L_e^o = 216 \text{ mm}. \quad (5.4)$$

The deformed length L is subsequently determined in terms of the active length L_{cc} at which the input strain was applied, as detailed in Section 5.3.2 below. The model expressed in (5.2) traditionally serves to establish a baseline for strain measurement, which can be used in characterising the sensor behaviour under different strain conditions.

5.2 Experimental Set-Up and Testing Procedure

This section presents a thorough overview of the experimental set-up, also known as the testing rig, and testing procedure employed in this study. The description begins with an explanation of the mechanical framework of the testing rig, crucial for achieving precise control over input strain. By detailing these components, insight is provided into the mechanical precision of the set-up and its reliability in ensuring consistent experimental conditions.

Then, the features of the [DAQ](#) system are explored, highlighting its fundamental principles and capabilities in facilitating resistance measurement. This description emphasises the synergy between the [DAQ](#) system and the mechanical framework in capturing accurate data essential for the analysis.

Additionally, within this section, the testing procedure is outlined in meticulous detail. The outlined procedure offers a detailed account of the experimental protocol used in this study.

By providing a comprehensive overview of the testing procedure, valuable context is offered to readers for interpreting the acquired results and understanding their implications within the broader scope of this study.

5.2.1 Mechanical Framework of the Testing Rig

The mechanical framework of the testing rig comprises a sophisticated slider mechanism complemented by two 3D-printed brackets labelled A and B. These brackets are strategically positioned at the fixed and movable ends of the slider, as illustrated in Figure 5.2(a). The sensor terminations are securely affixed to both brackets, with the sensor's ends aligned and attached to the small pins on the brackets, ensuring stability and precision during strain application.

Central to the mechanical system is a NEMA23 stepper motor featuring a step angle of 1.8° , coupled with a screw mechanism boasting a theoretical lead of 5 mm/rev . Driving the stepper motor with fine accuracy and control is the GeckoDrive GM215 module, seamlessly integrated with a microcontroller (Arduino Mega 2560).

This integration enables the generation of customised strain profiles and precise control over testing speeds through hardware-generated pulse-width modulation (PWM) signals, as depicted in Figure 5.2(b). Leveraging this technique guarantees a stable PWM signal and consistent speed, critical for reliable and reproducible experimental results.

The GeckoDrive GM215 module offers versatile operation, providing options for full, half, and micro-stepping modes, corresponding to step resolutions of 200, 400, and 2,000 *steps/rev*, respectively. Consequently, the theoretical linear precision of the testing rig varies accordingly, with resolutions of 25, 12.5, and $2.5\text{ }\mu\text{m}$, respectively. For a comprehensive overview of the complete testing rig set-up, please refer to Figure 5.2(b).

5.2.2 Data Acquisition (DAQ) System

Previous studies [241, 242, 284] employed a constant current technique for measuring sensor resistance. However, stability issues arose with this technique during experiments, particularly in response to slight variations in laboratory temperature, which remained below 2°C . Consequently, a constant voltage technique was adopted, resulting in improved stability in resistance measurement.

The custom-built DAQ system comprises three sections, Figure 5.2(c). The first section features a voltage regulating unit equipped with an LM7805 chip, ensuring a constant 5 V supply to the subsequent sections. The second section incorporates a 24-bit differential ADC, utilising the ADS1256 chip from Texas Instruments.

The ADC measures the resistance by reading the local voltage across the sensor ends at various sampling speeds. Additionally, the ADC incorporates a dedicated circuit for filtering low frequencies and ripple noises. It also includes a high-precision benchmark chip (ADR431) by Analog Devices, providing a reference voltage of 2.500 V for accurate measurements.

The third section of the DAQ system comprises a voltage divider circuit employing two resistors, R_1 and R_2 , in series with the sensor. Both resistors have a nominal value of $220\text{ }\Omega$, which was determined to be optimal for the input range of the ADS1256 chip.

The sensor resistance R_x can then be calculated using the equation:

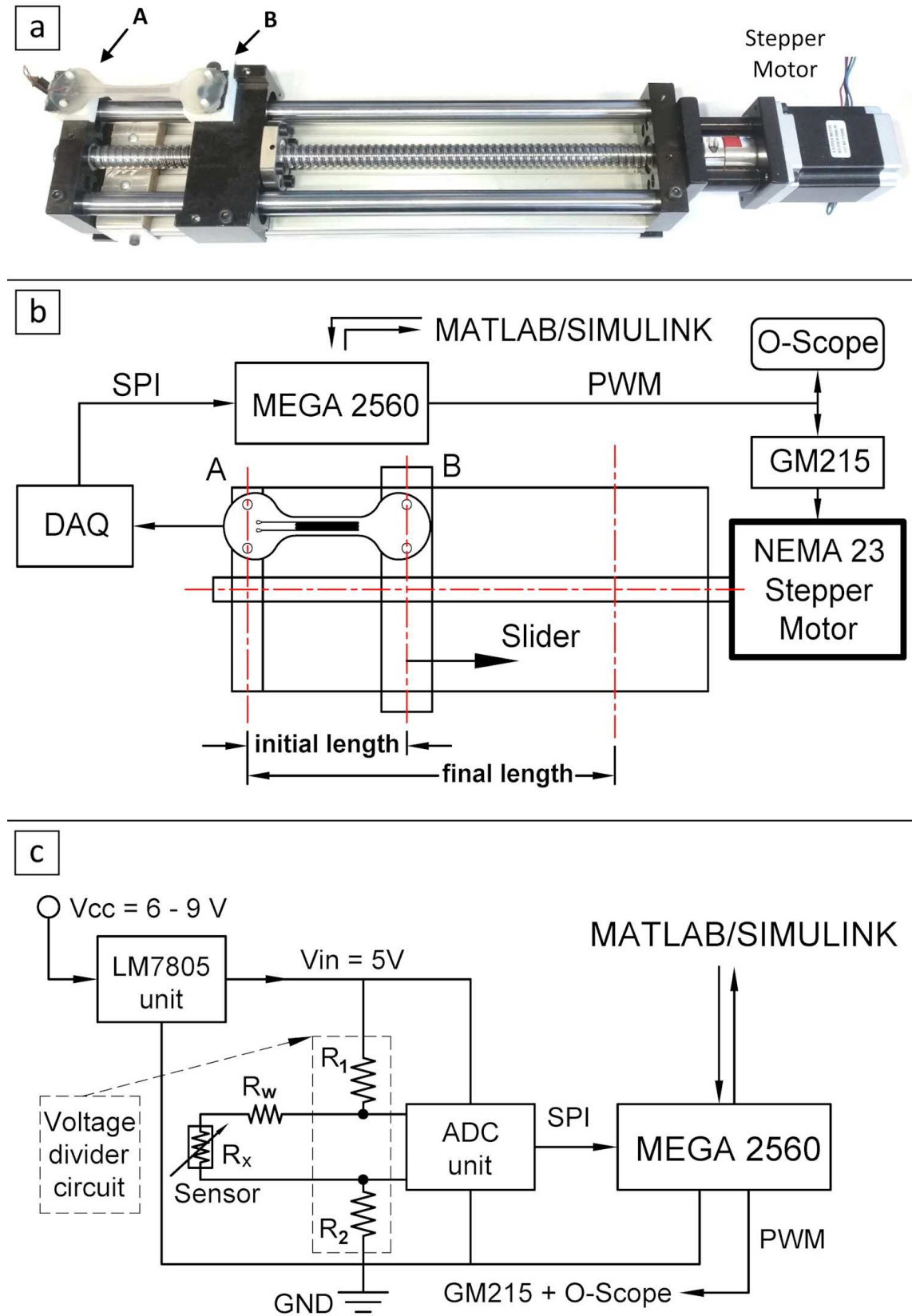


Figure 5.2: The components of the experimental set-up, also known as the testing rig: (a) Showcases the mechanical framework, with a soft sensor connected to brackets A and B mounted on the slider. (b) Provides an overview of the entire testing rig, while part (c) illustrates the electrical diagram of the DAQ system.

$$R_x = (R_1 + R_2) \frac{V_x}{V_{in} - V_x} - R_w. \quad (5.5)$$

Here, R_w represents the resistance of the wires connecting the sensor to the DAQ system, V_x denotes the DAQ measurement, and R_1 , R_2 , and V_{in} are as illustrated in Figure 5.2(c). The DAQ system transmits measurements via an SPI protocol to the Arduino microcontroller, ensuring synchronisation between strain and resistance measurements, Figure 5.2(b,c).

Although the DAQ system is capable of reading at 30k samples per second (SPS), it was configured to operate at 2k SPS. This setting allowed for averaging readings, with each acquired measurement (sample) representing an average of 15 readings. It was determined that 2k SPS struck a suitable balance between measurement speed and accuracy. Before acquiring all measurements, a thorough calibration process was conducted for both the mechanical framework and the DAQ system.

5.2.3 Calibration of The Experimental Set-Up

For the successful training of the numerical model, as detailed in Section 5.3.4 below, it is imperative to synchronise strain and resistance measurements. This synchronisation is orchestrated by the control algorithm implemented within the Arduino code. While the DAQ system handles resistance measurement, achieving precise strain measurements poses a challenge due to the need for a certain accuracy to match the precision of resistance measurements.

While initially opting for absolute linear encoders primarily for their potential to reset the slider reliably to the initial position, the calibration process uncovered significant accuracy limitations compared to the study requirements. Despite aiming for a linear precision of $2.5 \mu m$, the performance of these encoders fell short during testing.

This discrepancy prompted a reassessment of the strain measurement approach and highlighted the need to explore other options. Despite this change in the measurement approach, which required reallocating resources and effort, it ultimately enhanced the robustness of the experimental setup. This experience underscores the iterative nature of scientific enquiry, where adjustments and refinements are integral to achieving optimal outcomes.

An alternative consideration was rotary encoders, known for their ability to provide highly accurate measurements. However, adopting rotary encoders would necessitate an additional *homing sequence*, along with the incorporation of related sensors, to reset the slider's location to the reference position before each test. Such a modification would entail significant adjustments to both the mechanical framework and the control algorithm, thereby substantially increasing the complexity of building and operating the experimental set-up.

An alternative method for obtaining position feedback involved calculating the steps of the stepper motor within the Arduino code. While this method provided a workaround for obtaining position feedback, it came with its own set of challenges. One significant limitation was the considerable time and effort required to manually

reset the slider location to the exact initial position before each test in order to ensure result consistency.

This process proved to be labour-intensive and time-consuming, especially considering the extensive duration of the experimental testing period spanning several weeks. Despite these challenges, maintaining a consistent reference start within a narrow tolerance of $10\ \mu\text{m}$ was imperative to ensure the reliability and accuracy of the experimental results. This meticulous approach underscored the commitment to achieving a high level of consistency in the experimental set-up, despite the inherent limitations of the chosen methodology.

To determine the actual lead of the slider screw, the GM215 module was set to half stepping. Using a digital function generator (Tektronix AFG3022), a total of 12,000 square-wave pulses at 1 kHz were generated as a PWM input for the GM215 to move the slider a distance of $150\ \text{mm}$. Subsequently, a calibrated digital calliper with an accuracy of $10\ \mu\text{m}$ was utilised to measure the actual distance. The measured actual screw lead was found to be $4.998\ \text{mm/rev}$, a value crucial for the subsequent training of the numerical model.

In regards to the DAQ system, the actual values of the voltage divider resistors R1 and R2 were measured using a digital multimeter (Tektronix DM2510). The resistors were found to have respective values of $220.8\ \Omega$ and $220.6\ \Omega$. Special care was taken in selecting connections and wires between the sensor and the DAQ circuit board to minimise connection resistance. The total resistance of the connection, R_w , was measured and found to be $0.04\ \Omega$, representing an additional resistor connected with the sensor in series, as depicted in Figure 5.2(c).

Finally, the input voltage V_{in} was logged at each test to be used for calculations, ranging between $4.968\ \text{V}$ and $4.969\ \text{V}$ within a laboratory temperature range of $24 - 26^\circ\text{C}$, ensuring experimental consistency.

5.2.4 Strain Along The Sensing Element Length

As detailed in Section 5.1, the sensing element constitutes the region of the sensor responsible for measuring applied strain, housing the EGaIn microchannels, as depicted in Figure 5.3(a). To facilitate comparison with findings from prior literature, it's imperative to establish a relationship between the local strain along the sensing element length denoted by L_s and the input strain applied along the active length represented by L_{cc} .

Several soft sensors were fabricated for the purposes of this study. One soft sensor was selected randomly due to identical specifications. Employing the testing rig, the chosen sensor was subjected to incremental stretching from its initial length of $76.3\ \text{mm}$ up to $146.3\ \text{mm}$, with increments of $10\ \text{mm}$. At each increment, the sensing length L_s was manually measured using the digital calliper. The resulting measurements are depicted as red markers in Figure 5.3(b), with an illustrative representation provided in Figure 5.3(c), showcasing the manual stretching of the soft sensor to exhibit changes in the sensing element length L_s .

These gathered measurements can be utilised to derive a quadratic equation, which is modelled using the 'poly2' function in MATLAB, and can be expressed as:

$$L_s = a_1 L_{cc}^2 + b_1 L_{cc} + c_1, \quad (5.6)$$

where the resulting fitting parameters are: $a_1 = -0.001182$, $b_1 = 1.035$, and $c_1 = -42.25$, with an [RMSE](#) of 0.2396 mm and a goodness of fit $R\text{-Squared} = 0.9998$.

Eq. (5.6) will be employed in Section 5.3.2 to compare the theoretical resistance change, based on the sensor geometry and applied strain, with the experimental resistance measurements obtained under the same strain conditions.

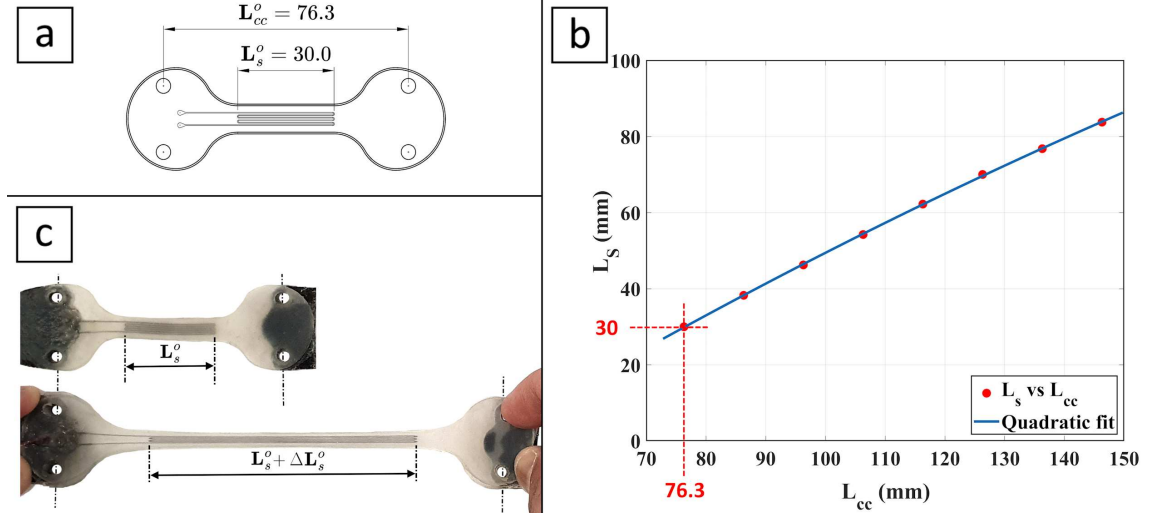


Figure 5.3: The relationship between the active length L_{cc} and the sensing length L_s under strain variation, depicted as follows: (a) illustrates both lengths when the sensor is at rest, (b) demonstrates the change in both lengths during incremental stretching using the testing rig, along with the quadratic regression model, and (c) displays a photograph of the change in sensing length L_s during manual stretching.

5.2.5 Testing Procedure

To investigate the relationship between resistance and strain, as well as strain rate, a meticulously designed testing procedure was implemented. Two sensors, S1 and S2, fabricated using identical geometrical designs, were subjected to the experimental procedure. Sensor S1 underwent a series of six tests with a consistent slider speed set at 100 mm/min . This velocity selection was informed by previously published findings [269, Figure 7f].

In contrast, Sensor S2 was subjected to thirteen tests, each performed at varying slider speeds, corresponding to different strain rates. These speeds ranged nominally from $1,000 \text{ mm/min}$ to 10 mm/min , as detailed in Table 5.1 and illustrated in Figure 5.5(b). The speeds were labelled numerically from No.1 (highest) to No.13 (lowest).

The speeds were manually chosen to encompass the entire range from highest to slowest, with particular emphasis on the lower speeds within the range. During the tests, the Arduino microcontroller encountered some [PWM](#) stability issues at specific speeds. Consequently, the [PWM](#) periods listed in Table 5.1 were manually adjusted and extensively tested to ensure smooth and stable operation. In addition,

a digital oscilloscope (Tektronix TDS2014) was employed to continuously monitor the stability of the PWM signal, providing instantaneous visual indications of any anomalies, as depicted in Figure 5.2(b,c).

Subsequently, the actual speeds were calculated based on the selected PWM periods and the actual lead of the slider screw. These speeds will be referenced later in Section 5.3.1. The strain rate is determined by dividing the actual speed by the initial active length L_{cc}^o and converting the time unit from minutes to seconds. Then, The total time for each run is calculated using the selected PWM periods and the timing sequence, which will be elaborated on later in this section. Refer to Figure 5.4 for a visual representation of the timing sequence.

The selected strain rate range encompasses values relevant to a wide spectrum of soft sensing applications. For example, [355] reported a strain rate range of 9 to 15%/sec when employing a soft sensor for human motion monitoring, while [356] documented a range of 0 to 37.4%/sec for detecting both large and small human body movements. Although these studies utilised non-EGaIn-based sensors, the reported strain rates are indicative of typical operational conditions across various soft sensing technologies. For a more comprehensive overview of strain rates in different soft sensing applications, readers are referred to [233, 234, 246, 357, 358].

Table 5.1: PWM periods and the corresponding actual testing speeds, along with the strain rate and the total time per run.

Test No.	PWM Period (μsec)	Stepping ($steps/rev$)	Target Speed (mm/min)	Actual Speed (mm/min)	Strain Rate ($\%/sec$)	Total Time per Run (sec)
1	740	400	1,000	1,013.1	22.13	13.1
2	820	400	900	914.3	19.97	14.1
3	940	400	800	797.6	17.42	15.6
4	1080	400	700	694.2	15.16	17.3
5	1248	400	600	600.7	13.12	19.3
6	1664	400	450	450.5	9.84	24.4
7	500	2,000	300	299.9	6.55	34.5
8	600	2,000	250	249.9	5.46	40.6
9	750	2,000	200	199.9	4.37	49.7
10	1,000	2,000	150	149.9	3.27	64.9
11	1,500	2,000	100	100	2.18	95.4
12	6,000	2,000	25	25	0.55	369.1
13	15,000	2,000	10	10	0.22	916.5

Notably, the results obtained from Sensor S1 were divided into two distinct sets: one comprising two-thirds of the total data for modelling purposes and the other

consisting of the remaining one-third for validation, following the validation protocol elaborated on in Chapter 2, Section 2.3.3.

Additionally, the measurements obtained from Sensor S2 were primarily utilised to explore the influence of strain rate on the observed resistance changes. This comprehensive testing regimen aimed to capture a nuanced understanding of the sensors' behaviour under varying strain conditions.

The testing procedure for each sensor included subjecting it to loading and unloading cycles. The sensor was stretched (loading) to achieve a strain of approximately 100% of its original length of 76.3 mm, and then returned (unloading) to its original length for four consecutive runs.

The total change in length was 76.0 mm (instead of 76.3 mm) as the resistance was chosen to be measured at each 1.0 mm interval during the length change. The first run served to overcome the Mullins effect [269] and was conducted without recording measurements. Subsequent runs involved measuring the resistance during both the loading and unloading phases.

The loading-unloading procedure was repeated three times to improve the reliability of the results. Resistance measurements were initially recorded at 0% strain and continued up to approximately 100% strain, resulting in a total of 77 measurements per direction or 154 measurements in total for each run. Collectively, there were 18 runs conducted for Sensor S1 and 39 runs for Sensor S2.

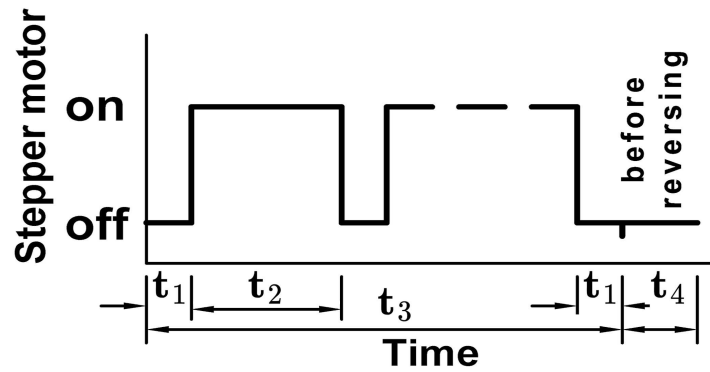


Figure 5.4: Visual representation of the timing sequence used in each run of the testing procedure.

The timing sequence is depicted in Figure 5.4. In this context, t_1 represents the time required by the DAQ to perform a measurement, approximately 788 μsec at a sampling rate of 2k SPS.

Next in the sequence is t_2 , representing the time needed to move 1.0 mm, calculated based on the selected PWM period as follows:

$$t_2 = \frac{\text{PWM period} * \text{Stepping}}{\text{Actual screw lead}}. \quad (5.7)$$

Then, t_3 indicates the total time required to transition from 0% to approximately 100% strain and complete the 77 measurements, expressed as:

$$t_3 = 76 (t_1 + t_2) + t_1. \quad (5.8)$$

Accounting for hysteresis resulting from viscoelastic behaviour, a delay of $t_4 = 4 \text{ sec}$ was determined to be necessary for the sensor to settle before reversing direction (unloading) and returning to 0% strain in each run.

Finally, the total time per run, denoted as t_{rn} , is calculated as follows:

$$t_{rn} = 2 t_3 + t_4. \quad (5.9)$$

The determination of the total time per run not only offers practical insights into the duration of each test but also serves as a valuable reference for researchers seeking to replicate or adapt the testing methodology outlined in this thesis. This information facilitates efficient planning and resource allocation for future experimental endeavours, contributing to the reproducibility and accessibility of the research findings.

5.3 Analysis and Discussion of Test Results

In this section, the testing results acquired from both sensors undergo comprehensive analysis to evaluate their performance in comparison to the theoretical predictions. Initially, the raw data acquired from the experiments is presented to offer insights into how the resistance evolves under different strain rates. This presentation aims to provide a clear understanding of the fundamental behaviour of the sensors and lay the groundwork for further analysis.

Following the raw data presentation, a detailed comparison is made between the theoretical predictions derived from the analytical model introduced in Section 5.1 and the actual experimental results. This comparison is essential for evaluating any disparities between the predicted and observed behaviours, thereby revealing the limitations of the analytical model.

Following this, a numerical model is introduced as an alternative approach to the analytical model, utilising regression analysis. This numerical model is accompanied by a thorough validation process to assess its accuracy and reliability. Furthermore, an exploration is undertaken to examine how changes in strain rate influence the accuracy of the numerical model, shedding light on its performance under varying conditions.

Finally, an algorithm for strain estimation based on resistance measurements is introduced, accounting for the observed hysteresis effect. This algorithm provides a practical framework for translating resistance measurements into accurate estimates of strain, thereby enhancing the utility of the sensor system.

5.3.1 Raw Experimental Results

The experimental results for both sensors were examined by plotting all loading and unloading measurements against the input strain in the L_{cc} length. This presentation of raw data, depicted in Figure 5.5(a), offers valuable insights into the behaviour of the sensors under the different loading conditions of this study.

The active length L_{cc} represents the length measured when mounting the sensor on the soft actuators. A notable observation from the raw measurements is the recognisable difference between sensors S1 and S2, despite both being subjected to the same strain. This variance can be attributed to discrepancies in microchannel sizes' accuracy and variations in the volume of liquid metal injected into each sensor.

To quantify this, the liquid volume in each sensor was calculated using equations (5.4), (5.3), and (5.1). Starting from the initial microchannel length of 216 mm (defined in (5.4)) and the measured resistance at 0% strain, the average cross-sectional area was determined using (5.3). This approach assumes that variations in the cross-sectional area along the microchannel do not significantly affect the result, as the constant liquid volume can be represented by a single effective cross-sectional area, supported by principles from integration (mean value theorem) and statistical averaging [349]. Finally, the liquid volume was determined using (5.1).

The actual volumes were determined to be 10.46 mm^3 for sensor S1 and 17.53 mm^3 for sensor S2, respectively, whereas the theoretical design volume is 8.1 mm^3 based on the geometrical design dimensions shown earlier in Figure 5.1. Theoretically, for the same microchannel length, a larger volume generates a thicker cross-section, resulting in lower resistance. This discrepancy may elucidate why sensor S2 exhibits lower resistance compared to sensor S1.

Regarding sensor S2, the effect of speed was found to be minimal but not negligible. The slowest speed, 10 mm/min , was chosen as the reference for studying the strain rate effect due to its minimal hysteresis. Resistance differences between the reference speed and other speeds were then calculated, as shown in Figure 5.5(b). Notably, an anomalous behaviour is evident where the strain was below 5%. Resistance differences near 100% strain for loading were plotted against speed in Figure 5.5(c) to examine the trends.

The experimental results were analysed using a 'power1' fit model in MATLAB. This choice was justified by the expectation that the resistance difference should be zero when the speed is zero. The trend persists even at our reference speed of 10 mm/min , which is practically negligible when compared to the maximum speed of $1,013.1 \text{ mm/min}$ employed in the tests. It is important to note that the sensor remains functional even at speeds lower than the reference speed or when stationary, i.e., speed = 0 mm/min .

The analysis revealed a notable trend: at equivalent applied strain levels, the difference in resistance becomes more noticeable at lower speeds, particularly below 200 mm/min . Beyond this threshold, the difference tends to narrow, exhibiting a semi-linear pattern. Further exploration of the impact of strain rate is discussed in Section 5.3.5.

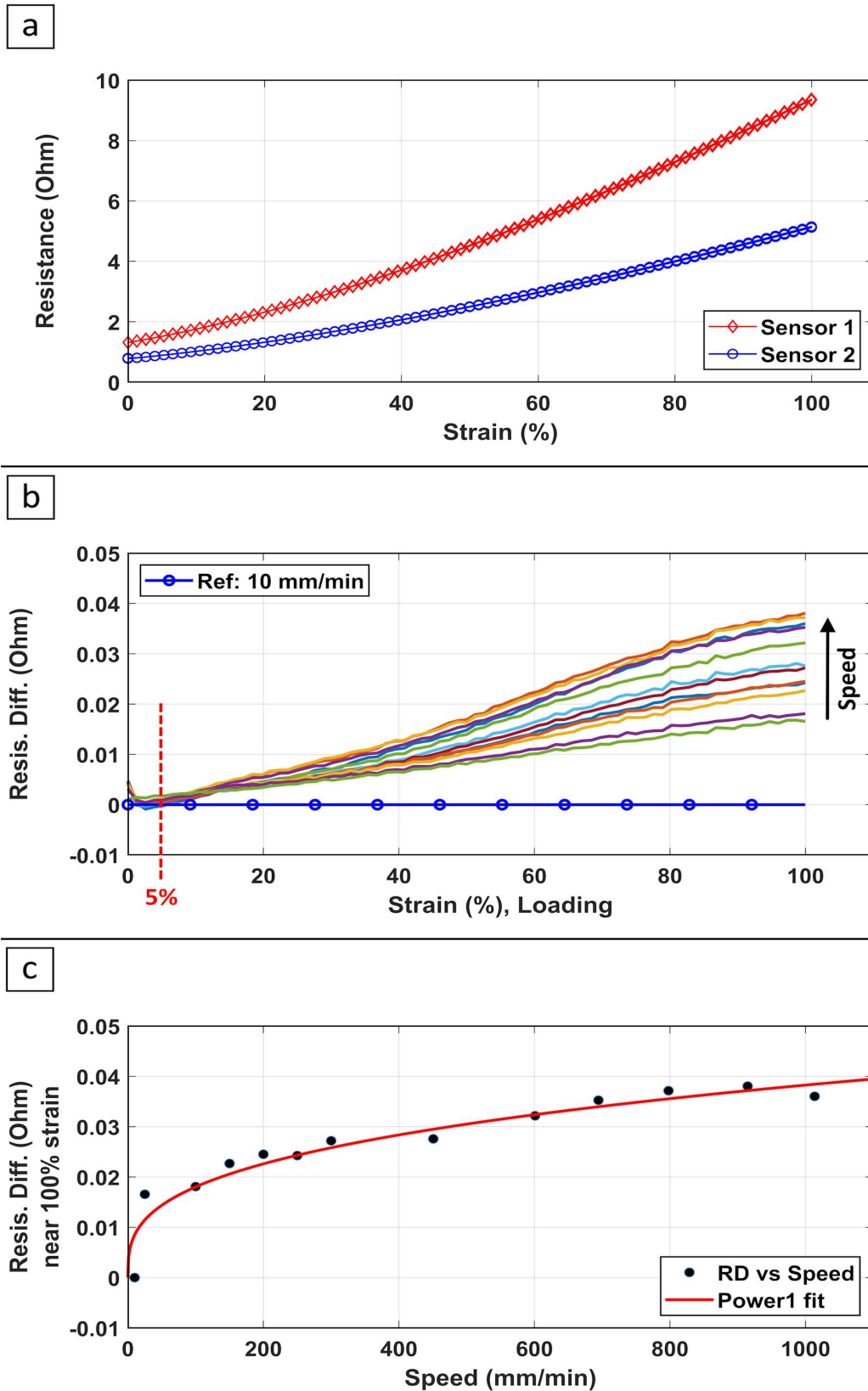


Figure 5.5: Raw experimental resistance measurements R_x : (a) Loading and unloading results for both sensors. (b) Differences in sensor S2 measurements between the reference speed and the other speeds, listed in Table 5.1, show that higher speeds lead to higher differences. The region below 5% strain, marked by a dashed red line, exhibits anomalous behaviour. (c) Resistance difference at near 100% strain with 'Power1' fit model.

5.3.2 Theoretical Model Versus Experimental Results

This subsection undertakes a detailed comparison between the experimental results obtained from sensor S1 and the theoretical predictions derived from the theoretical model (5.2). Sensor S1 was chosen for this comparison due to its consistent testing conditions, with all experimental runs conducted at a fixed speed of 100 mm/min . By maintaining uniform testing parameters, the aim is to isolate the effects of strain distribution on the sensor's performance and gain insights into its behaviour under controlled conditions.

Through this comparative analysis, the objective is to elucidate any discrepancies between the experimental data and theoretical expectations, thereby refining the understanding of the sensor's operational characteristics and informing potential alternatives to the current theoretical model.

To ensure a meaningful comparison between the experimental results of sensor S1 and the theoretical model (5.2), it was essential to align the strain values with the corresponding geometrical length. This comparison was conducted by applying the input strain to the active length L_{cc} .

Subsequently, the lengths of the strained sensing element, L_s , was determined using the relationship provided by (5.6). Utilising L_s , the length of the strained microchannel, L , was computed by leveraging the ratio:

$$\frac{L}{L_o} = \frac{L_s}{L_s^o}. \quad (5.10)$$

After calculating the strain in the deformed length L , the resistance ratio $\Delta R/R_o$ was determined theoretically using (5.2). Meanwhile, experimental measurements from sensor S1 provided the empirical $\Delta R/R_o$. Subsequently, both theoretical and experimental resistance ratios were plotted in Figure 5.6(a). The examination of these results revealed a consistent trend across all the 18 runs of sensor S1, though consistently demonstrating values lower than the theoretical predictions.

To further analyse the discrepancy, the investigation focused on two aspects. Firstly, the theoretical results were examined with the assumption of a constant length for L_e , as proposed by [269], the source of the theoretical model under investigation. This assumption implies that only L_s is strained, which deviates from the actual scenario.

The difference between the theoretical cases, one with a stretched L_e and the other with a constant L_e , and the experimental data were plotted in Figure 5.6(b). The results indicate that when assuming a constant L_e , the discrepancy is notably reduced, although it remains significant, with an error of approximately 4% near maximum strain.

In the sensor design, L_e traverses two different silicon types and incorporates a small portion of connection wires. Consequently, the strain experienced by the liquid metal within L_e is significantly less compared to the rest of L , although not entirely negligible.

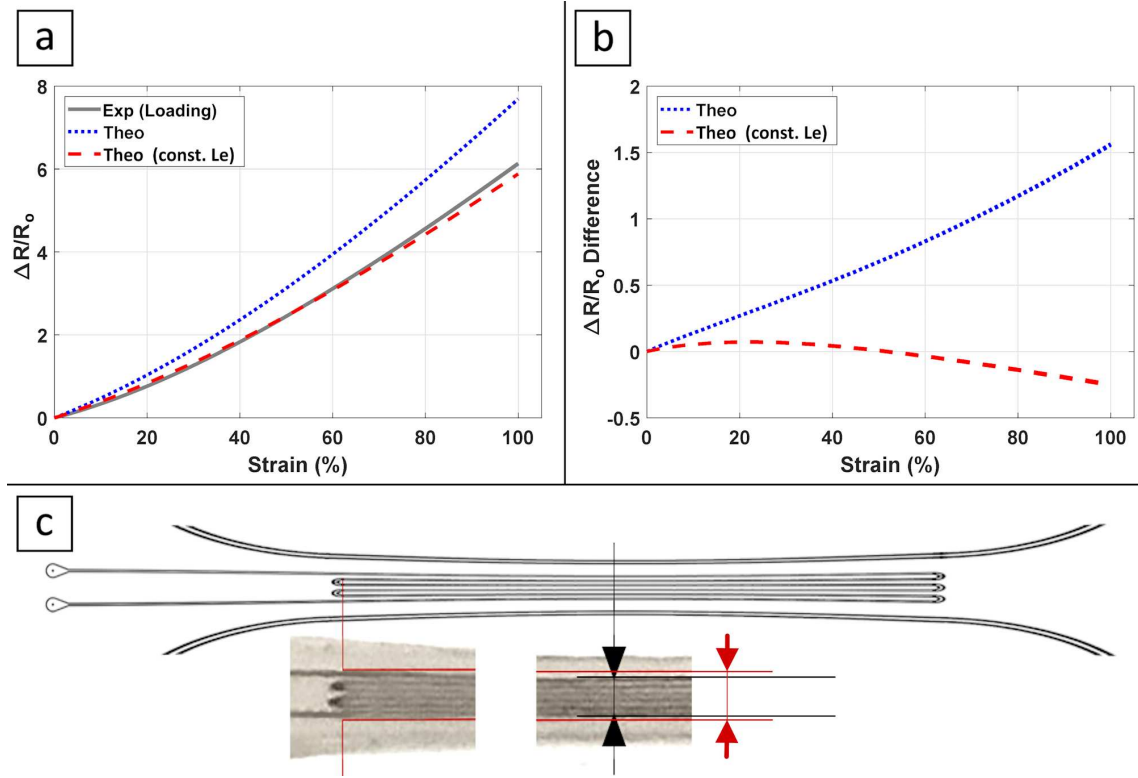


Figure 5.6: Theoretical versus experimental data comparison for sensor S1: (a) Comparison of results based on strain in L_{cc} length, (b) Discrepancy in results for both stretched and constant connection length L_e , and (c) Zoomed section revealing narrowing width at the middle region, resembling a concave rectangular shape. The red marks outline the theoretical width while the black marks outline the narrowed width.

The other aspect relates to the uniformity of microchannel stretching. Theoretical assumptions dictate that the dimensions of the deformed microchannel cross-sectional area A remain uniform across the length of the microchannel at any given strain. This assumption was made by [269] in the experimental characterisation for their choice of sensor design, as well as by [282] in the FEA for their chosen sensor design.

However, the sensor design employed in this study exhibits a distinct behaviour: the sensing element undergoes a narrowing of width in the middle region, resembling a concave rectangular shape, as illustrated in Figure 5.6(c). This observed behaviour potentially contributes to the discrepancy between the theoretical model and the experimental results.

Following an exhaustive examination of the measurements and comparing it with the expected outcomes, it became clear that the sensor design results in a non-uniform distribution of strain. This is in contrast to the theoretical model (5.2), which assumes uniform strain. The observed discrepancy between the experimental observations and theoretical predictions likely arises from the intricate design of the sensor, highlighting the necessity for an alternative model capable of accommodating non-uniform strain distribution.

5.3.3 Numerical Modelling and Underlying Assumptions

In pursuit of a robust numerical model capable of accurately estimating strain based on resistance measurements, various mathematical functions were explored, aiming to capture the intricate relationship between resistance and strain.

The envisioned model needs to estimate strain (or elongation E) based on resistance measurements R_x , following a form expressed by:

$$E = f(R_x). \quad (5.11)$$

Among the suggested models is the rational function [359]. This function offers versatility in accommodating a wide range of data trends, exhibiting good interpolatory properties, excellent extrapolatory capabilities, and the ability to model complex structures with a moderately low degree in both the numerator and denominator [360].

The development of the mathematical model for strain estimation was predicated upon several key assumptions to facilitate analytical tractability and ensure consistency in the interpretation of results:

Firstly, it was assumed that the rational function framework possesses the requisite flexibility to adequately capture the nonlinear relationship between strain and resistance. This assumption is substantiated by the model's robust performance, as demonstrated in the results section, where its fitting accuracy and predictive capability are evaluated against the experimental dataset.

Secondly, the influence of temperature variations on the electrical properties of the sensor was considered negligible, given that all experimental procedures were conducted under controlled ambient conditions, with temperature fluctuations constrained within 2°C. Such minimal variation is considered insufficient to induce significant alterations in the **EGaIn**'s resistive characteristics within the context of this study.

Thirdly, the model assumes that the sensor's behaviour remains stable throughout the duration of the experimental testing, despite repeated loading and unloading cycles. While potential temporal degradation effects such as material fatigue or hysteresis accumulation may occur over extended use, this study focuses on short-term behaviour under controlled laboratory conditions.

Lastly, the analysis was confined to linear deformation modes, with the experimental design explicitly tailored to isolate uniaxial strain, thereby excluding the complexities associated with torsional or multi-axial loading conditions. Collectively, these assumptions delineate the operational boundaries of the model, providing a controlled framework for evaluating strain estimation within the specified experimental parameters.

5.3.4 Validation and Performance Assessment

The rational function is represented by the following general form:

$$rat_{nm}(x) = \frac{p_n x^n + p_{n-1} x^{n-1} + \cdots + p_1 x + p_0}{x^m + q_{m-1} x^{m-1} + \cdots + q_1 x + q_0}, \quad (5.12)$$

where $1 \leq n, m \leq 5$. To explore various complexities, a sequence of eight combinations were selected, each with increasing complexity based on the order and number of fitting parameters of the rational function. These combinations, illustrated in Figure 5.7(a), are labelled from the simplest (rat_{11}) to the most complex (rat_{55}).

As previously outlined in Section 5.2.5, the dataset from sensor S1 was divided, with two-thirds used for modelling and the remaining one-third for validation purposes. The Curve Fitting Toolbox in MATLAB was utilised to derive the fitting parameters of (5.12) and calculate the RMSE using the modelling dataset from sensor S1. Subsequently, the performance of the combinations was assessed using the validation dataset to gauge their efficacy [296].

Examining Figure 5.7(a), it is evident that rat_{32} exhibits the lowest validation RMSE, rendering it a promising candidate for fitting among the alternatives. However, after careful consideration, the decision was made to go with rat_{22} as the optimal model due to its lower complexity. Despite the slightly higher validation RMSE of rat_{22} compared to rat_{33} , the simplicity of rat_{22} outweighs the marginal differences in performance.

The validation errors for both rat_{22} and rat_{32} are illustrated in Figure 5.7(b) and (c) respectively. Both models exhibit an error range of $+0.05 \text{ mm}$ to -0.15 mm in the elongation estimation, indicating a consistent deviation. This error margin is influenced by both the sensor's inherent accuracy and the fitting precision of the model.

Subsequently, the mathematical model (5.11) adopted for the remainder of this chapter takes the following general form:

$$E = f(R_x) = \frac{p_2 R_x^2 + p_1 R_x + p_0}{R_x^2 + q_1 R_x + q_0}. \quad (5.13)$$

The fitting parameters of (5.13) need to be recalculated for each new sensor, which can be achieved through a single test at the reference speed. This model establishes a direct relationship between the measured resistance R_x and the estimated elongation E of the active length L_{cc} . In fact, the model comprises two implicit steps. The first step establishes the relationship between the measured resistance R_x and the elongation of the microchannel length L . The second step determines the relationship between the elongations of the microchannel length L and the active length L_{cc} .

Despite appearing as a direct relation between R_x and E , it is equivalent to performing two manual steps outlined previously in Section 5.3.2 when the theoretical and experimental resistance values were compared.

Furthermore, in comparison to the generalised least squares approach proposed by [284], the selected model 5.13 demonstrated superior accuracy. It is inferred that the model may be suitable for capturing the behaviour of the soft sensor even beyond

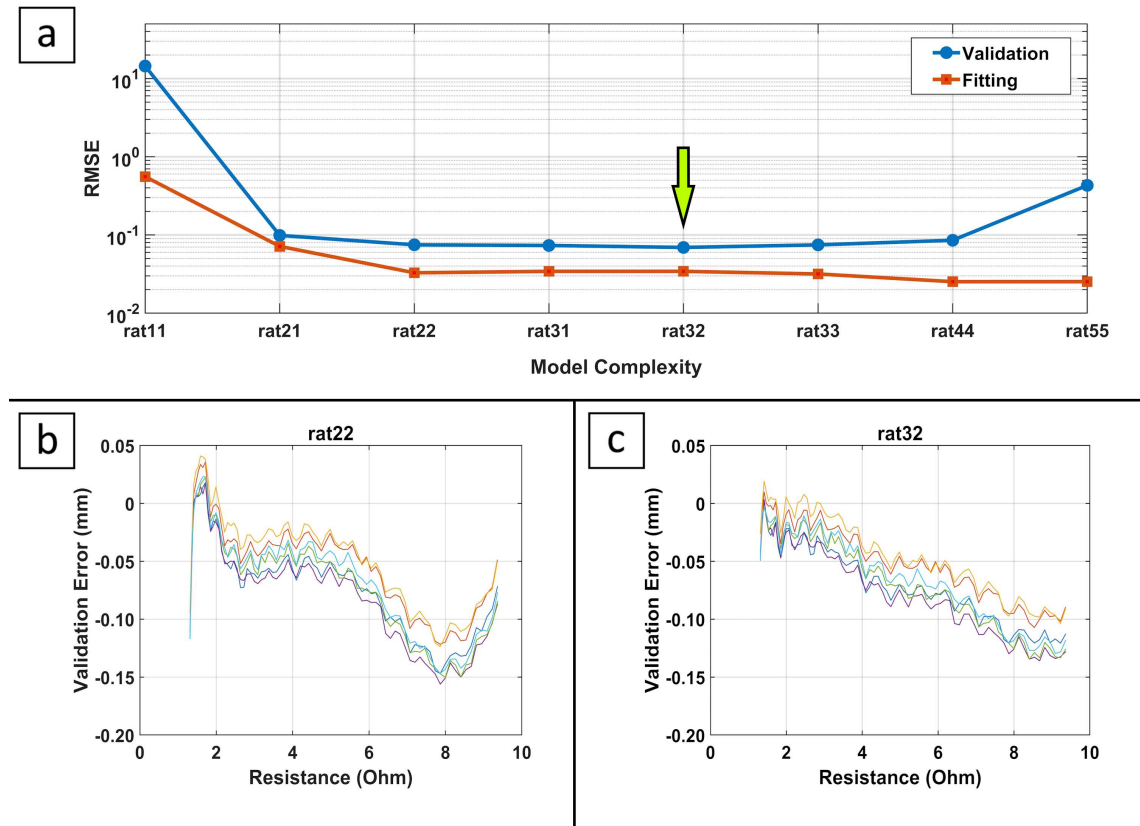


Figure 5.7: Comparison of rational fitting models. (a) RMSE for fitting and validation for each model. The model exhibiting the lowest validation RMSE is marked with a green arrow. (b) Validation errors for rat_{22} and (c) for rat_{32} , respectively, for each run in the validation dataset of sensor S1.

our calibrated range, i.e., strain $> 100\%$, owing to the exceptional extrapolatory capabilities inherent in the rational function.

5.3.5 Strain Rate Influence on Hysteresis

As elucidated in Chapter 2, Section 2.3.1, silicon-based sensors often exhibit viscoelastic behaviour, which manifests as noticeable hysteresis. This phenomenon stems from the material's inherent capability to store and release energy over time, contributing to the sensor's dynamic response.

In dynamic scenarios where real-time feedback is indispensable [246], variations in strain rate can significantly influence hysteresis, thereby imposing limitations on the practical utility of soft sensors in such dynamic environments.

While certain research endeavours concentrate on mitigating or entirely eliminating hysteresis effects [239], other researches are dedicated to the modelling of this phenomenon [305]. The present study focuses on the development of a relatively straightforward yet robust model aimed at characterising hysteresis behaviour. The primary goal is to facilitate its integration into real-time applications, particularly in scenarios where soft sensors are employed for monitoring the state of soft actuators.

The measurements from sensor S2 were utilised to analyse the effect of strain rate

on hysteresis. Figure 5.5(b) illustrates anomalous performance of the sensor for strains below 5%, prompting exclusion of measurements from this region for further analysis, to be investigated in future studies.

To prioritise fitting accuracy over sensing accuracy, the average of three runs for each speed was calculated. Sensor S2 was recalibrated by re-evaluating the fitting parameters of (5.13) using measurements from the lowest speed (Test No.13) as the reference, as previously mentioned in Section 5.3.4. Results from the highest speed (Test No.1) were plotted against strain in Figure 5.8(a), featuring a zoomed-in section highlighting the discrepancy between loading and unloading curves.

The behaviour of the sensor observed through the experimental setup exhibited significantly less hysteresis compared to the results published by the designers [242]. This difference can be attributed to the fact that the measurements were conducted using a testing rig, whereas the designers' measurements were obtained during the analysis of human gaits. Moreover, the upper limit of the selected speed (and strain rate) range in this study is constrained by the mechanical capabilities of the testing rig rather than the intrinsic limitations of the sensor.

The procedure outlined in [242] was followed to calculate the hysteresis percentage, and the results are depicted in Figure 5.8(b). Consistent with the trend observed in the resistance differences illustrated in Figure 5.5, as discussed in Section 5.3.2, it was observed that the hysteresis percentage can be effectively modelled using the MATLAB fit model 'power1', as follows:

$$H = a_2 S^{b_2}, \quad (5.14)$$

where H represents the hysteresis percentage, and S denotes the testing speed. The fitting parameters for the model are determined to be $a_2 = 0.02894$ and $b_2 = 0.3337$, indicating an RMSE of 0.0119 and a high goodness of fit R-squared value of 0.974.

Utilising the 'power1' fitting model instead of 'power2' is logically justified, as the hysteresis percentage should indeed be zero when the speed (or strain rate) equals 0 mm/min . While the 'power2' model may fit the data adequately, it does not inherently ensure this property, unlike the 'power1' model, which better aligns with the expected behaviour of the system. This hysteresis model (5.14) holds promise for future applications, as it can be employed to adjust estimated strain values based on strain rate data.

Table 5.2: Fitting parameters of (5.13) for loading, unloading and combined models at reference speed.

Model	p_1	p_2	p_3	q_1	q_2
Loading	541.7	561.9	-751.9	33.99	14.57
Unloading	539.3	614.8	-789.3	34.1	16.26
Combined	540.4	587.9	-770.3	34.04	15.4

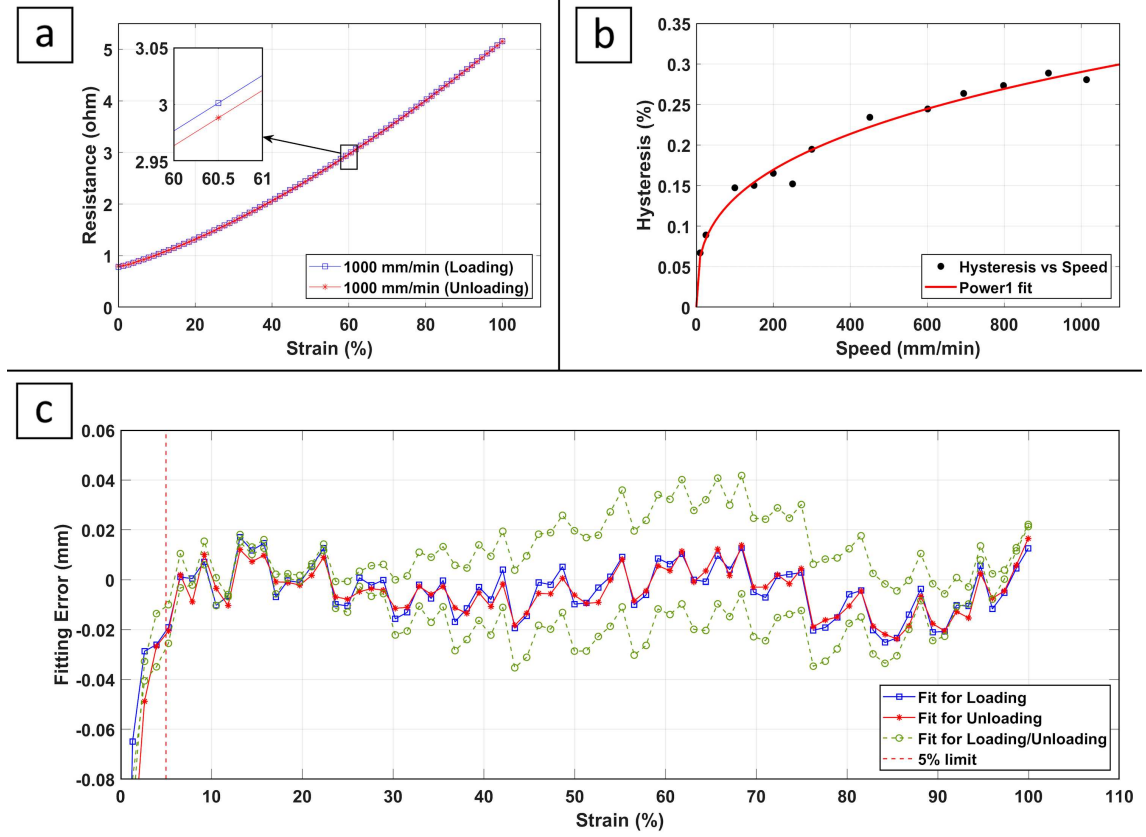


Figure 5.8: Strain rate effect on hysteresis in fitting sensor S2 measurements: (a) Raw measurements during loading and unloading phases at the highest speed (Test No.1), (b) Hysteresis percentage with 'Power1' fit model, and (c) Comparison of fitting with separate models versus a combined model at the reference speed. The dashed red line indicates the region within 5% strain exhibiting the anomalous behaviour.

Given the presence of hysteresis, two distinct approaches may be adopted for calculating the fitting parameters of (5.13): either employing separate models for loading and unloading data, or utilising a single combined model. The fitting parameters for both approaches were computed at the reference speed, Table 5.2, followed by an evaluation of the fitting error. The results are depicted in Figure 5.8(c), where the fitting error is plotted against strain.

The dashed line represents the excluded anomalous data of the 5% region. The continuous blue curve illustrates the error in fitting the measurements from the loading phase only, while the continuous red curve depicts the error for the measurements from the unloading phase only. The dashed green curve represents the fitting errors of a model generated by using both loading and unloading measurements.

The separate models demonstrated a fitting error of approximately ± 0.02 mm each, while the combined model exhibited an error of approximately ± 0.04 mm. However, opting for a combined model offers a straightforward solution if its error margin is deemed acceptable for any intended application.

Conversely, in applications requiring higher accuracy, employing separate models necessitates a dedicated algorithm to determine the direction of the sensor's move-

ment, thereby increasing the overall complexity of strain estimation. The decision between higher accuracy and reduced estimation complexity requires careful consideration based on the specific requirements of each application.

5.3.6 Loading Versus Unloading Direction Determination

Theoretically, a straightforward method to ascertain the sensor's movement, whether it is 'loading', 'unloading', or 'holding position' state, involves examining the sign of the resistance rate. An algorithm can be devised to compute the resistance rate, which essentially represents the mathematical derivative of the real-time resistance measurement with respect to time, assuming the measurements are obtained at regular time intervals.

Based on this, the algorithm can discern the sensor's movement; positive resistance rates indicate 'loading,' negative rates signify 'unloading,' and a rate of zero suggests the sensor is 'holding position'.

However, in practice, the noise of the measurement can introduce inaccuracies into the derivative calculation, potentially generating misleading readings. Through experimentation, it was observed that incorporating a dead zone—neglecting a small range around zero rate—can significantly improve the accuracy of the direction detection algorithm.

Determining the appropriate dead zone range requires finding a balance between acceptable hysteresis percentage and the tolerable impact of noise. Within this dead zone, the resistance rate is treated as zero, indicating a 'holding position' state. This signals the algorithm to transition to a combined fitting model instead of utilising separate models.

Outside the dead zone, the algorithm relies on the sign of the resistance rate to determine whether to use the loading or unloading fitting model. Hence, the dead zone range needs to be set as low as possible, based on the actual noise level, to minimise the impact on the algorithm's ability to detect very small strain rates.

To assess the measurement noise, a solid resistor was connected to the [DAQ](#), and measurements were logged over a one-hour period. The use of a solid resistor instead of the soft sensor enhances the reliability of noise measurement characterisation, ensuring that any changes in measurement are attributed solely to noise rather than strain if the sensor was connected instead.

The choice of the solid resistor's value was informed by the findings depicted in Figure 5.8(c), where the maximum error occurred within the 60 – 70% strain range. Consequently, by cross-referencing with sensor S2 measurements in Figure 5.5(a), a resistor with a nominal resistance of $3.3\ \Omega$ was selected for the test.

An equivalent elongation for the measured noisy resistance was calculated using the combined model depicted in Figure 5.8(c). In other words, the same fitting parameters of the combined model acquired from fitting Sensor S2 data at the reference speed, suitable for the 'holding position' case, were utilised.

The noise in elongation measurement is illustrated in Figure 5.9(a). An error range of about $0.025\ \text{mm}$ in elongation is evident in the entire log, attributable to measure-

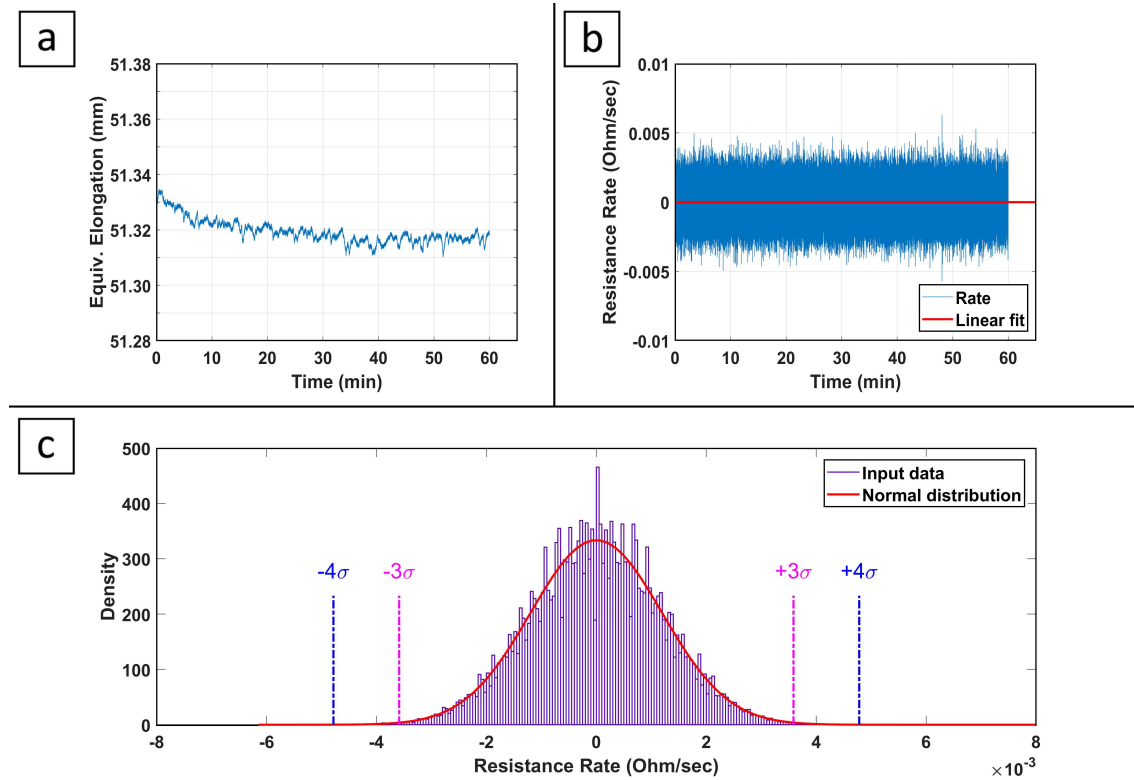


Figure 5.9: Characterisation of measurement noise: (a) Noise in elongation estimation, (b) Resistance rate. The red line represents a linear fit for the resistance rate over the entire logging time, (c) Normal distribution of the resistance rate illustrating the locations corresponding to three and four standard deviations from the mean.

ment noise and a decreasing trend in the first half-hour of logging. The error trend maintained an approximate position for the next half-hour, believed to be related to a small change in room temperature during the test. This logging error is distinct from the fitting error discussed earlier in Section 5.3.4.

Additionally, Figure 5.9(b) illustrates the resistance rate and its normal distribution, as depicted in Figure 5.9(c). An analysis of the rate behaviour revealed that the noise margin exhibits no time-related bias, as indicated by the red linear fit line in Figure 5.9(b).

The analysis suggests three suitable values for the dead zone limits. The first value is $\pm 0.005 \text{ Ohms/sec}$, as depicted in Figure 5.9(b). The second value is $\mu \pm 3\sigma$ (or 99.73%), which corresponds to $\pm 0.0036 \text{ Ohms/sec}$. While the third value is $\mu \pm 4\sigma$ (or 99.99%), which corresponds to $\pm 0.0048 \text{ Ohms/sec}$. The second and third values are illustrated in Figure 5.9(c).

The selection of suitable dead zone limits was based on careful consideration of various factors. The first value, $\pm 0.005 \text{ Ohms/sec}$, was an obvious choice as it is clearly depicted in Figure 5.9(b) and provides a straightforward threshold for identifying noise in the measurement. As for the values derived from statistical analysis, namely $\mu \pm 3\sigma$ (or 99.73%) and $\mu \pm 4\sigma$ (or 99.99%), they offer a robust approach for capturing the majority of noise within a certain range of the mean.

By utilising these values, the algorithm can effectively filter out noise while maintaining sensitivity to meaningful changes in resistance rate. This approach ensures that the dead zone limits strike a balance between minimising the impact of noise and preserving the accuracy of the direction determination process. The choice between the three suggested values for the dead zone limits depends on the specific requirements and constraints of the intended application.

Finally, a Simulink model representing the direction determination algorithm is presented in Figure 5.10, where the resistance rate can provide three states: negative, zero (dead zone), or positive for the 'unloading', 'hold position', or 'loading' states, respectively.

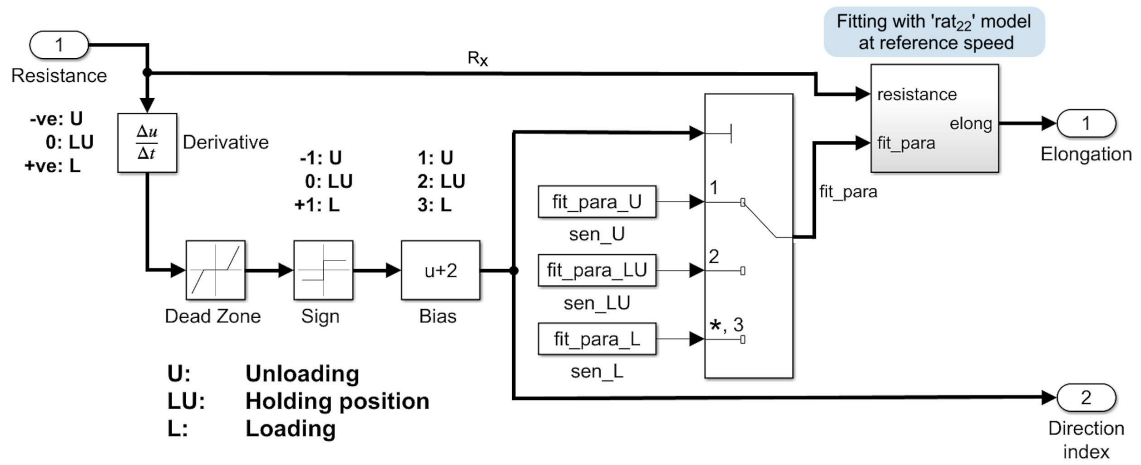


Figure 5.10: Simulink model representing the direction determination algorithm for the sensor's movement.

5.4 Conclusion

This chapter introduces a novel numerical model designed to accurately estimate strain with sub-millimetre precision based on measured resistance, even in scenarios involving non-uniform stretching within microchannels. Through rigorous analysis, the model's ability to provide precise strain estimations under diverse conditions was assessed.

To ensure practical applicability and cost-effectiveness, a specific design proposed by [242] was selected after evaluating various EGaIn-based soft sensor designs available in the literature. Fabricated according to this design, Sensors S1 and S2 were subjected to comprehensive testing and analysis. Sensor S1 underwent a series of six tests with a consistent slider speed of 100 mm/min . This choice of velocity was informed by prior research findings [269, Figure 7f]. In contrast, Sensor S2 underwent thirteen tests, each conducted at varying slider speeds to represent a range of strain rates, spanning nominally from $1,000 \text{ mm/min}$ to 10 mm/min .

A simplified theoretical model, as proposed by [269], characterises the relationship between applied strain and the corresponding change in resistance, and providing a pragmatic framework for analysing sensor behaviour. The examination of experimental data primarily centred on elucidating any discrepancies between the theo-

retical model and measured resistance. It was found that the non-uniform distribution of strain within the microchannel contributed to this discrepancy, underscoring the limitations of the theoretical model in accommodating such variability. Consequently, the necessity emerged for an alternative model capable of accommodating non-uniform strain distributions.

Alternatively, attention shifted towards numerical models, with a particular focus on rational functions, owing to their exceptional capabilities in modelling complex data trends. Eight combinations were meticulously selected for analysis, each chosen to represent increasing levels of complexity. Subsequent to the modelling and validation processes outlined earlier, the validation error associated with each combination was calculated and assessed.

The results indicated that the rational model rat_{22} offered optimal accuracy for strain estimation. While this study does not encompass the characterisation of the precision and accuracy of the chosen sensor, it examines the accuracy of modelling the sensor behaviour. Additionally, a method was introduced to estimate hysteresis percentage as a function of testing speed, serving as a proxy for strain rate.

Given the presence of hysteresis, the benefits of employing separate models for loading and unloading phases to enhance fitting accuracy over a combined model were demonstrated. A novel concept of utilising a dead zone to aid in estimating movement direction was introduced, followed by the development of an algorithm for implementation with the separate models approach. This framework is poised to advance proprioceptive sensing for soft actuator deformation and state estimation.

Collectively, the study presented in this chapter contributes a novel mathematical model capable of accurately estimating strain or elongation based on resistance measurements at a sub-millimetre scale. Additionally, it proposes a solution to mitigate errors induced by hysteresis in strain estimation. Moreover, it furnishes a comprehensive testing methodology, acting as a practical guide for researchers aiming to replicate or build upon the experimental methodologies delineated in this study.

The developed mathematical model exhibits strong predictive accuracy for strain estimation based on resistance measurements, with its generalisability rooted in the empirical nature of the regression approach. Rather than being constrained by specific sensor geometries or manufacturing variations, the model is designed to accommodate different sensor configurations through recalibration. This calibration process ensures that the fitting parameters accurately reflect the unique characteristics of each sensor, regardless of its shape or fabrication details.

However, it's imperative to recognise the limitations inherent in this study. While the model performs effectively under uniaxial strain conditions—as validated in this study—its applicability to more complex deformation modes, such as multi-axial loading or twisting, remains unverified. In multi-axial strain environments, the relationship between strain and resistance may involve complex interactions among multiple resistance and strain components, potentially requiring a multi-dimensional or tensor-based modelling approach.

As such, while the current model provides a robust framework for uniaxial strain estimation, its applicability to multi-axial deformation remains unverified and would

likely necessitate substantial theoretical and experimental extensions. Furthermore, the study was conducted within a temperature range fixed at 24–26°C. Investigating the temperature effect on measurement accuracy becomes crucial when operating the sensor outside this range.

To enhance the testing process, both hardware and software upgrades are needed for the testing rig. These upgrades should include a homing sequence, which would significantly reduce overall testing time by automating the process of resetting the slider to its initial position before each test. This automation ensures result consistency and eliminates the need for manual intervention.

On top of that, establishing consistency in the injected [EGaIn](#) volume during the manufacturing process of each sensor is vital for maintaining quality standards and ensuring reliable performance. Taking a constructive approach to addressing this aspect can enhance the credibility and robustness of the research findings.

Future endeavours will involve additional analysis aimed at refining the direction determination algorithm and improving the reliability of the sensor's movement detection. Addressing the impact of measurement noise on movement detection can be furthered through the implementation of advanced techniques such as noise filtering and adaptive dead zone adjustment mechanisms based on real-time noise levels and sensor performance.

While the rational function model exhibits strong extrapolative properties for strain estimation, it does not explicitly incorporate strain rate effects. Instead, hysteresis was empirically modelled as a function of speed using a power-law relationship, providing a separate means of quantifying its impact. Since the model was developed and validated within the tested speed (and strain rate) range, its accuracy at significantly higher strain rates remains unverified. A potential avenue for future research is to explore whether a unified model that integrates both strain and strain rate dependency can be developed, further enhancing predictive accuracy across a broader range of dynamic conditions.

Moreover, given the inherent measurement uncertainties associated with soft sensors, future efforts will prioritise augmenting sensing accuracy through the adoption of advanced state estimation techniques. Additionally, integrating multiple sensors or sensor modalities will be explored to enhance movement detection accuracy and robustness. Data fusion from diverse sensors can furnish complementary information, thus improving the overall reliability of the direction determination algorithm.

Further improvements to the algorithm's efficacy can be attained by conducting real-world validation and testing across varied operating conditions. This approach will facilitate an assessment of the algorithm's performance under diverse scenarios, validating its effectiveness in practical applications.

Chapter 6

Conclusions and Future Works

The development of SPAs holds significant promise for advancing the field of robotics, offering inherent compliance, manufacturability, and versatility [1]. These actuators are capable of a diverse range of motions, including extension, contraction, bending, twisting, and omnidirectional movement [3]. Nevertheless, compared to their rigid counterparts, SPAs often lack the ability to sense their state accurately in planar or 3D spaces.

The inherent flexibility of SPAs allows for infinite DOF and deformation under both internal and external loads. Yet, predicting the behaviour of these actuators solely through modelling is challenging due to their non-linear and viscoelastic nature [4]. Therefore, enhancing their manipulation and sensing capabilities is essential for improving state estimation accuracy and optimising control systems efficiency.

The findings presented in this thesis are meticulously examined, and their ramifications for the future development of SPAs are thoroughly discussed. Throughout the conclusion chapter, attention is given to addressing the challenges encountered, with forthcoming exploration of potential avenues for further research to overcome these limitations. Through the resolution of these challenges, I contribute to the broader field of soft robotics, facilitating the development of machines that are increasingly capable and adaptable, consequently extending their practical applications across various domains.

6.1 Key Findings, Limitations, and Implications

In this section, the key findings and insights of each study within the thesis are explored, elucidating their contributions to the broader landscape of soft robotics, including a discussion of any encountered limitations.

6.1.1 Inverse Kinematics

The PCC model streamlines the modelling approach for SPAs by segmenting actuators into constant curvature segments defined by rotation angle ϕ_s , bending angle θ_s , and length L_s . These parameters collectively form the configuration vector q_s for each segment. Despite its advantages, the PCC model faces challenges such as

inherent singularities at straight positions, $\theta_s = 0$, and discontinuities in rotation angles, which can disrupt the smooth operation of the [IK](#) solver.

To mitigate these issues, the study proposes a numerical solution by implementing a minimum bending limit where $\theta_s \geq 2 \times 10^{-4}$ to prevent singularities and introduced a two-step correction for Euler angles to minimise discontinuities. These measures proved effective in stabilising the [IK](#) solver and ensuring accurate spatial positioning within the required spatial accuracy ratio *SAR*. In addition, this numerical solution ensured the use of the most common representation, arc parameterisation, instead of the other analytical representations mentioned earlier in Chapter 2, Section 2.1.3.

The derivation of actuator Jacobian $J_f(q)$ and Hessian $H_f(q)$ was crucial in implementing the [IK](#) solver, with a focus on both linear and rotational components, although time constraints limited the full derivation of multi-segment actuator Hessian.

Classification of actuators into non-holonomic, holonomic, or redundant configurations based on segment count and movement constraints provided foundational insights. Two types of constraints—[NCX](#) and [CVX](#)—were explored, with corresponding optimisation Jacobian $J_g(q)$ and Hessian $H_g(q)$ derived for each constraints type. The study’s analysis of selected paths—horizontal line, helical curve, 3D curve, and figure-eight curve—illustrated their distinct complexities and their specific impacts on solver performance.

The evaluation revealed that a quadratic solver based on the exact Hessian generally outperformed both the quadratic solver with an approximated Hessian and the linear solver in terms of convergence speed and stability across different paths. However, the quadratic solver’s efficacy as an approximation of the gradient $\nabla g(q)$ rather than a [CVX](#) function itself necessitated some modifications to enhance the convergence.

These modifications, including the introduction of the Hessian stabilisation factor *hsf* and the replacement technique using the identity matrix to ensure positive definiteness, significantly improved convergence rates. This approach often resulted in optimal outcomes with minimal iterations per path point.

It’s important to recognise that these analyses were performed using simulations. In an experimental setup, a suitable sensing method is required to provide feedback on the actuator configuration. Factors such as measurement noise, state estimation efficiency, and the [PCC](#) assumption’s validity affect the accuracy of this feedback and the [IK](#) solver.

6.1.2 Spiral Sensors Modelling

The study introduces an innovative approach to address the limitations in accurately measuring twist in [SPAs](#). By developing a method to represent spiral sensors within [SPAs](#) using concepts of geodesics and covering spaces, a streamlined alternative to existing models is provided. This approach offers the potential to enhance proprioceptive sensing capabilities in [SPAs](#), enabling more precise control and manipulation in various applications.

In conjunction with the novel approach, a simplified [FEA](#) model for [SPAs](#) is developed. This model incorporates techniques to model reinforcement of pneumatic

muscles and procedures for efficient data extraction using B-splines. By reducing the computational burden of simulations and eliminating the need for modelling contact interactions, while also facilitating the exploration of SPA deformations, this model contributes significantly to the advancement of computational modelling in soft robotics research.

Complementing the FEA model, the implementation of the Kabsch algorithm to compute bending, rotation, and twist angles from FEA model data establishes a comprehensive simulation methodology. This methodology provides researchers with a valuable tool to analyse SPA deformations in depth and optimise simulation efforts. The developed codes further enhance accessibility and utility, facilitating efficient data extraction to MATLAB for further analysis.

Validation results demonstrate the accuracy of the proposed length model in representing both straight and spiral sensors across various testing scenarios. The low error in length measurements validates the effectiveness of the model in capturing sensor deformations accurately. This finding reinforces the potential of the length model to improve proprioceptive sensing in SPAs and enhance their performance in real-world applications.

However, it's important to acknowledge that the accuracy of the length model (4.18) relies on the adherence to the PCC assumption. Higher twisting moments may induce increased arching, potentially rendering the PCC model unsuitable for certain applications. This limitation underscores the need for careful consideration of the model's applicability in specific scenarios, particularly those involving significant twisting forces.

The models and methodologies developed in this study carry substantial implications for the advancement of soft robotics research. They offer opportunities to enhance proprioceptive sensing capabilities in SPAs, paving the way for improved control and manipulation across a range of applications.

6.1.3 EGaIn-Based Strain Sensors

This study introduces a novel numerical model designed to accurately estimate strain with sub-millimetre precision based on measured resistance, even in scenarios involving non-uniform stretching within microchannels. Through rigorous analysis, the model's ability to provide precise strain estimations under varied testing speeds was assessed.

A comprehensive testing methodology was provided, serving as a practical guide for researchers seeking to replicate or build upon the experimental methodologies outlined in this study. The examination of experimental data primarily centred on elucidating any discrepancies between the theoretical model, proposed by [269], and measured resistance.

The discrepancy was attributed to the non-uniform strain distribution within the microchannel, thus emphasising the theoretical model's limitations in accommodating such variability. Consequently, the necessity emerged for an alternative model capable of accommodating non-uniform strain distributions.

Further investigation led to the finding that the rational model rat_{22} provided opti-

mal accuracy for strain estimation, demonstrating its efficacy in modelling complex data trends. Moreover, the study included investigating the effect of hysteresis on modelling accuracy. Given the presence of hysteresis, the benefits of employing separate models for loading and unloading phases to enhance fitting accuracy over a combined model were demonstrated.

A novel concept of utilising a dead zone to aid in estimating movement direction was introduced, followed by the development of an algorithm for implementation with the separate models approach. This framework holds significant implications for the advancement of proprioceptive sensing for soft actuator deformation and state estimation.

The study has notable limitations. It solely examines uniaxial strain along the soft sensor's active length, overlooking lateral or perpendicular strains. Additionally, testing was constrained to a temperature range of $24 - 26^{\circ}\text{C}$, necessitating further exploration of temperature effects beyond this range.

Moreover, the testing process could benefit from upgrades to the hardware and software of the rig, particularly by implementing a homing sequence. This automation would streamline testing, reducing overall time and manual intervention of resetting the slider to its initial position before each test.

6.2 Future works

In this section, potential future research directions are outlined, suggesting areas for exploration and development within soft robotics, building upon the findings of each study within the thesis.

6.2.1 Inverse Kinematics

Future research directions should prioritise extending the derivation of multi-segment actuator Hessian to fully explore its implications for advanced [IK](#) solvers. Investigating the nuanced relationships between damping profiles and Jacobian algorithms under diverse path complexities will enhance understanding the [IK](#) solver performance. Moreover, enhancing non-convex quadratic solvers with convex optimisation techniques like linearisation and relaxation holds potential to enhance solver efficiency and stability in practical applications.

Exploration into enhanced manipulability indices and their application in predicting actuator performance near singularities remains a critical area of study. This includes developing robust methodologies for assessing and mitigating actuator vulnerabilities during complex trajectory executions. Furthermore, integrating real-time feedback mechanisms and adaptive control strategies will be essential for enhancing the reliability and adaptability of [IK](#) solvers in dynamic operational environments.

Addressing these research avenues not only expands the theoretical foundations of actuator control and manipulation but also contributes to practical advancements in robotics and automation. By bridging theoretical insights with practical applications, future studies can propel the field toward more sophisticated, efficient, and adaptable soft robotic systems.

6.2.2 Spiral Sensors Modelling

Future research directions may include further refinement of the length and [FEA](#) models to address specific limitations or enhance methodologies proposed in this study. Further investigations are required to explore the arching phenomena and propose enhancements to improve the accuracy of the length model ([4.18](#)).

Additionally, integrating the length model ([4.18](#)) into state estimation algorithms could facilitate the estimation of the [SPA](#)'s 3D state and twisting moment, even when the latter is unknown [[352](#)]. This integration has the potential to extend state estimation by incorporating experimental sensing measurements [[270](#)], thereby improving the overall accuracy and reliability of the system. Exploring real-time control applications and conducting additional experimental validation will further contribute to the advancement of knowledge within the field of soft robotics.

6.2.3 EGaIn-Based Strain Sensors

Future research efforts will focus on refining the direction determination algorithm and improving the reliability of the sensor's movement detection. Addressing the impact of measurement noise on movement detection through the implementation of advanced techniques such as noise filtering and adaptive dead zone adjustment mechanisms will be a priority.

The rational function model provides a flexible framework for strain estimation but does not explicitly incorporate strain rate effects. Hysteresis was independently characterised as a function of speed using a power-law model, highlighting the need for future work to develop an integrated approach.

Furthermore, exploring advanced state estimation techniques and the integration of multiple sensors or sensor modalities will be pursued to enhance sensing accuracy and robustness. Real-world validation and testing across varied operating conditions is intended to assess the algorithm's performance and validate its effectiveness in practical applications.

Bibliography

- [1] H. A. Sonar and J. Paik. “Soft Pneumatic Actuator Skin with Piezoelectric Sensors for Vibrotactile Feedback”. In: *Frontiers in Robotics and AI* 2 (2016).
- [2] E. H. Skorina and C. D. Onal. “Soft Pneumatic Actuators: Modeling, Control, and Application”. In: *Smart Materials: Considerations on Earth and in Space*. Cham: Springer International Publishing, 2022, pp. 129–219.
- [3] M. A. Robertson et al. “Soft Pneumatic Actuator Fascicles for High Force and Reliability”. In: *Soft Robotics* 4.1 (2017), pp. 23–32.
- [4] P. Polygerinos et al. “Soft Robotics: Review of Fluid-Driven Intrinsically Soft Devices; Manufacturing, Sensing, Control, and Applications in Human-Robot Interaction”. In: *Advanced Engineering Materials* 19.12 (2017), p. 1700016.
- [5] M. S. Xavier et al. “Soft Pneumatic Actuators: A Review of Design, Fabrication, Modeling, Sensing, Control and Applications”. In: *IEEE Access* 10 (2022), pp. 59442–59485.
- [6] D. Trivedi et al. “Soft robotics: Biological inspiration, state of the art, and future research”. In: *Applied Bionics and Biomechanics* 5.3 (2008), pp. 99–117.
- [7] R. F. Shepherd et al. “Multigait soft robot”. In: *Proceedings of the National Academy of Sciences* 108.51 (2011), pp. 20400–20403.
- [8] C. Rad, O. Hancu, and C. Lapusan. “Data-Driven Kinematic Model of PneuNets Bending Actuators for Soft Grasping Tasks”. In: *Actuators* 11.2 (2022), p. 58.
- [9] G. Mengaldo et al. “A concise guide to modelling the physics of embodied intelligence in soft robotics”. In: *Nature Reviews Physics* 4.9 (2022), pp. 595–610.
- [10] R. Pfeifer, M. Lungarella, and F. Iida. “Self-organization, embodiment, and biologically inspired robotics”. In: *science* 318.5853 (2007), pp. 1088–1093.
- [11] Z. Zhao et al. “Exploring embodied intelligence in soft robotics: a review”. In: *Biomimetics* 9.4 (2024), p. 248.
- [12] Y.-L. Park et al. “Networked bio-inspired modules for sensorimotor control of wearable cyber-physical devices”. In: *International Conference on Computing, Networking and Communications (ICNC)*. 2013, pp. 92–96.
- [13] G. Schillaci. “Sensorimotor learning and simulation of experience as a basis for the development of cognition in robotics”. Thesis. 2014.
- [14] J. Walker et al. “Soft Robotics: A Review of Recent Developments of Pneumatic Soft Actuators”. In: *Actuators* 9.1 (2020), p. 3.
- [15] M. Li et al. “Soft actuators for real-world applications”. In: *Nature Reviews Materials* 7.3 (2022), pp. 235–249.

- [16] D. C. Rucker and I. R. J. Webster. “Statics and Dynamics of Continuum Robots With General Tendon Routing and External Loading”. In: *IEEE Transactions on Robotics* 27.6 (2011), pp. 1033–1044.
- [17] R. J. Webster and B. A. Jones. “Design and Kinematic Modeling of Constant Curvature Continuum Robots: A Review”. In: *The International Journal of Robotics Research* 29.13 (2010), pp. 1661–1683.
- [18] M. W. Hannan and I. D. Walker. “Kinematics and the implementation of an elephant’s trunk manipulator and other continuum style robots”. In: *Journal of Robotic Systems* 20.2 (2003), pp. 45–63.
- [19] B. A. Jones and I. D. Walker. “Kinematics for multisection continuum robots”. In: *IEEE Transactions on Robotics* 22.1 (2006), pp. 43–55.
- [20] M. Pourafzal, H. A. Talebi, and K. Rabenoroso. “Piecewise constant strain kinematic model of externally loaded concentric tube robots”. In: *Mechatronics* 74 (2021), p. 102502.
- [21] C. Della Santina, A. Bicchi, and D. Rus. “On an Improved State Parametrization for Soft Robots With Piecewise Constant Curvature and Its Use in Model Based Control”. In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 1001–1008.
- [22] P. Schegg and C. Duriez. “Review on generic methods for mechanical modeling, simulation and control of soft robots”. In: *PLOS ONE* 17.1 (2022), e0251059.
- [23] T. George Thuruthel, F. Renda, and F. Iida. “First-Order Dynamic Modeling and Control of Soft Robots”. In: *Frontiers in Robotics and AI* 7 (2020).
- [24] J. L. Hicks et al. “Is my model good enough? Best practices for verification and validation of musculoskeletal models and simulations of movement”. In: *J Biomech Eng* 137.2 (2015), p. 020905.
- [25] F. P. Connolly. “Mathematical Models for Guiding Pneumatic Soft Actuator Design”. Dissertation. Harvard University, 2018.
- [26] R. Mengacci et al. “An Open-Source ROS-Gazebo Toolbox for Simulating Robots With Compliant Actuators”. In: *Frontiers in Robotics and AI* 8 (2021).
- [27] T. Hassan et al. “Finite-Element Modeling and Design of a Pneumatic Braided Muscle Actuator With Multifunctional Capabilities”. In: *IEEE/ASME Transactions on Mechatronics* 24.1 (2019), pp. 109–119.
- [28] R. L. Truby, C. Della Santina, and D. Rus. “Distributed Proprioception of 3D Configuration in Soft, Sensorized Robots via Deep Learning”. In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 3299–3306.
- [29] Y. X. Mak et al. “Adaptive control of a soft pneumatic actuator using experimental characterization data”. In: *Frontiers in Robotics and AI* 10 (2023).
- [30] W. Xiao et al. “A New Type of Soft Pneumatic Torsional Actuator With Helical Chambers for Flexible Machines”. In: *Journal of Mechanisms and Robotics* 13.1 (2020).
- [31] F. Chen et al. “Soft Twisting Pneumatic Actuators Enabled by Freeform Surface Design”. In: *IEEE Robotics and Automation Letters* 6.3 (2021), pp. 5253–5260.
- [32] G. Chen et al. “Design and Test of an Active Pneumatic Soft Wrist for Soft Grippers”. In: *Actuators* 11.11 (2022), p. 311.

- [33] D. Melancon et al. “Inflatable Origami: Multimodal Deformation via Multi-stability”. In: *Advanced Functional Materials* 32.35 (2022), p. 2201891.
- [34] G. Chen et al. “Design of an Active Flexible Spine for Wall Climbing Robot Using Pneumatic Soft Actuators”. In: *Journal of Bionic Engineering* 20.2 (2023), pp. 530–542.
- [35] H. Wang, M. Totaro, and L. Beccai. “Toward Perceptive Soft Robots: Progress and Challenges”. In: *Advanced Science* 5.9 (2018), p. 1800541.
- [36] K. C. Galloway et al. “Fiber Optic Shape Sensing for Soft Robotics”. In: *Soft Robotics* 6.5 (2019), pp. 671–684.
- [37] A. Gao et al. “Laser-Profiled Continuum Robot with Integrated Tension Sensing for Simultaneous Shape and Tip Force Estimation”. In: *Soft Robotics* 7.4 (2020), pp. 421–443.
- [38] O. Erin et al. “Design of a bio-inspired pneumatic artificial muscle with self-contained sensing”. In: *38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. 2016, pp. 2115–2119.
- [39] W. Felt et al. “An Inductance-Based Sensing System for Bellows-Driven Continuum Joints in Soft Robots”. In: *Proceedings of Robotics: Science and Systems (RSS)*. 2017.
- [40] T. Pinto et al. “CNT-based sensor arrays for local strain measurements in soft pneumatic actuators”. In: *International Journal of Intelligent Robotics and Applications* 1.2 (2017), pp. 157–166.
- [41] J. Tapia et al. “MakeSense: Automated Sensor Design for Proprioceptive Soft Robots”. In: *Soft Robotics* (2019).
- [42] N. Farrow, L. McIntire, and N. Correll. “Functionalized textiles for interactive soft robotics”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2017, pp. 5525–5531.
- [43] K. Lynch and F. Park. *Modern Robotics: Mechanics, Planning, and Control*. Cambridge University Press, 2017.
- [44] R. P. Paul. *Robot manipulators: mathematics, programming, and control: the computer control of robot manipulators*. MIT Press, 1981.
- [45] J. M. McCarthy and G. S. Soh. *Geometric design of linkages*. Vol. 11. Springer Science & Business Media, 2010.
- [46] A. Martín, A. Barrientos, and J. del Cerro. “The Natural-CCD Algorithm, a Novel Method to Solve the Inverse Kinematics of Hyper-redundant and Soft Robots”. In: *Soft Robotics* 5.3 (2018), pp. 242–257.
- [47] “Control Strategies for Soft Robotic Manipulators: A Survey”. In: *Soft Robotics* 5.2 (2018), pp. 149–163.
- [48] O. Robotics. *Snake-arm Robots*. URL: <https://www.ocrobotics.com/>. Accessed 17 Oct 2023.
- [49] T. F. Group. *Bionic Handling Assistant*. URL: https://www.festo.com/au/en/e/about-festo/research-and-development/bionic-learning-network/highlights-from-2010-to-2012/bionic-handling-assistant-id_33759/. Accessed 17 Oct 2023.
- [50] G. S. Chirikjian and J. W. Burdick. “A hyper-redundant manipulator”. In: *IEEE Robotics & Automation Magazine* 1.4 (1994), pp. 22–29.
- [51] S. Kolachalama and S. Lakshmanan. “Continuum Robots for Manipulation Applications: A Survey”. In: *Journal of Robotics* 2020 (2020), p. 4187048.

- [52] G. Bao et al. “Soft Robotics: Academic Insights and Perspectives Through Bibliometric Analysis”. In: *Soft Robotics* 5.3 (2018), pp. 229–241.
- [53] P. S. Gonthina et al. “Modeling Variable Curvature Parallel Continuum Robots Using Euler Curves”. In: *2019 International Conference on Robotics and Automation (ICRA)*, pp. 1679–1685.
- [54] I. Singh et al. “Modeling of Continuum Manipulators Using Pythagorean Hodograph Curves”. In: *Soft Robotics* 5.4 (2018), pp. 425–442.
- [55] X. Huang, J. Zou, and G. Gu. “Kinematic Modeling and Control of Variable Curvature Soft Continuum Robots”. In: *IEEE/ASME Transactions on Mechatronics* 26.6 (2021), pp. 3175–3185.
- [56] I. S. Godage and I. D. Walker. “Dual Quaternion based modal kinematics for multisection continuum arms”. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. 2015, pp. 1416–1422.
- [57] I. S. Godage et al. “Novel modal approach for kinematics of multisection continuum arms”. In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1093–1098.
- [58] I. S. Godage et al. “Modal kinematics for multisection continuum arms”. In: *Bioinspiration & Biomimetics* 10.3 (2015), p. 035002.
- [59] I. S. Godage, R. J. Webster, and I. D. Walker. “Center-of-Gravity-Based Approach for Modeling Dynamics of Multisection Continuum Arms”. In: *IEEE Transactions on Robotics* 35.5 (2019), pp. 1097–1108.
- [60] M. Ivanescu et al. “Coil function control problem for a hyperredundant robot”. In: *2007 IEEE/ASME international conference on advanced intelligent mechatronics*, pp. 1–6.
- [61] P. Sears and P. Dupont. “A Steerable Needle Technology Using Curved Concentric Tubes”. In: *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2850–2856.
- [62] R. J. Webster, J. M. Romano, and N. J. Cowan. “Mechanics of Precurved-Tube Continuum Robots”. In: *IEEE Transactions on Robotics* 25.1 (2009), pp. 67–78.
- [63] A. Chawla, C. Frazelle, and I. Walker. “A Comparison of Constant Curvature Forward Kinematics for Multisection Continuum Manipulators”. In: *2018 Second IEEE International Conference on Robotic Computing (IRC)*, pp. 217–223.
- [64] F. Renda et al. “A Geometric Variable-Strain Approach for Static Modeling of Soft Manipulators With Tendon and Fluidic Actuation”. In: *IEEE Robotics and Automation Letters* 5.3 (2020), pp. 4006–4013.
- [65] F. Renda et al. “Discrete Cosserat Approach for Multisection Soft Manipulator Dynamics”. In: *IEEE Transactions on Robotics* 34.6 (2018), pp. 1518–1533.
- [66] F. Renda et al. “Screw-Based Modeling of Soft Manipulators With Tendon and Fluidic Actuation”. In: *Journal of Mechanisms and Robotics* 9.4 (2017).
- [67] M. Russo et al. “Continuum Robots: An Overview”. In: *Advanced Intelligent Systems* 5.5 (2023), p. 2200367.
- [68] C. Armanini et al. “Soft Robots Modeling: A Structured Overview”. In: *IEEE Transactions on Robotics* 39.3 (2023), pp. 1728–1748.
- [69] H. Cheng et al. “Approximate piecewise constant curvature equivalent model and their application to continuum robot configuration estimation”. In: *IEEE*

- International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2020, pp. 1929–1936.
- [70] H. Cheng et al. “Orientation to Pose: Continuum Robots Shape Reconstruction Based on the Multi-Attitude Solving Approach”. In: *International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 3203–3209.
 - [71] T. M. Bieze et al. “Finite Element Method-Based Kinematics and Closed-Loop Control of Soft, Continuum Manipulators”. In: *Soft Robotics* 5.3 (2018), pp. 348–364.
 - [72] K. Chin, T. Hellebrekers, and C. Majidi. “Machine Learning for Soft Robotic Sensing and Control”. In: *Advanced Intelligent Systems* 2.6 (2020), p. 1900171.
 - [73] M. Gazzola et al. “Forward and inverse problems in the mechanics of soft filaments”. In: *Royal Society open science* 5.6 (2018), p. 171628.
 - [74] A. Al-Azzawi et al. “Length Modelling of Spiral Superficial Soft Strain Sensors Using Geodesics and Covering Spaces”. In: *Robotics* 12.6 (2023). 164.
 - [75] R. K. Katzschnmann et al. “Dynamic Motion Control of Multi-Segment Soft Robots Using Piecewise Constant Curvature Matched with an Augmented Rigid Body Model”. In: *2nd IEEE International Conference on Soft Robotics (RoboSoft)*. 2019, pp. 454–461.
 - [76] A. Garriga-Casanovas and F. Rodriguez y Baena. “Kinematics of Continuum Robots With Constant Curvature Bending and Extension Capabilities”. In: *Journal of Mechanisms and Robotics* 11.1 (2019).
 - [77] T. F. Allen et al. “Closed-Form Non-Singular Constant-Curvature Continuum Manipulator Kinematics”. In: *3rd IEEE International Conference on Soft Robotics (RoboSoft)*. 2020, pp. 410–416.
 - [78] C. Della Santina and D. Rus. “Control Oriented Modeling of Soft Robots: The Polynomial Curvature Case”. In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 290–298.
 - [79] P. Preechayasomboon and E. Rombokas. “Sensuator: A Hybrid Sensor–Actuator Approach to Soft Robotic Proprioception Using Recurrent Neural Networks”. In: *Actuators* 10.2 (2021), p. 30.
 - [80] M. Keyvanara, A. Goshtasbi, and I. A. Kuling. “A Geometric Approach towards Inverse Kinematics of Soft Extensible Pneumatic Actuators Intended for Trajectory Tracking”. In: *Sensors* 23.15 (2023), p. 6882.
 - [81] Z. Gong et al. “An inverse kinematics method of a soft robotic arm with three-dimensional locomotion for underwater manipulation”. In: *2018 IEEE International Conference on Soft Robotics (RoboSoft)*, pp. 516–521.
 - [82] D. Y. Kolpashchikov et al. “FABRIK-Based Inverse Kinematics For Multi-Section Continuum Robots”. In: *2018 18th International Conference on Mechatronics - Mechatronika (ME)*, pp. 1–8.
 - [83] T. G. Thuruthel et al. “Model-Based Reinforcement Learning for Closed-Loop Dynamic Control of Soft Robotic Manipulators”. In: *IEEE Transactions on Robotics* 35.1 (2019), pp. 124–134.
 - [84] G. Fang et al. “Efficient Jacobian-Based Inverse Kinematics With Sim-to-Real Transfer of Soft Robots by Learning”. In: *IEEE/ASME Transactions on Mechatronics* (2022), pp. 1–11.
 - [85] A. Melingui et al. “Adaptive Neural Network Control of a Compact Bionic Handling Arm”. In: *IEEE/ASME Transactions on Mechatronics* 20.6 (2015), pp. 2862–2875.

- [86] M. Li et al. “Model-Free Control for Continuum Robots Based on an Adaptive Kalman Filter”. In: *IEEE/ASME Transactions on Mechatronics* 23.1 (2018), pp. 286–297.
- [87] P. H. Nguyen et al. “Design and control of a 3-chambered fiber reinforced soft actuator with off-the-shelf stretch sensors”. In: *International Journal of Intelligent Robotics and Applications* 1.3 (2017), pp. 342–351.
- [88] D. Kim et al. “Review of machine learning methods in soft robotics”. In: *PLOS ONE* 16.2 (2021), e0246102.
- [89] R. F. Reinhart and J. J. Steil. “Hybrid Mechanical and Data-driven Modeling Improves Inverse Kinematic Control of a Soft Robot”. In: *Procedia Technology* 26 (2016), pp. 12–19.
- [90] Z. Q. Tang et al. “Model-based online learning and adaptive control for a “human-wearable soft robot” integrated system”. In: *The International Journal of Robotics Research* 40.1 (2021), pp. 256–276.
- [91] G. Lou et al. “Controlling Soft Robotic Arms Using Hybrid Modelling and Reinforcement Learning”. In: *IEEE Robotics and Automation Letters* (2024).
- [92] Z. Chen et al. “Data-driven methods applied to soft robot modeling and control: A review”. In: *IEEE Transactions on Automation Science and Engineering* (2024).
- [93] M. Su and C. Hsueh. “An SOM-Like Approach to Inverse Kinematics Modeling”. In: *Int J Comput Softw Eng* 2.112 (2017), p. 2.
- [94] N. Jorge and J. W. Stephen. *Numerical optimization*. 2nd ed. Springer Series in Operations Research and Financial Engineering. New York, NY: Springer, 2006.
- [95] S. Ruder. “An overview of gradient descent optimization algorithms”. In: *arXiv preprint arXiv:1609.04747* (2016).
- [96] M. J. Kochenderfer and T. A. Wheeler. *Algorithms for optimization*. Mit Press, 2019.
- [97] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.
- [98] P. Corke. *Robotics, Vision and Control: Fundamental Algorithms In MATLAB® Second, Completely Revised, Extended And Updated Edition*. 2nd ed. Vol. 118. Springer Tracts in Advanced Robotics. Springer Cham, 2017, p. 693.
- [99] S. R. Buss. *Introduction to Inverse Kinematics with Jacobian Transpose, Pseudoinverse and Damped Least Squares methods*. Report. Department of Mathematics, University of California, San Diego, Oct. 2009.
- [100] A. Aristidou et al. “Inverse kinematics techniques in computer graphics: A survey”. In: *Computer graphics forum*. Vol. 37. Wiley Online Library, 2018, pp. 35–58.
- [101] C. Welman. “Inverse kinematics and geometric constraints for articulated figure manipulation”. Master Thesis. 1993.
- [102] P. Baerlocher and R. Boulic. “An inverse kinematics architecture enforcing an arbitrary number of strict priority levels”. In: *The Visual Computer* 20.6 (2004), pp. 402–417.
- [103] C. Kyong-Sok and O. Khatib. “Manipulator control at kinematic singularities: a dynamically consistent strategy”. In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*. Vol. 3. 1995, pp. 84–88.

- [104] D. Di Vito, C. Natale, and G. Antonelli. “A Comparison of Damped Least Squares Algorithms for Inverse Kinematics of Robot Manipulators”. In: *IFAC-PapersOnLine* 50.1 (2017), pp. 6869–6874.
- [105] S. R. Buss and J.-S. Kim. “Selectively Damped Least Squares for Inverse Kinematics”. In: *Journal of Graphics Tools* 10.3 (2005), pp. 37–49.
- [106] P. Harish et al. “Parallel inverse kinematics for multithreaded architectures”. In: *ACM Transactions on Graphics (TOG)* 35.2 (2016), pp. 1–13.
- [107] Y. Liu et al. “Evolutionary multi-objective trajectory optimization for a redundant robot in Cartesian space considering obstacle avoidance”. In: *Mech. Sci.* 13.1 (2022), pp. 41–53.
- [108] P. Caserman, P. Achenbach, and S. Göbel. “Analysis of inverse kinematics solutions for full-body reconstruction in virtual reality”. In: *IEEE 7th International Conference on Serious Games and Applications for Health (SeGAH)*. IEEE, pp. 1–8.
- [109] Y. Nakamura and H. Hanafusa. “Inverse Kinematic Solutions With Singularity Robustness for Robot Manipulator Control”. In: *Journal of Dynamic Systems, Measurement, and Control* 108.3 (1986), pp. 163–171.
- [110] N. Ramezani Taghiabadi. “Inverse kinematics, kinematic control and redundancy resolution for chained-link robotic manipulators”. Thesis. 2016.
- [111] L. M. Phuoc et al. “Damped least square based genetic algorithm with Gaussian distribution of damping factor for singularity-robust inverse kinematics”. In: *Journal of Mechanical Science and Technology* 22.7 (2008), pp. 1330–1338.
- [112] X. Wang et al. “Two Optimized General Methods for Inverse Kinematics of 6R Robots Based on Machine Learning”. In: *Mathematical Problems in Engineering* (2020), Article ID 8174924.
- [113] J. Stewart. *Calculus 6th Edition*. Cengage Learning EMEA, 2008, p. 1310.
- [114] N. Doikov, E. M. Chayti, and M. Jaggi. “Second-Order Optimization with Lazy Hessians”. In: *Proceedings of the 40th International Conference on Machine Learning*. Vol. 202. PMLR, pp. 8138–8161.
- [115] J. Gill and G. King. “Numerical Issues Involved in Inverting Hessian Matrices”. In: *Numerical Issues in Statistical Computing for the Social Scientist*. 2003, pp. 143–176.
- [116] A. Aristidou and J. Lasenby. *Inverse kinematics: a review of existing techniques and introduction of a new fast iterative solver*. Technical Report. Cambridge University Engineering Department, 2009.
- [117] N. Ye, F. Roosta-Khorasani, and T. Cui. “Optimization Methods for Inverse Problems”. In: *2017 MATRIX Annals*. Ed. by J. de Gier, C. E. Praeger, and T. Tao. Cham: Springer International Publishing, 2019, pp. 121–140.
- [118] S. Lloyd, R. A. Irani, and M. Ahmadi. “Fast and Robust Inverse Kinematics of Serial Robots Using Halley’s Method”. In: *IEEE Transactions on Robotics* 38.5 (2022), pp. 2768–2780.
- [119] K. Erleben and S. Andrews. “Solving inverse kinematics using exact Hessian matrices”. In: *Computers & Graphics* 78 (2019), pp. 1–11.
- [120] I. S. Godage et al. “Dynamics for variable length multisection continuum arms”. In: *The International Journal of Robotics Research* 35.6 (2016), pp. 695–722.

- [121] I. S. Godage, R. J. Webster, and I. D. Walker. “Center-of-Gravity-Based Approach for Modeling Dynamics of Multisection Continuum Arms”. In: *IEEE Transactions on Robotics* 35.5 (2019), pp. 1097–1108.
- [122] R. Grosse. “Taylor Approximations”. In: *Neural Network Training Dynamics. Lecture Notes, University of Toronto* (2022).
- [123] J. Watt, R. Borhani, and A. Katsaggelos. *Machine Learning Refined: Foundations, Algorithms, and Applications*. Cambridge University Press, 2016.
- [124] J. V. Lambers and A. C. Sumner. *Explorations in Numerical Analysis*. World Scientific, 2018.
- [125] T. G. Kolda and B. W. Bader. “Tensor Decompositions and Applications”. In: *SIAM Review* 51.3 (2009), pp. 455–500.
- [126] M. Skorski. “Chain rules for hessian and higher derivatives made easy by tensor calculus”. In: *arXiv preprint arXiv:1911.13292* (2019).
- [127] H. Dai, G. Izatt, and R. Tedrake. “Global inverse kinematics via mixed-integer convex optimization”. In: *The International Journal of Robotics Research* 38.12-13 (2019), pp. 1420–1441.
- [128] T. Yenamandra et al. “Convex Optimisation for Inverse Kinematics”. In: *International Conference on 3D Vision (3DV)*. 2019, pp. 318–327.
- [129] S. M. Mustaza; et al. “Dynamic Modeling of Fiber-Reinforced Soft Manipulator: A Visco-Hyperelastic Material-Based Continuum Mechanics Approach”. In: *Soft Robotics* 6.3 (2019), pp. 305–317.
- [130] Y. Sun et al. “A miniature soft robotic manipulator based on novel fabrication methods”. In: *IEEE Robotics and Automation Letters* 1.2 (2016), pp. 617–623.
- [131] J. G. Webster and H. Eren. *Measurement, Instrumentation, and Sensors Handbook: Electromagnetic, Optical, Radiation, Chemical, and Biomedical Measurement*. CRC Press, 2014, p. 1921.
- [132] H. I. Christensen and G. D. Hager. “Sensing and Estimation”. In: *Springer Handbook of Robotics*. Ed. by B. Siciliano and O. Khatib. Cham: Springer International Publishing, 2016. Chap. 5, pp. 91–112.
- [133] T. Helps and J. Rossiter. “Proprioceptive Flexible Fluidic Actuators Using Conductive Working Fluids”. In: *Soft Robotics* 5.2 (2017), pp. 175–189.
- [134] Y. Yang and Y. Chen. “Innovative Design of Embedded Pressure and Position Sensors for Soft Actuators”. In: *IEEE Robotics and Automation Letters* 3.2 (2018), pp. 656–663.
- [135] L. Rupert, T. Duggan, and M. D. Killpack. “Improved Continuum Joint Configuration Estimation Using a Linear Combination of Length Measurements and Optimization of Sensor Placement”. In: *Frontiers in Robotics and AI* 8 (2021).
- [136] L. O. Tiziani and F. L. Hammond. “Optical Sensor-Embedded Pneumatic Artificial Muscle for Position and Force Estimation”. In: *Soft Robotics* 7.4 (2020), pp. 462–477.
- [137] J. Fraś et al. “New STIFF-FLOP module construction idea for improved actuation and sensing”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2015.
- [138] J. Malloch. “A Framework and Tools for Mapping of Digital Musical Instruments”. PhD Thesis. McGill University, 2013.
- [139] A. M. Hart, T. Cahoon, and F. L. Hammond. “Measuring multimodal deformations in soft inflatable actuators using embedded strain sensors”. In: *26th*

- IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*. 2017, pp. 263–270.
- [140] V. Wall, G. Zöller, and O. Brock. “A method for sensorizing soft actuators and its application to the RBO hand 2”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2017, pp. 4965–4970.
- [141] D. Hu et al. “Coupling Field Simulation of Soft Capacitive Sensors Toward Soft Robot Perception”. In: *IEEE Sensors Journal* 23.10 (2023), pp. 10588–10596.
- [142] Y. Kozhubaev et al. “Incremental Machine Learning for Soft Pneumatic Actuators with Symmetrical Chambers”. In: *Symmetry* 15.6 (2023). 1206.
- [143] F. Khan et al. “Curvature, twist and pose measurements using fiber Bragg gratings in multi-core fiber: A comparative study between helical and straight core fibers”. In: *Sensors and Actuators A: Physical* 317 (2021), p. 112442.
- [144] H. Wang et al. “Shape Detection Algorithm for Soft Manipulator Based on Fiber Bragg Gratings”. In: *IEEE/ASME Transactions on Mechatronics* 21.6 (2016), pp. 2977–2982.
- [145] J. Wei et al. “Novel Integrated Helical Design of Single Optic Fiber for Shape Sensing of Flexible Robot”. In: *IEEE Sensors Journal* 17.20 (2017), pp. 6627–6636.
- [146] R. Xu, A. Yurkewich, and R. V. Patel. “Curvature, Torsion, and Force Sensing in Continuum Robots Using Helically Wrapped FBG Sensors”. In: *IEEE Robotics and Automation Letters* 1.2 (2016), pp. 1052–1059.
- [147] L. A. Fernandes et al. “Fiber optic stress-independent helical torsion sensor”. In: *Optics Letters* 40.4 (2015), pp. 657–660.
- [148] G. Yin et al. “Simultaneous Measurement of Bending and Torsion in Optical Fiber Shape Sensor”. In: *Journal of Lightwave Technology* 41.6 (2023), pp. 1851–1857.
- [149] K. Tapp. *Differential geometry of curves and surfaces*. Springer International Publishing, 2016.
- [150] R. G. Budynas, J. K. Nisbett, and J. E. Shigley. *Shigley’s Mechanical Engineering Design*. 9th. McGraw-Hill series in mechanical engineering. McGraw-Hill, 2011.
- [151] L. Wang and N. Simaan. “Geometric Calibration of Continuum Robots: Joint Space and Equilibrium Shape Deviations”. In: *IEEE Transactions on Robotics* 35.2 (2019), pp. 387–402.
- [152] J. C. Case et al. “Robotic Skins That Learn to Control Passive Structures”. In: *IEEE Robotics and Automation Letters* 4.3 (2019), pp. 2485–2492.
- [153] X. Li et al. “Advantageous Strain Sensing Performances of FBG Strain Sensors Equipped with Planar UV-Curable Resin”. In: *Sensors* 23.5 (2023). 2811.
- [154] T. Kim et al. “Soft and Deformable Sensors Based on Liquid Metals”. In: *Sensors* 19.19 (2019).
- [155] M. S. Xavier, A. J. Fleming, and Y. K. Yong. “Finite Element Modeling of Soft Fluidic Actuators: Overview and Recent Developments”. In: *Advanced Intelligent Systems* 3.2 (2021), p. 2000187.
- [156] C. Tawk and G. Alici. “Finite Element Modeling in the Design Process of 3D Printed Pneumatic Soft Actuators and Sensors”. In: *Robotics* 9.3 (2020). 52.
- [157] P. Polygerinos et al. “Modeling of Soft Fiber-Reinforced Bending Actuators”. In: *IEEE Transactions on Robotics* 31.3 (2015), pp. 778–789.

- [158] Y. Elsayed et al. "Finite Element Analysis and Design Optimization of a Pneumatically Actuating Silicone Module for Robotic Surgery Applications". In: *Soft Robotics* 1.4 (2014), pp. 255–262.
- [159] P. Moseley et al. "Modeling, Design, and Development of Soft Pneumatic Actuators with Finite Element Method ". In: *Advanced Engineering Materials* 18.6 (2016), pp. 978–988.
- [160] X. Peng et al. "Dimension optimization of pneumatically actuated soft continuum manipulators". In: *2nd IEEE International Conference on Soft Robotics (RoboSoft)*. IEEE, 2019, pp. 13–18.
- [161] A. El Hami and B. Radi. *Fluid-structure interactions and uncertainties: ansys and fluent tools*. Vol. 6. John Wiley & Sons, 2017.
- [162] E. T. Roche et al. "A Bioinspired Soft Actuated Material". In: *Advanced Materials* 26.8 (2014), pp. 1200–1206.
- [163] S. K. Chimakurthi et al. "ANSYS Workbench System Coupling: a state-of-the-art computational framework for analyzing multiphysics problems". In: *Engineering with Computers* 34.2 (2018), pp. 385–411.
- [164] M. Pozzi et al. "Efficient FEM-Based Simulation of Soft Robots Modeled as Kinematic Chains". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2018.
- [165] K. Wandke and Y. Z. "MOOSE-Based Finite Element Hyperelastic Modeling for Soft Robot Simulations". In: *IEEE Access* 9 (2021), pp. 139627–139635.
- [166] H. Wang et al. "A Novel Soft Robotic Hand Design With Human-Inspired Soft Palm: Achieving a Great Diversity of Grasps". In: *IEEE Robotics & Automation Magazine* 28.2 (2021), pp. 37–49.
- [167] F. Faure et al. "SOFA: A Multi-Model Framework for Interactive Physical Simulation". In: *Soft Tissue Biomechanical Modeling for Computer Assisted Surgery*. Ed. by Y. Payan. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 283–321.
- [168] E. Coevoet et al. "Software toolkit for modeling, simulation, and control of soft robots". In: *Advanced Robotics* 31.22 (2017), pp. 1208–1224.
- [169] C. Duriez. "Control of elastic soft robots based on real-time finite element method". In: *IEEE International Conference on Robotics and Automation*. 2013, pp. 3982–3987.
- [170] F. Largilliere et al. "Real-time control of soft-robots using asynchronous finite element modeling". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2015, pp. 2550–2555.
- [171] M. Thieffry et al. "Control Design for Soft Robots Based on Reduced-Order Model". In: *IEEE Robotics and Automation Letters* 4.1 (2019), pp. 25–32.
- [172] W. Chen et al. "Fabrication and Dynamic Modeling of Bidirectional Bending Soft Actuator Integrated with Optical Waveguide Curvature Sensor". In: *Soft Robotics* 6.4 (2019), pp. 495–506.
- [173] A. H. Khan and S. Li. "Sliding Mode Control With PID Sliding Surface for Active Vibration Damping of Pneumatically Actuated Soft Robots". In: *IEEE Access* 8 (2020), pp. 88793–88800.
- [174] M. Bäcker, E. Knoop, and C. Schumacher. "Design and Control of Soft Robots Using Differentiable Simulation". In: *Current Robotics Reports* 2.2 (2021), pp. 211–221.

- [175] Y. Hu et al. “ChainQueen: A Real-Time Differentiable Physical Simulator for Soft Robotics”. In: *International Conference on Robotics and Automation (ICRA)*. 2019, pp. 6265–6271.
- [176] A. Spielberg et al. “Advanced soft robot modeling in ChainQueen”. In: *Robotica* 41.1 (2023), pp. 74–104.
- [177] T. Du et al. “Diffpd: Differentiable projective dynamics”. In: *ACM Transactions on Graphics (TOG)* 41.2 (2021), pp. 1–21.
- [178] T. Du et al. “Underwater soft robot modeling and control with differentiable simulation”. In: *IEEE Robotics and Automation Letters* 6.3 (2021), pp. 4994–5001.
- [179] J. Chen et al. “Manipulation task simulation of a soft pneumatic gripper using ros and gazebo”. In: *IEEE International Conference on Real-time Computing and Robotics (RCAR)*. IEEE, 2018, pp. 378–383.
- [180] B. Willis and P. Liu. “Simulation, design and control of a soft robotic arm with integrated bending sensing”. In: *20th International Conference on Advanced Robotics (ICAR)*. IEEE, 2021, pp. 26–31.
- [181] A. T. Mathew et al. “Sorosim: A matlab toolbox for hybrid rigid-soft robots based on the geometric variable-strain approach”. In: *IEEE Robotics & Automation Magazine* (2022).
- [182] S. Kriegman et al. “Simulating the evolution of soft and rigid-body robots”. In: *Proceedings of the genetic and evolutionary computation conference companion (GECCO)*. 2017, pp. 1117–1120.
- [183] J. Hiller and H. Lipson. “Dynamic simulation of soft multimaterial 3d-printed objects”. In: *Soft robotics* 1.1 (2014), pp. 88–101.
- [184] J. Collins et al. “A review of physics simulators for robotic applications”. In: *IEEE Access* 9 (2021), pp. 51416–51431.
- [185] F. Schmitt et al. “Soft Robots Manufacturing: A Review”. In: *Frontiers in Robotics and AI* 5.84 (2018).
- [186] B. Tondu. “Modelling of the McKibben artificial muscle: A review”. In: *Journal of Intelligent Material Systems and Structures* 23.3 (2012), pp. 225–253.
- [187] J. Fras and K. Althoefer. “Soft Fiber-Reinforced Pneumatic Actuator Design and Fabrication: Towards Robust, Soft Robotic Systems”. In: *Annual Conference Towards Autonomous Robotic Systems*. 2019, pp. 103–114.
- [188] J. Yan et al. “Design and Test of a New Spiral Driven Pure Torsional Soft Actuator”. In: *Intelligent Robotics and Applications*. Ed. by Y. Huang et al. 2017, pp. 127–139.
- [189] S. M. H. Sadati et al. “A Geometry Deformation Model for Braided Continuum Manipulators”. In: *Frontiers in Robotics and AI* 4.22 (2017).
- [190] F. Connolly et al. “Mechanical Programming of Soft Actuators by Varying Fiber Angle”. In: *Soft Robotics* 2.1 (2015), pp. 26–32.
- [191] E. W. Hawkes, D. L. Christensen, and A. M. Okamura. “Design and implementation of a 300% strain soft artificial muscle”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2016, pp. 4022–4029.
- [192] A. B. Dawood et al. “Fusing Dexterity and Perception for Soft Robot-Assisted Minimally Invasive Surgery: What We Learnt from STIFF-FLOP”. In: *Applied Sciences* 11.14 (2021), p. 6586.

- [193] M. Cianchetti et al. “STIFF-FLOP surgical manipulator: Mechanical design and experimental characterization of the single module”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2013.
- [194] S. H. Sadati et al. “A geometry deformation model for braided continuum manipulators”. In: *Frontiers in Robotics and AI* 4 (2017), p. 22.
- [195] T. C. Searle et al. “An optical curvature sensor for flexible manipulators”. In: *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 4415–4420.
- [196] J. Fr as et al. “Soft manipulator actuation module—with reinforced chambers”. In: *Soft and Stiffness-controllable Robotics Solutions for Minimally Invasive Surgery*. River Publishers, 2018, pp. 47–63.
- [197] S. M. H. Sadati et al. “Control Space Reduction and Real-Time Accurate Modeling of Continuum Manipulators Using Ritz and Ritz–Galerkin Methods”. In: *IEEE Robotics and Automation Letters* 3.1 (2018), pp. 328–335.
- [198] H. Abidi et al. “Highly dexterous 2-module soft robot for intra-organ navigation in minimally invasive surgery”. In: *The International Journal of Medical Robotics and Computer Assisted Surgery* 14.1 (2018), e1875.
- [199] W. King et al. “Design and Characterisation of a Variable Stiffness Soft Actuator Based on Tendon Twisting”. In: *Towards Autonomous Robotic Systems: 22nd Annual Conference (TAROS)*. Springer, 2021, pp. 402–407.
- [200] Q. Xie et al. “Design and modeling of a hydraulic soft actuator with three degrees of freedom”. In: *Smart Materials and Structures* 29.12 (2020), p. 125017.
- [201] Y. Toshimitsu et al. “SoPrA: Fabrication & Dynamical Modeling of a Scalable Soft Continuum Robotic Arm with Integrated Proprioceptive Sensing”. In: *arXiv preprint arXiv:2103.10726 [cs.RO]* (2021).
- [202] J. Yan et al. “A Wide-Range Stiffness-Tunable Soft Actuator Inspired by Deep-Sea Glass Sponges”. In: *Soft Robotics* (2021).
- [203] L. Guo et al. “Design and experiments of pneumatic soft actuators”. In: *Robotica* 39.10 (2021), pp. 1806–1815.
- [204] B. Caasenbrood, A. Pogromsky, and H. Nijmeijer. “Control-Oriented Models for Hyperelastic Soft Robots Through Differential Geometry of Curves”. In: *Soft Robotics* 10.1 (2023), pp. 129–148.
- [205] Z. Chen et al. “Pneumatically actuated soft robotic arm for adaptable grasping”. In: *Acta mechanica solida sinica* 31 (2018), pp. 608–622.
- [206] R. K. Katschmann et al. “Dynamically Closed-Loop Controlled Soft Robotic Arm using a Reduced Order Finite Element Model with State Observer”. In: *2nd IEEE International Conference on Soft Robotics (RoboSoft)*. 2019, pp. 717–724.
- [207] A. Al-Ibadi et al. “Novel design and position control strategy of a soft robot arm”. In: *Robotics* 7.4 (2018), p. 72.
- [208] T. Wang, L. Ge, and G. Gu. “Programmable design of soft pneu-net actuators with oblique chambers can generate coupled bending and twisting motions”. In: *Sensors and Actuators A: Physical* 271 (2018), pp. 131–138.
- [209] L. Cappello et al. “Exploiting textile mechanical anisotropy for fabric-based pneumatic actuators”. In: *Soft robotics* 5.5 (2018), pp. 662–674.
- [210] G. Krishnan et al. “Evaluating mobility behavior of fluid filled fiber-reinforced elastomeric enclosures”. In: *International Design Engineering Technical Con-*

- ferences and Computers and Information in Engineering Conference*. Vol. 45035. American Society of Mechanical Engineers, 2012, pp. 1089–1099.
- [211] J. Bishop-Moser and S. Kota. “Design and modeling of generalized fiber-reinforced pneumatic soft actuators”. In: *IEEE Transactions on Robotics* 31.3 (2015), pp. 536–545.
 - [212] F. Connolly, C. J. Walsh, and K. Bertoldi. “Automatic design of fiber-reinforced soft actuators for trajectory matching”. In: *Proceedings of the National Academy of Sciences* 114.1 (2017), pp. 51–56.
 - [213] S. Kurumaya et al. “A modular soft robotic wrist for underwater manipulation”. In: *Soft robotics* 5.4 (2018), pp. 399–409.
 - [214] G. Singh and G. Krishnan. “Designing fiber-reinforced soft actuators for planar curvilinear shape matching”. In: *Soft robotics* 7.1 (2020), pp. 109–121.
 - [215] G. Agarwal et al. “Stretchable Materials for Robust Soft Actuators towards Assistive Wearable Devices”. In: *Scientific Reports* 6 (2016), p. 34224.
 - [216] D. Rus and M. T. Tolley. “Design, fabrication and control of soft robots”. In: *Nature* 521.7553 (2015), pp. 467–475.
 - [217] M. Memarian, R. Gorbet, and D. Kulić. “Modelling and experimental analysis of a novel design for soft pneumatic artificial muscles”. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 1718–1724.
 - [218] L. Paez, G. Agarwal, and J. Paik. “Design and analysis of a soft pneumatic actuator with origami shell reinforcement”. In: *Soft Robotics* 3.3 (2016), pp. 109–119.
 - [219] O. Byrne et al. “Additive Manufacture of Composite Soft Pneumatic Actuators”. In: *Soft Robotics* 5.6 (2018), pp. 726–736.
 - [220] Y. Noh et al. “A three-axial body force sensor for flexible manipulators”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 6388–6393.
 - [221] H.-C. Fu et al. “Interfacing Soft and Hard: A Spring Reinforced Actuator”. In: *Soft Robotics* 7.1 (2020), pp. 44–58.
 - [222] Y. Li, X. Wang, and K.-W. Kwok. “Towards Adaptive Continuous Control of Soft Robotic Manipulator using Reinforcement Learning”. In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 7074–7081.
 - [223] C. Sun et al. “A hybrid continuum robot based on pneumatic muscles with embedded elastic rods”. In: *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 234.1 (2020), pp. 318–328.
 - [224] T. E. Pillsbury, N. M. Wereley, and Q. Guan. “Comparison of contractile and extensile pneumatic artificial muscles”. In: *Smart Materials and Structures* 26.9 (2017).
 - [225] J. Garbulinski, S. C. Balasankula, and N. M. Wereley. “Characterization and Analysis of Extensile Fluidic Artificial Muscles”. In: *Actuators* 10.2 (2021), p. 26.
 - [226] D. Sangian. “From Traditional Braiding Methods to Additive Manufacturing for Fabricating McKibben Artificial Muscles”. In: *Biomedical Journal of Scientific & Technical Research* 38.5 (2021), pp. 30728–30735.

- [227] T. Nozaki and T. Noritsugu. “Motion analysis of McKibben type pneumatic rubber artificial muscle with finite element method”. In: *International Journal of Automation Technology* 8.2 (2014), pp. 147–158.
- [228] B. Wang et al. “A bimorph pneumatic bending actuator by control of fiber braiding angle”. In: *Sensors and Actuators A: Physical* 257 (2017), pp. 173–184.
- [229] M. G. Antonelli et al. “Numerical modelling and experimental validation of a McKibben pneumatic muscle actuator”. In: *Journal of Intelligent Material Systems and Structures* 28.19 (2017), pp. 2737–2748.
- [230] J. G. Webster. *The Measurement, Instrumentation, and Sensors: Handbook*. CRC Press, 1999.
- [231] A. S. Morris. *Measurement and Instrumentation Principles*. Elsevier Science, 2001.
- [232] C. W. De Silva. *Sensor Systems: Fundamentals and Applications*. CRC Press, 2016.
- [233] Y. Luo et al. “Technology Roadmap for Flexible Sensors”. In: *ACS Nano* 17.6 (2023), pp. 5211–5295.
- [234] C. Hegde et al. “Sensing in soft robotics”. In: *ACS nano* 17.16 (2023), pp. 15277–15307.
- [235] K. Kruusamäe et al. “Self-sensing ionic polymer actuators: a review”. In: *Actuators* 4.1 (2015), pp. 17–38.
- [236] T. Hainsworth et al. “A fabrication free, 3D printed, multi-material, self-sensing soft actuator”. In: *IEEE Robotics and Automation Letters* 5.3 (2020), pp. 4118–4125.
- [237] C.-Y. Lo et al. “Highly stretchable self-sensing actuator based on conductive photothermally-responsive hydrogel”. In: *Materials Today* 50 (2021), pp. 35–43.
- [238] C. B. Cooper et al. “Stretchable Capacitive Sensors of Torsion, Strain, and Touch Using Double Helix Liquid Metal Fibers”. In: *Advanced Functional Materials* 27.20 (2017), p. 1605630.
- [239] D. Y. Choi et al. “Highly Stretchable, Hysteresis-Free Ionic Liquid-Based Strain Sensor for Precise Human Motion Monitoring”. In: *ACS Applied Materials & Interfaces* 9.2 (2017), pp. 1770–1780.
- [240] C. Liu et al. “Multifunctional highly sensitive multiscale stretchable strain sensor based on a graphene/glycerol–KCl synergistic conductive network”. In: *ACS applied materials & interfaces* 10.37 (2018), pp. 31716–31724.
- [241] E. L. White, J. C. Case, and R. K. Kramer. “Multi-mode strain and curvature sensors for soft robotic applications”. In: *Sensors and Actuators A: Physical* 253 (2017), pp. 188–197.
- [242] Y. Mengüç et al. “Wearable soft sensing suit for human gait measurement”. In: *The International Journal of Robotics Research* 33.14 (2014), pp. 1748–1764.
- [243] L. O. Tiziani, T. W. Cahoon, and F. L. Hammond. “Sensorized pneumatic muscle for force and stiffness control”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2017, pp. 5545–5552.
- [244] J. Morrow et al. “Improving Soft Pneumatic Actuator fingers through integration of soft sensors, position and force control, and rigid fingernails”. In:

- IEEE International Conference on Robotics and Automation (ICRA)*. 2016, pp. 5024–5031.
- [245] G.-S. Cho and Y.-J. Park. “Soft Gripper with EGaIn Soft Sensor for Detecting Grasp Status”. In: *Applied Sciences* 11.15 (2021), p. 6957.
- [246] Y. Hao et al. “A review of smart materials for the boost of soft actuators, soft sensors, and robotics applications”. In: *Chinese Journal of Mechanical Engineering* 35.1 (2022), pp. 1–16.
- [247] S. Stassi et al. “Flexible Tactile Sensing Based on Piezoresistive Composites: A Review”. In: *Sensors* 14.3 (2014), p. 5296.
- [248] G. Selleri et al. “Self-sensing soft skin based on piezoelectric nanofibers”. In: *Polymers* 15.2 (2023), p. 280.
- [249] M. Amjadi et al. “Stretchable, Skin-Mountable, and Wearable Strain Sensors and Their Potential Applications: A Review”. In: *Advanced Functional Materials* 26.11 (2016), pp. 1678–1698.
- [250] A. Fiorillo, C. Critello, and S. Pullano. “Theory, technology and applications of piezoresistive sensors: A review”. In: *Sensors and Actuators A: Physical* 281 (2018), pp. 156–175.
- [251] A. Georgopoulou and F. Clemens. “Piezoresistive elastomer-based composite strain sensors and their applications”. In: *ACS Applied Electronic Materials* 2.7 (2020), pp. 1826–1842.
- [252] J. L. Molnar et al. “Optical sensing and control methods for soft pneumatically actuated robotic manipulators”. In: *IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 3355–3362.
- [253] K.-H. Ha et al. “Soft capacitive pressure sensors: trends, challenges, and perspectives”. In: *ACS nano* 16.3 (2022), pp. 3442–3448.
- [254] S. Homayoonnia and S. Zeinali. “Design and fabrication of capacitive nanosensor based on MOF nanoparticles as sensing layer for VOCs detection”. In: *Sensors and Actuators B: Chemical* 237 (2016), pp. 776–786.
- [255] A. J. Cheng et al. “Recent Advances of Capacitive Sensors: Materials, Microstructure Designs, Applications, and Opportunities”. In: *Advanced Materials Technologies* (2023), p. 2201959.
- [256] A. Atalay et al. “Batch Fabrication of Customizable Silicone-Textile Composite Capacitive Strain Sensors for Human Motion Tracking”. In: *Advanced Materials Technologies* 2.9 (2017).
- [257] P. Cataldi et al. “Carbon nanofiber versus graphene-based stretchable capacitive touch sensors for artificial electronic skin”. In: *Advanced Science* 5.2 (2018), p. 1700587.
- [258] B. Li, A. K. Fontecchio, and Y. Visell. “Mutual capacitance of liquid conductors in deformable tactile sensing arrays”. In: *Applied Physics Letters* 108.1 (2016).
- [259] M. McCandless, F. J. Wise, and S. Russo. “A soft robot with three dimensional shape sensing and contact recognition multi-modal sensing via tunable soft optical sensors”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 573–580.
- [260] C. To, T. L. Hellebrekers, and Y.-L. Park. “Highly stretchable optical sensors for pressure, strain, and curvature measurement”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2015, pp. 5898–5903.

- [261] N. F. Lepora. “Soft biomimetic optical tactile sensing with the TacTip: A review”. In: *IEEE Sensors Journal* 21.19 (2021), pp. 21131–21143.
- [262] A. G. Leal-Junior et al. “Highly stretchable polymer optical fiber for mechanical sensing in artificial tendons: Towards novel sensors for soft robotics”. In: *Actuators*. Vol. 9. MDPI, 2020, p. 125.
- [263] K.-Y. Lin, A. Gamboa-Gonzalez, and M. Wehner. “Soft Robotic Sensing, Proprioception via Cable and Microfluidic Transmission”. In: *Electronics* 10.24 (2021), p. 3166.
- [264] Z. Xing et al. “Inductive strain sensor with high repeatability and ultra-low hysteresis based on mechanical spring”. In: *IEEE Sensors Journal* 20.24 (2020), pp. 14670–14675.
- [265] A. R. P. Mathai, T. Stalin, and P. V. y Alvarado. “Flexible fiber inductive coils for soft robots and wearable devices”. In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 5711–5718.
- [266] S. K. Sahu et al. “A Spring-Based Inductive Sensor for Soft and Flexible Robots”. In: *IEEE Sensors Journal* 22.20 (2022), pp. 19931–19940.
- [267] S. Joe et al. “Sensing deformation in vacuum driven foam-based actuator via inductive method”. In: *Frontiers in Robotics and AI* 8 (2021), p. 742885.
- [268] M. Luo et al. “Toward Modular Soft Robotics: Proprioceptive Curvature Sensing and Sliding-Mode Control of Soft Bidirectional Bending Modules”. In: *Soft Robotics* 4.2 (2017), pp. 117–125.
- [269] J. C. Case, E. L. White, and R. K. Kramer. “Soft Material Characterization for Robotic Applications”. In: *Soft Robotics* 2.2 (2015), pp. 80–87.
- [270] A. Al-Azzawi et al. “Modelling of Uniaxial EGaIn-Based Strain Sensors for Proprioceptive Sensing of Soft Robots”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2019.
- [271] M. Á. Rodríguez-Valverde and M. Tirado-Miranda. “A simpler derivation of the integral formula of electrical resistance”. In: *European journal of physics* 30.4 (2009), p. L47.
- [272] Z. Wang et al. “STEV: Stretchable Triboelectric E-skin enabled Proprioceptive Vibration Sensing for Soft Robot”. In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 588–593.
- [273] P. K. Ghosh and P. Sundaravadivel. “Stretchable Sensors for Soft Robotic Grippers in Edge-Intelligent IoT Applications”. In: *Sensors* 23.8 (2023), p. 4039.
- [274] R. Chirila, A. S. Dahiya, and R. Dahiya. “Micromolding-in-Capillary based Fabrication of Liquid Metal Patterned Structures for Soft Robotics”. In: *2023 IEEE Applied Sensing Conference (APSCON)*. IEEE, 2023, pp. 1–3.
- [275] S. Chen et al. “Liquid Metal Smart Materials toward Soft Robotics”. In: *Advanced Intelligent Systems* (2023), p. 2200375.
- [276] C. Majidi, R. Kramer, and R. J. Wood. “A non-differential elastomer curvature sensor for softer-than-skin electronics”. In: *Smart Materials and Structures* 20.10 (2011), p. 105017.
- [277] Y.-L. Park, B.-R. Chen, and R. J. Wood. “Design and fabrication of soft artificial skin using embedded microchannels and liquid conductors”. In: *IEEE Sensors Journal* 12.8 (2012), pp. 2711–2718.
- [278] J. B. Chossat et al. “A Soft Strain Sensor Based on Ionic and Metal Liquids”. In: *IEEE Sensors Journal* 13.9 (2013), pp. 3405–3414.

- [279] C. Zlebici et al. "Comparison of resistive and capacitive strain gauge sensors printed on polyimide substrate using ink-jet printing technology". In: *Proceedings of 29th International Conference on Microelectronics (MIEL)*. IEEE, 2014, pp. 141–144.
- [280] N. Farrow and N. Correll. "A soft pneumatic actuator that can sense grasp and touch". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2015, pp. 2317–2323.
- [281] F. Pineda et al. "Using electrofluidic devices as hyper-elastic strain sensors: Experimental and theoretical analysis". In: *Microelectronic Engineering* 144 (2015), pp. 27–31.
- [282] J. T. Overvelde et al. "Mechanical and electrical numerical analysis of soft liquid-embedded deformation sensors analysis". In: *Extreme Mechanics Letters* 1 (2014), pp. 42–46.
- [283] K. Min-gu et al. "Size-Scalable and High-Density Liquid-Metal-Based Soft Electronic Passive Components and Circuits Using Soft Lithography". In: *Advanced Functional Materials* 27.3 (2017), p. 1604466.
- [284] E. L. White, J. C. Case, and R. K. Kramer. "Multi-Element Strain Gauge Modules for Soft Sensory Skins". In: *IEEE Sensors Journal* 16.8 (2016), pp. 2607–2616.
- [285] S. Han et al. "Use of Deep Learning for Characterization of Microfluidic Soft Sensors". In: *IEEE Robotics and Automation Letters* 3.2 (2018), pp. 873–880.
- [286] H. Yang and W. Wu. "A review: Machine learning for strain sensor-integrated soft robots". In: *Frontiers in Electronic Materials* 2 (2022), p. 1000781.
- [287] H. Yang et al. "Automatic strain sensor design via active learning and data augmentation for soft machines". In: *Nature Machine Intelligence* 4.1 (2022), pp. 84–94.
- [288] R. K. Kramer et al. "Soft curvature sensors for joint angle proprioception". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2011, pp. 1919–1926.
- [289] J. C. Case, E. L. White, and R. K. Kramer. "Sensor enabled closed-loop bending control of soft beams". In: *Smart Materials and Structures* 25.4 (2016), p. 045018.
- [290] H.-S. Shin et al. "Enhanced performance of microfluidic soft pressure sensors with embedded solid microspheres". In: *Journal of Micromechanics and Microengineering* 26.2 (2016), p. 025011.
- [291] N. Mangado et al. "Analysis of uncertainty and variability in finite element computational models for biomedical engineering: characterization and propagation". In: *Frontiers in Bioengineering and Biotechnology* 4 (2016), p. 85.
- [292] N.-H. Kim. *Introduction to nonlinear finite element analysis*. New York, USA: Springer Science & Business Media, 2014.
- [293] Y. Zhang et al. "Construction of liquid metal-based soft microfluidic sensors via soft lithography". In: *Journal of Nanobiotechnology* 20.1 (2022), p. 246.
- [294] T. Dong et al. "Low-hysteresis Flexible Strain Sensors Based on Liquid Metal for Human-Robot Interaction". In: *International Conference on Intelligent Robotics and Applications*. Springer, 2023, pp. 243–252.
- [295] D. Kim et al. "Adaptive calibration of soft sensors using optimal transportation transfer learning for mass production and long-term usage". In: *Advanced Intelligent Systems* 2.6 (2020), p. 1900178.

- [296] J. Friedman, T. Hastie, and R. Tibshirani. *The Elements of Statistical Learning*. Vol. 1. Springer series in statistics New York, NY, USA: 2009.
- [297] A. Alwosheel, S. van Cranenburgh, and C. G. Chorus. “Is your dataset big enough? Sample size requirements when using artificial neural networks for discrete choice analysis”. In: *Journal of choice modelling* 28 (2018), pp. 167–182.
- [298] G. Li et al. “Skin-inspired quadruple tactile sensors integrated on a robot hand enable object recognition”. In: *Science Robotics* 5.49 (2020).
- [299] C. Watson and T. K. Morimoto. “Permanent magnet-based localization for growing robots in medical applications”. In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 2666–2673.
- [300] L. Fahrmeir et al. *Regression Models, Methods and Applications*. Springer Berlin, Heidelberg, 2013.
- [301] K. Tyagi et al. “Chapter 4 - Regression analysis”. In: *Artificial Intelligence and Machine Learning for EDGE Computing*. Ed. by R. Pandey et al. Academic Press, 2022, pp. 53–63.
- [302] J. Arkes. *Regression analysis: a practical introduction*. 1st. Taylor & Francis, 2019.
- [303] J. P. King et al. “Design, modeling, and control of pneumatic artificial muscles with integrated soft sensing”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2017, pp. 4985–4990.
- [304] R. A. Bilodeau, E. L. White, and R. K. Kramer. “Monolithic fabrication of sensors and actuators in a soft robotic gripper”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2015, pp. 2324–2329.
- [305] S. Han et al. “Use of Deep Learning for Characterization of Microfluidic Soft Sensors”. In: *IEEE Robotics and Automation Letters* 3.2 (2018), pp. 873–880.
- [306] A. Joubair et al. “Absolute accuracy analysis and improvement of a hybrid 6-DOF medical robot”. In: *Industrial Robot: An International Journal* 42.1 (2015), pp. 44–53.
- [307] R. M. Murray, Z. Li, and S. S. Sastry. *A mathematical introduction to robotic manipulation*. 1st edition. CRC press, 1994.
- [308] G. Runge, M. Wiese, and A. Raatz. “FEM-based training of artificial neural networks for modular soft robots”. In: *IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2017, pp. 385–392.
- [309] J. Angeles. *Rational kinematics*. Vol. 34. Springer Science & Business Media, 2013.
- [310] E. L. White, J. C. Case, and R. Kramer-Bottiglio. “A Soft Parallel Kinematic Mechanism”. In: *Soft Robotics* 5.1 (2018), pp. 36–53.
- [311] D. Zhuang et al. “A Finite Element Model and Performance Analysis of a Hybrid Continuum Robot”. In: *The 12th International Conference on Intelligent Robotics and Applications (ICIRA)*. 2019, pp. 171–182.
- [312] H. Shen et al. “Obstacle Avoidance Algorithm for 7-DOF Redundant Anthropomorphic Arm”. In: *Journal of Control Science and Engineering* 2015 (2015), p. 540259.
- [313] M. Kelemen et al. “A Novel Approach for a Inverse Kinematics Solution of a Redundant Manipulator”. In: *Applied Sciences* 8.11 (2018), p. 2229.
- [314] S. Dey. *Matrix Calculus*. Report. University of Southern California, April 2019.

- [315] M. M. Hammad and M. M. Yahia. *Mathematics for Machine Learning and Data Science: Optimization with Mathematica Applications*. Cornell University Library, arXiv.org, 2023.
- [316] J. R. Magnus and H. Neudecker. *Matrix differential calculus with applications in statistics and econometrics*. 3rd. John Wiley & Sons, 2019.
- [317] J. E. Gentle. *Matrix Algebra. Theory, Computations and Applications in Statistics*. 2nd. Springer International Publishing AG, 2017.
- [318] S. Ahmed and E. G. Saleeby. “On Volumes of Hyper-Ellipsoids”. In: *Mathematics Magazine* 91.1 (2018), pp. 43–50.
- [319] W. Marais and A. Göktogan. “A New Manipulability Measure for the Control of CRAM: A Cable-driven Remote Access Manipulator”. In: *Proceedings of the Australasian Conference on Robotics and Automation (ACRA-2017), Sydney, Australia*. 2017, pp. 11–13.
- [320] K. Dufour and W. Suleiman. “On integrating manipulability index into inverse kinematics solver”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 6967–6972.
- [321] M. Bamdad and M. M. Bahri. “Kinematics and Manipulability Analysis of a Highly Articulated Soft Robotic Manipulator”. In: *Robotica* 37.5 (2019), pp. 868–882.
- [322] J. Varela-Aldás et al. “Inverse kinematics of a redundant manipulator robot using constrained optimization”. In: *International Conference on Information Technology & Systems*. Springer, 2020, pp. 233–242.
- [323] Z. Zou. “A general method for the manipulability analysis of serial robot manipulators”. In: *International Journal of Advanced Robotic Systems* 20.6 (2023), p. 17298806231210665.
- [324] T. Yoshikawa. “Manipulability of Robotic Mechanisms”. In: *The International Journal of Robotics Research* 4.2 (1985), pp. 3–9.
- [325] J. Haviland and P. Corke. “A systematic approach to computing the manipulator Jacobian and Hessian using the elementary transform sequence”. In: *arXiv preprint arXiv:2010.08696* (2020).
- [326] M. Betancourt. “A geometric theory of higher-order automatic differentiation”. In: *arXiv preprint arXiv:1812.11592* (2018).
- [327] A. Hourtash. “The kinematic hessian and higher derivatives”. In: *International Symposium on Computational Intelligence in Robotics and Automation*. IEEE, 2005, pp. 169–174.
- [328] Y. Lu et al. “Kinematics/statics analysis of a novel serial-parallel robotic arm with hand”. In: *Journal of Mechanical Science and Technology* 29 (2015), pp. 4407–4416.
- [329] I. S. Godage et al. “Real-time dynamic models for soft bending actuators”. In: *IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2018, pp. 1310–1315.
- [330] T. Sun and S. Yang. “An approach to formulate the Hessian matrix for dynamic control of parallel robots”. In: *IEEE/ASME Transactions on Mechatronics* 24.1 (2019), pp. 271–281.
- [331] Z. Fu et al. “Analytical expressions of serial manipulator jacobians and their high-order derivatives based on lie theory”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 7095–7100.

- [332] B. A. Jones and I. D. Walker. “Practical Kinematics for Real-Time Implementation of Continuum Robots”. In: *IEEE Transactions on Robotics* 22.6 (2006), pp. 1087–1099.
- [333] A. Dietrich, C. Ott, and A. Albu-Schäffer. “An overview of null space projections for redundant, torque-controlled robots”. In: *The International Journal of Robotics Research* 34.11 (2015), pp. 1385–1400.
- [334] M. Cui et al. “A new approach for determining damping factors in Levenberg-Marquardt algorithm for solving an inverse heat conduction problem”. In: *International Journal of Heat and Mass Transfer* 107 (2017), pp. 747–754.
- [335] J. R. Dahlquist and C. D. Kirkpatrick II. *Technical analysis: the complete resource for financial market technicians*. FT press, 2010.
- [336] R. Moradi, R. Berangi, and B. Minaei. “A survey of regularization strategies for deep models”. In: *Artificial Intelligence Review* 53.6 (2020), pp. 3947–3986.
- [337] S. Paternain, A. Mokhtari, and A. Ribeiro. “A Newton-Based Method for Nonconvex Optimization with Fast Evasion of Saddle Points”. In: *SIAM Journal on Optimization* 29.1 (2019), pp. 343–368.
- [338] L. Moroni. “The toric sections: a simple introduction”. In: *arXiv:1708.00803 [math.HO]* (2017).
- [339] M. L. Irons. *The geodesics of the torus*. Report. 2005. URL: <http://www.rdrop.com/~half/math/torus/torus.geodesics.pdf>.
- [340] P. Stadler. “The short ruler on the torus”. In: *Journal of Difference Equations and Applications* 25.9-10 (2019), pp. 1382–1403.
- [341] J. P. May. *A concise course in algebraic topology*. University of Chicago press, 1999.
- [342] R. V. Martinez et al. “Robotic Tentacles with Three-Dimensional Mobility Based on Flexible Elastomers”. In: *Advanced Materials* 25.2 (2013), pp. 205–212.
- [343] B. Nedjar. “An anisotropic viscoelastic fibre-matrix model at finite strains: Continuum formulation and computational aspects”. In: *Computer Methods in Applied Mechanics and Engineering* 196.9 (2007), pp. 1745–1756.
- [344] T. Lauwagie et al. “Mixed numerical–experimental identification of elastic properties of orthotropic metal plates”. In: *NDT & E International* 36.7 (2003), pp. 487–495.
- [345] P. Polygerinos et al. “Towards a soft pneumatic glove for hand rehabilitation”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2013, pp. 1512–1517.
- [346] G. E. Farin and G. Farin. *Curves and surfaces for CAGD: a practical guide*. Morgan Kaufmann, 2002.
- [347] P. Lan and A. A. Shabana. “Integration of B-spline geometry and ANCF finite element analysis”. In: *Nonlinear Dynamics* 61.1 (2010), pp. 193–206.
- [348] B. O’neill. *Elementary differential geometry*. Elsevier, 2006.
- [349] A. D. Polyanin and A. V. Manzhirov. *Handbook of mathematics for engineers and scientists*. Chapman and Hall/CRC, 2006.
- [350] K. S. Arun, T. S. Huang, and S. D. Blostein. “Least-Squares Fitting of Two 3-D Point Sets”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-9.5 (1987), pp. 698–700.

- [351] J. R. Vig and F. L. Walls. “A review of sensor sensitivity and stability”. In: *Proceedings of the IEEE/EIA International Frequency Control Symposium and Exhibition*. 2000, pp. 30–33.
- [352] H. Kong et al. “Filtering for Systems subject to Unknown Inputs without a priori Initial Information”. In: *Automatica* 120 (2020), p. 109122.
- [353] G. Decroly et al. “Design, characterization and optimization of a soft fluidic actuator for minimally invasive surgery”. In: *International Journal of Computer Assisted Radiology and Surgery* 15.2 (2020), pp. 333–340.
- [354] Z. Song, Y. Li, and J. Hu. “Directional Torsion Sensor Based on a Two-Core Fiber with a Helical Structure”. In: *Sensors* 23.6 (2023), p. 2874.
- [355] J. Tolvanen, J. Hannu, and H. Jantunen. “Stretchable and washable strain sensor based on cracking structure for human motion monitoring”. In: *Scientific Reports* 8.1 (2018), p. 13241.
- [356] J. Lee et al. “Ultrasensitive strain sensor based on separation of overlapped carbon nanotubes”. In: *Small* 15.12 (2019), p. 1805120.
- [357] H. Souri et al. “Wearable and stretchable strain sensors: materials, sensing mechanisms, and applications”. In: *Advanced Intelligent Systems* 2.8 (2020), p. 2000039.
- [358] J. Park et al. “Soft Sensors and Actuators for Wearable Human–Machine Interfaces”. In: *Chemical Reviews* 124.4 (2024), pp. 1464–1534.
- [359] W. H. Press et al. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press, 2007.
- [360] NIST/SEMATECH. “Rational Functions”. In: *e-Handbook of Statistical Methods*. Accessed 8/Feb/2024. URL: <https://doi.org/10.18434/M32189>.
- [361] J. Dattorro. *Convex Optimization and Euclidean Distance Geometry*. 2nd. Meboo Publishing USA, 2019.
- [362] A. M. Solo. “Multidimensional matrix mathematics: Multidimensional matrix transpose, symmetry, antisymmetry, determinant, and inverse, part 4 of 6”. In: *Proceedings of the World Congress on Engineering (WCE)*. Vol. III. 2010.
- [363] K. B. Petersen and M. S. Pedersen. “The matrix cookbook”. In: *Technical University of Denmark* 7.15 (2008).
- [364] R. T. Jantzen. “Geodesics on the torus and other surfaces of revolution clarified using undergraduate physics tricks with bonus: nonrelativistic and relativistic Kepler problems”. In: *arXiv:1212.6206* (2012).
- [365] J. Oprea. *Differential Geometry and Its Applications*. 2nd. USA: The Mathematical Association of America, 2007.

Appendix A

Hessian Chain Rule (an Example)

Based on the analysis of [126], consider the following functions (all notations are irrelevant to the main body of the thesis):

$$\begin{aligned} f(x) : \mathbb{R}^n &\rightarrow \mathbb{R}, \\ g(y) : \mathbb{R}^m &\rightarrow \mathbb{R}^n. \end{aligned} \tag{A.1}$$

The Hessian of f with respect to y is defined as:

$$H(f \circ g)(y) = (J_g(y))^T H_f(g) J_g(y) + \sum_{k=1}^n \frac{\partial f}{\partial g_k} H_{g_k}(y). \tag{A.2}$$

Assuming $m = 2$ and $n = 3$, the vector y and the function $g(y)$ have the following forms:

$$\begin{aligned} y &= \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \in \mathbb{R}^m, \\ g(y) &= \begin{bmatrix} g_1 \\ g_2 \\ g_3 \end{bmatrix} \in \mathbb{R}^n. \end{aligned} \tag{A.3}$$

Assuming $f(g(y)) = f \in \mathbb{R}$, we can derive $J_g(y)$, $H_f(g)$, and $H_{g_i}(y)$ using the *numerator layout convention* [314]. Starting with the Jacobian $J_g(y)$ as follow:

$$J_g(y) = \frac{\partial g}{\partial y} = \begin{bmatrix} \frac{\partial g_1}{\partial y_1} & \frac{\partial g_1}{\partial y_2} \\ \frac{\partial g_2}{\partial y_1} & \frac{\partial g_2}{\partial y_2} \\ \frac{\partial g_3}{\partial y_1} & \frac{\partial g_3}{\partial y_2} \end{bmatrix} \in \mathbb{R}^{n \times m}. \tag{A.4}$$

Then, the Hessian $H_f(g)$ as follow:

$$H_f(g) = \begin{bmatrix} \frac{\partial^2 f}{\partial g_1^2} & \frac{\partial^2 f}{\partial g_1 \partial g_2} & \frac{\partial^2 f}{\partial g_1 \partial g_3} \\ \frac{\partial^2 f}{\partial g_2 \partial g_1} & \frac{\partial^2 f}{\partial g_2^2} & \frac{\partial^2 f}{\partial g_2 \partial g_3} \\ \frac{\partial^2 f}{\partial g_3 \partial g_1} & \frac{\partial^2 f}{\partial g_3 \partial g_2} & \frac{\partial^2 f}{\partial g_3^2} \end{bmatrix} \in \mathbb{R}^{n \times n}. \quad (\text{A.5})$$

The Hessian of $g(y)$ is a third-order tensor. Let us derive the gradient of $g(y)$ first, then use it to find the Hessian.

$$\nabla g(y) = \begin{bmatrix} \frac{\partial g_1}{\partial y_1} & \frac{\partial g_2}{\partial y_1} & \frac{\partial g_3}{\partial y_1} \\ \frac{\partial g_1}{\partial y_2} & \frac{\partial g_2}{\partial y_2} & \frac{\partial g_3}{\partial y_2} \end{bmatrix} \in \mathbb{R}^{m \times n}, \quad (\text{A.6})$$

which gives the Hessian as:

$$H_g(y) = \nabla^2 g(y) = \begin{bmatrix} \nabla \frac{\partial g_1}{\partial y_1} & \nabla \frac{\partial g_2}{\partial y_1} & \nabla \frac{\partial g_3}{\partial y_1} \\ \nabla \frac{\partial g_1}{\partial y_2} & \nabla \frac{\partial g_2}{\partial y_2} & \nabla \frac{\partial g_3}{\partial y_2} \end{bmatrix} \in \mathbb{R}^{m \times n \times m}, \quad (\text{A.7})$$

where

$$\begin{aligned} \nabla \frac{\partial g_1}{\partial y_1} &= \begin{bmatrix} \frac{\partial^2 g_1}{\partial y_1^2} \\ \frac{\partial^2 g_1}{\partial y_2 \partial y_1} \end{bmatrix}, \nabla \frac{\partial g_2}{\partial y_1} = \begin{bmatrix} \frac{\partial^2 g_2}{\partial y_1^2} \\ \frac{\partial^2 g_2}{\partial y_2 \partial y_1} \end{bmatrix}, \nabla \frac{\partial g_3}{\partial y_1} = \begin{bmatrix} \frac{\partial^2 g_3}{\partial y_1^2} \\ \frac{\partial^2 g_3}{\partial y_2 \partial y_1} \end{bmatrix}, \\ \nabla \frac{\partial g_1}{\partial y_2} &= \begin{bmatrix} \frac{\partial^2 g_1}{\partial y_1 \partial y_2} \\ \frac{\partial^2 g_1}{\partial y_2^2} \end{bmatrix}, \nabla \frac{\partial g_2}{\partial y_2} = \begin{bmatrix} \frac{\partial^2 g_2}{\partial y_1 \partial y_2} \\ \frac{\partial^2 g_2}{\partial y_2^2} \end{bmatrix}, \nabla \frac{\partial g_3}{\partial y_2} = \begin{bmatrix} \frac{\partial^2 g_3}{\partial y_1 \partial y_2} \\ \frac{\partial^2 g_3}{\partial y_2^2} \end{bmatrix}. \end{aligned} \quad (\text{A.8})$$

Rearranging gives the Hessian $H_g(y)$ as:

$$\begin{aligned} H_g(y)[:, :, 1] &= \begin{bmatrix} \frac{\partial^2 g_1}{\partial y_1^2} & \frac{\partial^2 g_2}{\partial y_1^2} & \frac{\partial^2 g_3}{\partial y_1^2} \\ \frac{\partial^2 g_1}{\partial y_1 \partial y_2} & \frac{\partial^2 g_2}{\partial y_1 \partial y_2} & \frac{\partial^2 g_3}{\partial y_1 \partial y_2} \end{bmatrix}, \\ H_g(y)[:, :, 2] &= \begin{bmatrix} \frac{\partial^2 g_1}{\partial y_2 \partial y_1} & \frac{\partial^2 g_2}{\partial y_2 \partial y_1} & \frac{\partial^2 g_3}{\partial y_2 \partial y_1} \\ \frac{\partial^2 g_1}{\partial y_2^2} & \frac{\partial^2 g_2}{\partial y_2^2} & \frac{\partial^2 g_3}{\partial y_2^2} \end{bmatrix}. \end{aligned} \quad (\text{A.9})$$

The Hessian $H_g(y)$ in (A.9) can be represented using the following notation [361]:

$$H_g(y) = [H_{g_1}(y) \quad H_{g_2}(y) \quad \cdots \quad H_{g_n}(y)] \in \mathbb{R}^{m \times n \times m}, \quad (\text{A.10})$$

where each $H_{g_k}(y) \in \mathbb{R}^{m \times m}$ and $k = 1, 2, \dots, n$ which can be represented as:

$$\begin{aligned}
H_{g_1}(y) &= \begin{bmatrix} \frac{\partial^2 g_1}{\partial y_1^2} & \frac{\partial^2 g_1}{\partial y_2 \partial y_1} \\ \frac{\partial^2 g_1}{\partial y_1 \partial y_2} & \frac{\partial^2 g_1}{\partial y_2^2} \end{bmatrix}, \\
H_{g_2}(y) &= \begin{bmatrix} \frac{\partial^2 g_2}{\partial y_1^2} & \frac{\partial^2 g_2}{\partial y_2 \partial y_1} \\ \frac{\partial^2 g_2}{\partial y_1 \partial y_2} & \frac{\partial^2 g_2}{\partial y_2^2} \end{bmatrix}, \\
H_{g_3}(y) &= \begin{bmatrix} \frac{\partial^2 g_3}{\partial y_1^2} & \frac{\partial^2 g_3}{\partial y_2 \partial y_1} \\ \frac{\partial^2 g_3}{\partial y_1 \partial y_2} & \frac{\partial^2 g_3}{\partial y_2^2} \end{bmatrix}.
\end{aligned} \tag{A.11}$$

Now, substituting in (A.2), gives:

$$H(f \circ g)(y) = H_1 + H_2, \tag{A.12}$$

where

$$\begin{aligned}
H_1 &= (J_g(y))^T H_f(g) J_g(y), \\
H_2 &= \frac{\partial f}{\partial g_1} H_{g_1}(y) + \frac{\partial f}{\partial g_2} H_{g_2} + \frac{\partial f}{\partial g_3} H_{g_3},
\end{aligned} \tag{A.13}$$

in which $H_1, H_2 \in \mathbb{R}^{m \times m}$.

Appendix B

Jacobian and Hessian Arrays

B.1 Derivative Rules

For the arbitrary functions $f()$, $x()$, $y()$, and $z()$, the following derivative rules [113] can be used:

$$\begin{aligned}f = \frac{x}{y} &\rightarrow \dot{f} = \frac{\dot{x}y - x\dot{y}}{y^2}, \\f = xy &\rightarrow \dot{f} = \dot{x}y + x\dot{y}, \\f = xyz &\rightarrow \dot{f} = \dot{x}yz + x\dot{y}z + xy\dot{z},\end{aligned}$$

where $(\dot{})$ represents differentiating with respect to time t .

B.2 Tensor Transpose

A tensor can be represented as a multidimensional array. The tensor order represents its number of dimensions. For example, a vector is a first-order tensor. Similarly, a matrix is a second-order tensor. A third-order tensor or higher is just called a tensor.

Figure B.1 shows an arbitrary third-order tensor $X \in \mathbb{R}^{I \times J \times K}$ where the mode-1 (columns) fibres $X_{:jk}$, mode-2 (row) fibres $X_{i:k}$, and mode-3 (tube) fibres $X_{ij:}$ are shown in different colours. Only one layer (or slice) of the tensor was coloured to provide easier illustration and visualisation.

Following the notation of [362], A tensor transpose is indicated by the letter T followed by the dimensions being transposed in parentheses, all in a superscript text. This transpose can be carried out for any two dimensions in the tensor. For example, $Y = X^{T(3,1)}$ indicates transposing the first and the third dimensions. The resulting tensor Y has the size $K \times J \times I$.

This transpose operation can be easily performed in MATLAB by using a single command, $Y = \text{permute}(X, [3 \ 2 \ 1])$, in which the second dimension remains in its original location while the first and third dimensions are swapped.

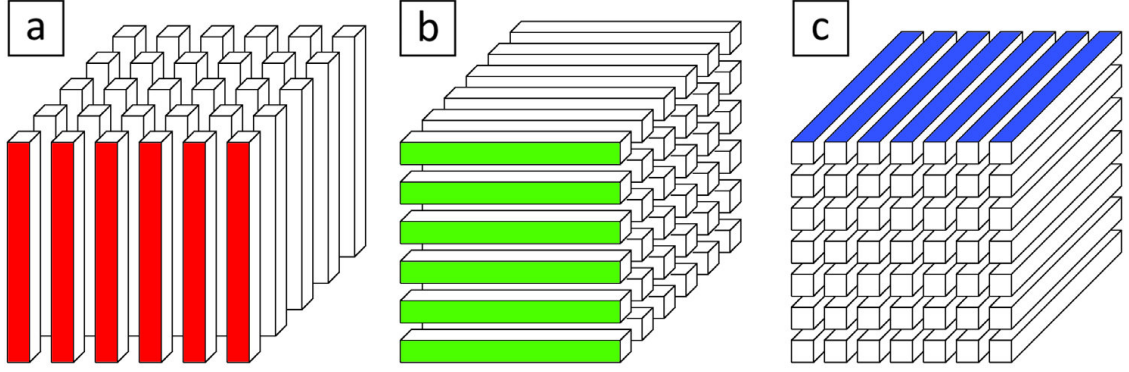


Figure B.1: Fibres of a third-order tensor where (a) shows mode-1 (columns) fibres $X_{:jk}$, (b) shows mode-2 (row) fibres $X_{i:k}$, and (c) shows mode-3 (tube) fibres $X_{ij:}$. The red, green, and blue colours are for illustration purpose only.

B.3 Tensor n -Mode Product

There are two types of n -mode product [125]. The first type is the multiplication of a tensor by a matrix, denoted by $\times_n$, where the tensor order remains the same. The other type is the multiplication of a tensor by a vector, denoted by $\bar{\times}_n$, where the tensor order reduced by one. The second type, $\bar{\times}_n$, is being utilised in this thesis.

Let $X \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ be an arbitrary tensor, and $a \in \mathbb{R}^{I_\nu}$ be an arbitrary vector where $\nu = 1, 2, \dots, N$. The n -mode product of X with a is denoted by $X \bar{\times}_n a$ and has the size $I_1 \times \dots \times I_{\nu-1} \times I_{\nu+1} \times \dots \times I_N$. Basically, the dimension of tensor X that matches the size of vector a , that is dimension I_ν , is removed as an outcome of the n -mode product and it is defined element-wise as:

$$(X \bar{\times}_n a)_{i_1 \dots i_{\nu-1} i_{\nu+1} \dots i_N} = \sum_{i_\nu=1}^{I_\nu} x_{i_1 i_2 \dots i_N} a_{i_\nu}.$$

For example, let $X \in \mathbb{R}^{I \times J \times K}$, $a \in \mathbb{R}^L$, and $Y = X \bar{\times}_n a$ where X is the following $3 \times 4 \times 2$ tensor:

$$X_{::1} = \begin{bmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{bmatrix}, X_{::2} = \begin{bmatrix} 13 & 16 & 19 & 22 \\ 14 & 17 & 20 & 23 \\ 15 & 18 & 21 & 24 \end{bmatrix},$$

and $a = [1 \ 2 \ 3 \ 4]^T$. Since the dimension L of vector a is matching dimension J (mode-2) of tensor X , the matrix Y is the mode-2 product of X with a and has the size 3×2 as follow:

$$Y = X \bar{\times}_2 a = \begin{bmatrix} 70 & 190 \\ 80 & 200 \\ 90 & 210 \end{bmatrix},$$

where Y has the size $I \times K$ and can be simply calculated as:

$$y_{ik} = \sum_{j=1}^J x_{ijk} \cdot a_j ,$$

for each $i = 1, \dots, I$ and $k = 1, \dots, K$. For instance, the first column of Y is calculated as:

$$Y_{:,1} = X_{:,1} a = \begin{bmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} = \begin{bmatrix} 70 \\ 80 \\ 90 \end{bmatrix} .$$

B.4 Gradient, Jacobian, and Hessian Basics

This thesis follows the *numerator layout convention* [314, 315], and the interested reader is encouraged to check these references for further details. The summary in this section is a collection of notes gathered from different sources [314, 315, 361, 363].

B.4.1 Scalar-Valued Function

For an arbitrary function $g(x) : \mathbb{R}^n \rightarrow \mathbb{R}$, the gradient element-wise definition is $\nabla g(x)_i = \frac{\partial g}{\partial x_i}$ which is a column vector as follow:

$$\nabla g(x) \triangleq \begin{bmatrix} \frac{\partial g(x)}{\partial x_1} \\ \frac{\partial g(x)}{\partial x_2} \\ \vdots \\ \frac{\partial g(x)}{\partial x_n} \end{bmatrix} \in \mathbb{R}^n .$$

The Jacobian is the transpose of the gradient, $\nabla g(x)^T$, and its element-wise definition is $J_g(x)_j = \frac{\partial g}{\partial x_j}$ which is a row vector as follow:

$$J_g(x) \triangleq \frac{\partial g(x)}{\partial x} = \begin{bmatrix} \frac{\partial g(x)}{\partial x_1} & \frac{\partial g(x)}{\partial x_2} & \dots & \frac{\partial g(x)}{\partial x_n} \end{bmatrix} \in \mathbb{R}^{1 \times n} .$$

The Hessian $H_g(x)$ is defined as the second-order gradient, $\nabla^2 g(x)$, or the transpose of the gradient derivative $\left(\frac{\partial \nabla g(x)}{\partial x} \right)^T$, and its element-wise definition is $H_g(x)_{ij} = \frac{\partial^2 g}{\partial x_i \partial x_j}$ which is a square matrix as follow:

$$H_g(x) \triangleq \begin{bmatrix} \frac{\partial^2 g(x)}{\partial x_1^2} & \frac{\partial^2 g(x)}{\partial x_1 \partial x_2} & \dots & \frac{\partial^2 g(x)}{\partial x_1 \partial x_n} \\ \frac{\partial^2 g(x)}{\partial x_2 \partial x_1} & \frac{\partial^2 g(x)}{\partial x_2^2} & \dots & \frac{\partial^2 g(x)}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 g(x)}{\partial x_n \partial x_1} & \frac{\partial^2 g(x)}{\partial x_n \partial x_2} & \dots & \frac{\partial^2 g(x)}{\partial x_n^2} \end{bmatrix} \in \mathbb{R}^{n \times n} . \quad (\text{B.1})$$

To understand this matrix, let us analyse the second derivative step by step. First, the Hessian $H_g(x)$ is the second-order gradient as mentioned above:

$$H_g(x) = \nabla^2 g(x) = \nabla(\nabla g(x)),$$

which means that the second derivative matrix is impractical since both the gradients are column vectors. The solution is to use the Jacobian for the second derivative, since its a row vector, then transpose the results. The element-wise definition becomes:

$$H_g(x)_{ij} = \left(\frac{\partial g}{\partial x_j \partial x_i} \right)^T = \frac{\partial g}{\partial x_i \partial x_j},$$

which yields the Hessian $H_g(x)$ as:

$$H_g(x) = \left(\frac{\partial \nabla g(x)}{\partial x} \right)^T = \begin{bmatrix} \left[\frac{\partial^2 g(x)}{\partial x_1^2} & \dots & \frac{\partial^2 g(x)}{\partial x_n \partial x_1} \right]^T \\ \vdots \\ \left[\frac{\partial^2 g(x)}{\partial x_1 \partial x_n} & \dots & \frac{\partial^2 g(x)}{\partial x_n^2} \right]^T \end{bmatrix},$$

which is similar to the result of (B.1) above.

Finally, the Hessian $H_g(x)$ matrix is symmetric if the derivatives are continuous, that is when $\frac{\partial g}{\partial x_i \partial x_j} = \frac{\partial g}{\partial x_j \partial x_i}$ to be specific.

B.4.2 Vector-Valued Function

For an arbitrary function $f(q) : \mathbb{R}^n \rightarrow \mathbb{R}^m$, the gradient element-wise definition is $\nabla f(q) = \frac{\partial f_j}{\partial q_i}$ which is an $n \times m$ matrix as follow:

$$\nabla f(q) \triangleq \begin{bmatrix} \frac{\partial f_1}{\partial q_1} & \dots & \frac{\partial f_m}{\partial q_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_1}{\partial q_n} & \dots & \frac{\partial f_m}{\partial q_n} \end{bmatrix} \in \mathbb{R}^{n \times m}.$$

As mentioned earlier in Section B.4.1, the Jacobian is the transpose of the gradient, $\nabla f(q)^T$, and its element-wise definition is $J_f(q) = \frac{\partial f_i}{\partial q_j}$ which is an $m \times n$ matrix as follow:

$$J_f(q) \triangleq \begin{bmatrix} (\nabla f_1)^T \\ \vdots \\ (\nabla f_m)^T \end{bmatrix} = \begin{bmatrix} \frac{\partial f(q)}{\partial q_1} & \dots & \frac{\partial f(q)}{\partial q_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial q_1} & \dots & \frac{\partial f_1}{\partial q_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial q_1} & \dots & \frac{\partial f_m}{\partial q_n} \end{bmatrix} \in \mathbb{R}^{m \times n}.$$

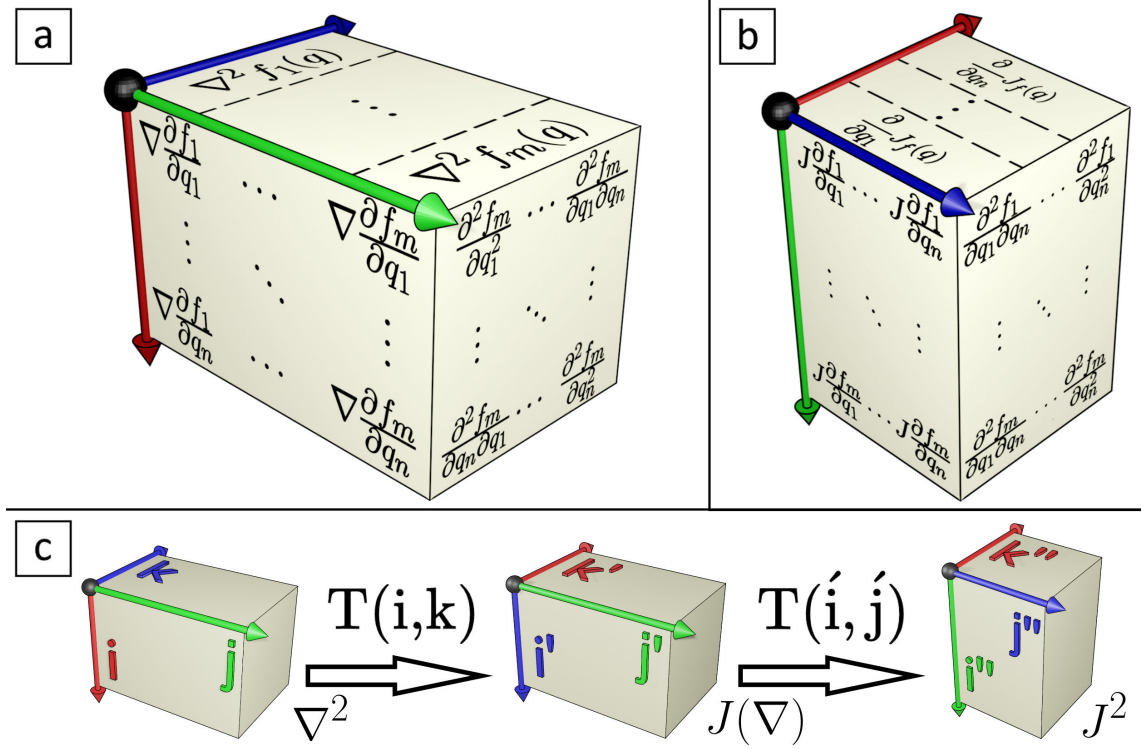


Figure B.2: The second-order partial derivatives of $f(q)$ represented as a third-order tensor where (a) shows the visualisation of The Hessian $H_f(q)$ tensor. The red axis represents the fibres direction of the rows $H_{:jk}$, the green axis is for the columns $H_{i:k}$, and the blue axis is for the tubes $H_{ij:}$. The second-order Jacobian $J_f^2(q)$ is the double transpose of the Hessian $H_f(q)$ tensor where (b) shows the visualisation of $J_f^2(q)$ tensor, and (c) shows the double transpose sequence.

Similarly, The Hessian $H_f(q)$ is the second-order gradient, $\nabla^2 f(q)$, or the transpose of the gradient derivative, $\left(\frac{\partial \nabla f(q)}{\partial q}\right)^T$. The result is a third-order tensor and arranged as an array of m Hessian matrices as follow:

$$H_f(q) = [H_{f_1}(q) \quad H_{f_2}(q) \quad \cdots \quad H_{f_m}(q)] \in \mathbb{R}^{n \times m \times n}.$$

Figure B.2(a) shows a visualisation of the Hessian $H_f(q)$ tensor where the element-wise definition is:

$$H_f(q)_{ijk} = \frac{\partial^2 f_j}{\partial q_i \partial q_k}. \quad (\text{B.2})$$

This analysis in thesis requires deriving another partial derivative tensor which is the second-order Jacobian, denoted as J^2 , or the Jacobian of the Jacobian, as explained earlier in Chapter 3, Section 3.3.3. This tensor is arranged as an array of n Jacobian derivatives as follow:

$$J_f^2(q) = \left[\frac{\partial J_f(q)}{\partial q_1} \quad \frac{\partial J_f(q)}{\partial q_2} \quad \cdots \quad \frac{\partial J_f(q)}{\partial q_n} \right] \in \mathbb{R}^{m \times n \times n},$$

where the first and second dimensions, $m \times n \times \dots$, are for the Jacobian $J_f(q)$, and the third dimension, $\dots \times \dots \times n$, is for each partial derivative of the Jacobian as shown in Figure B.2(b). The $J_f^2(q)$ tensor has the following element-wise definition:

$$J_f^2(q)_{ijk} = \frac{\partial^2 f_i}{\partial q_k \partial q_j}. \quad (\text{B.3})$$

By comparing (B.2) and (B.3), the required transpose sequence to generate one tensor from the other can be determined. As an example, to generate $J_f^2(q)$ tensor from $H_f(q)$ tensor, two transpose steps are required since $H_f(q) = \nabla^2 f(q)$. The first transpose is for the outer gradient, and the second transpose is for the inner gradient as follow:

$$\underbrace{\frac{\partial^2 f_j}{\partial q_i \partial q_k}}_{\nabla^2} \xrightarrow{\text{T}(i,k)} \underbrace{\frac{\partial^2 f_{j'}}{\partial q_{i'} \partial q_{k'}}}_{J(\nabla)} \xrightarrow{\text{T}(i',j')} \underbrace{\frac{\partial^2 f_{i''}}{\partial q_{k''} \partial q_{j''}}}_{J^2},$$

and as shown in Figure B.2(c). This double transpose can be performed in MATLAB using the command `JJ = permute(H, [2 3 1])`. However, The first transpose, $\text{T}(i, k)$, is unnecessary if the Hessian $H_f(q)$ is symmetric.

B.5 Derivation of J_{sv}

Starting with the translation vector $\tilde{t}$, defined earlier in (3.3), re-writing to:

$$\tilde{t} = \frac{L_s}{\theta_s} \begin{bmatrix} c_{\phi_s}(1 - c_{\theta_s}) \\ s_{\phi_s}(1 - c_{\theta_s}) \\ s_{\theta_s} \end{bmatrix} = \begin{bmatrix} \tilde{t}1 \\ \tilde{t}2 \\ \tilde{t}3 \end{bmatrix}. \quad (\text{B.4})$$

Using the identities described in section B.1, the time derivative of the first row $\tilde{t}1$ is as follow:

$$\begin{aligned} \dot{\tilde{t}1} &= \frac{\dot{L}_s \theta_s - L_s \dot{\theta}_s}{\theta_s^2} c_{\phi_s}(1 - c_{\theta_s}) - \frac{L_s}{\theta_s} s_{\phi_s}(1 - c_{\theta_s}) \dot{\phi}_s + \frac{L_s}{\theta_s} c_{\phi_s} s_{\theta_s} \dot{\theta}_s \\ &= \frac{s_{\phi_s}(c_{\theta_s} - 1)L_s}{\theta_s} \dot{\phi}_s + \frac{L_s}{\theta_s} c_{\phi_s} s_{\theta_s} \dot{\theta}_s - \frac{L_s}{\theta_s^2} c_{\phi_s}(1 - c_{\theta_s}) \dot{\theta}_s + \frac{c_{\phi_s}(1 - c_{\theta_s})}{\theta_s} \dot{L}_s \\ &= \begin{bmatrix} \frac{s_{\phi_s}(c_{\theta_s} - 1)L_s}{\theta_s} & \frac{c_{\phi_s}(c_{\theta_s} + \theta_s s_{\theta_s} - 1)L_s}{\theta_s^2} & \frac{c_{\phi_s}(1 - c_{\theta_s})}{\theta_s} \end{bmatrix} \begin{bmatrix} \dot{\phi}_s \\ \dot{\theta}_s \\ \dot{L}_s \end{bmatrix}. \end{aligned} \quad (\text{B.5})$$

Similarly, for the second row $\tilde{t}2$, the time derivative is:

$$\begin{aligned}
\dot{\tilde{t}}2 &= \frac{\dot{L}_s \theta_s - L_s \dot{\theta}_s}{\theta_s^2} s_{\phi_s} (1 - c_{\theta_s}) + \frac{L_s}{\theta_s} c_{\phi_s} (1 - c_{\theta_s}) \dot{\phi}_s + \frac{L_s}{\theta_s} s_{\phi_s} s_{\theta_s} \dot{\theta}_s \\
&= \frac{c_{\phi_s} (1 - c_{\theta_s}) L_s}{\theta_s} \dot{\phi}_s + \frac{L_s}{\theta_s} s_{\phi_s} s_{\theta_s} \dot{\theta}_s - \frac{L_s}{\theta_s^2} s_{\phi_s} (1 - c_{\theta_s}) \dot{\theta}_s + \frac{s_{\phi_s} (1 - c_{\theta_s})}{\theta_s} \dot{L}_s \\
&= \begin{bmatrix} \frac{c_{\phi_s} (1 - c_{\theta_s}) L_s}{\theta_s} & \frac{s_{\phi_s} (c_{\theta_s} + \theta_s s_{\theta_s} - 1) L_s}{\theta_s^2} & \frac{s_{\phi_s} (1 - c_{\theta_s})}{\theta_s} \end{bmatrix} \begin{bmatrix} \dot{\phi}_s \\ \dot{\theta}_s \\ \dot{L}_s \end{bmatrix}.
\end{aligned} \tag{B.6}$$

Finally, for the third row $\tilde{t}3$, the time derivative is:

$$\begin{aligned}
\dot{\tilde{t}}3 &= \frac{\dot{L}_s \theta_s - L_s \dot{\theta}_s}{\theta_s^2} s_{\theta_s} + \frac{L_s}{\theta_s} c_{\theta_s} \dot{\theta}_s \\
&= \frac{L_s}{\theta_s} c_{\theta_s} \dot{\theta}_s - \frac{L_s}{\theta_s^2} s_{\theta_s} \dot{\theta}_s + \frac{s_{\theta_s}}{\theta_s} \dot{L}_s \\
&= \begin{bmatrix} 0 & \frac{(\theta_s c_{\theta_s} - s_{\theta_s}) L_s}{\theta_s^2} & \frac{s_{\theta_s}}{\theta_s} \end{bmatrix} \begin{bmatrix} \dot{\phi}_s \\ \dot{\theta}_s \\ \dot{L}_s \end{bmatrix}.
\end{aligned} \tag{B.7}$$

Substituting (B.5), (B.6), and (B.7) in (3.46) gives the translational part $J_{sv}(q_s)$ of the coordinate Jacobian as:

$$J_{sv}(q_s) = \begin{bmatrix} \frac{s_{\phi_s} (c_{\theta_s} - 1) L_s}{\theta_s} & \frac{c_{\phi_s} (c_{\theta_s} + \theta_s s_{\theta_s} - 1) L_s}{\theta_s^2} & \frac{c_{\phi_s} (1 - c_{\theta_s})}{\theta_s} \\ \frac{c_{\phi_s} (1 - c_{\theta_s}) L_s}{\theta_s} & \frac{s_{\phi_s} (c_{\theta_s} + \theta_s s_{\theta_s} - 1) L_s}{\theta_s^2} & \frac{s_{\phi_s} (1 - c_{\theta_s})}{\theta_s} \\ 0 & \frac{(\theta_s c_{\theta_s} - s_{\theta_s}) L_s}{\theta_s^2} & \frac{s_{\theta_s}}{\theta_s} \end{bmatrix}. \tag{B.8}$$

B.6 Derivation of J_{sw}

Starting from (3.51):

$$\omega_s = \vee_{\times} (J_{sR1} R^T) \dot{q}_{s1} + \cdots + \vee_{\times} (J_{sRn} R^T) \dot{q}_{sn}.$$

The Rotation matrix transpose R^T is:

$$R^T = \begin{bmatrix} c_{\phi_s} c_{\theta_s} & s_{\phi_s} c_{\theta_s} & -s_{\theta_s} \\ -s_{\phi_s} & c_{\phi_s} & 0 \\ c_{\phi_s} s_{\theta_s} & s_{\phi_s} s_{\theta_s} & c_{\theta_s} \end{bmatrix}.$$

Each segment of the soft actuator has $n = 3$. For simplicity, arbitrary matrices A , B , and C were used to derive the time derivative Jacobians J_{sRi} product with the rotation matrix transpose R^T where:

$$A = \frac{\partial R}{\partial \phi_s} R^T = \begin{bmatrix} A11 & A12 & A13 \\ A21 & A22 & A23 \\ A31 & A32 & A33 \end{bmatrix},$$

$$B = \frac{\partial R}{\partial \theta_s} R^T = \begin{bmatrix} B11 & B12 & B13 \\ B21 & B22 & B23 \\ B31 & B32 & B33 \end{bmatrix},$$

$$C = \frac{\partial R}{\partial L_s} R^T = [0]_{3 \times 3}.$$

Since the rotation matrix R is a function of ϕ_s and θ_s only, the matrix $C = [0]_{3 \times 3}$ obviously. The time derivative Jacobians are:

$$\frac{\partial R}{\partial \phi_s} = \begin{bmatrix} -s_{\phi_s} c_{\theta_s} & -c_{\phi_s} & -s_{\phi_s} s_{\theta_s} \\ c_{\phi_s} c_{\theta_s} & -s_{\phi_s} & c_{\phi_s} s_{\theta_s} \\ 0 & 0 & 0 \end{bmatrix} \dot{\phi}_s,$$

$$\frac{\partial R}{\partial \theta_s} = \begin{bmatrix} -c_{\phi_s} s_{\theta_s} & 0 & c_{\phi_s} c_{\theta_s} \\ -s_{\phi_s} s_{\theta_s} & 0 & s_{\phi_s} c_{\theta_s} \\ -c_{\theta_s} & 0 & -s_{\theta_s} \end{bmatrix} \dot{\theta}_s.$$

Now, performing the product of $\frac{\partial R}{\partial \phi_s}$ and R^T gives the elements of matrix A as follow:

$$\begin{aligned}
A11 &= (-s_{\phi_s} c_{\theta_s})(c_{\phi_s} c_{\theta_s}) + (-c_{\phi_s})(-s_{\phi_s}) + (-s_{\phi_s} s_{\theta_s})(c_{\phi_s} s_{\theta_s}) \\
&= s_{\phi_s} c_{\phi_s} (1 - s_{\theta_s}^2 - c_{\theta_s}^2) = 0, \\
A12 &= (-s_{\phi_s} c_{\theta_s})(s_{\phi_s} c_{\theta_s}) + (-c_{\phi_s})(c_{\phi_s}) + (-s_{\phi_s} s_{\theta_s})(s_{\phi_s} s_{\theta_s}) \\
&= -s_{\phi_s}^2 (c_{\theta_s}^2 + s_{\theta_s}^2) - c_{\phi_s}^2 = -1, \\
A13 &= (-s_{\phi_s} c_{\theta_s})(-s_{\theta_s}) + (-c_{\phi_s})(0) + (-s_{\phi_s} s_{\theta_s})(c_{\theta_s}) \\
&= s_{\phi_s} s_{\theta_s} c_{\theta_s} - s_{\phi_s} s_{\theta_s} c_{\theta_s} = 0, \\
A21 &= (c_{\phi_s} c_{\theta_s})(c_{\phi_s} c_{\theta_s}) + (-s_{\phi_s})(-s_{\phi_s}) + (c_{\phi_s} s_{\theta_s})(c_{\phi_s} s_{\theta_s}) \\
&= c_{\phi_s}^2 (c_{\theta_s}^2 + s_{\theta_s}^2) + s_{\phi_s}^2 = 1, \\
A22 &= (c_{\phi_s} c_{\theta_s})(s_{\phi_s} c_{\theta_s}) + (-s_{\phi_s})(c_{\phi_s}) + (c_{\phi_s} s_{\theta_s})(s_{\phi_s} s_{\theta_s}) \\
&= s_{\phi_s} c_{\phi_s} (c_{\theta_s}^2 - 1 + s_{\theta_s}^2) = 0, \\
A23 &= (c_{\phi_s} c_{\theta_s})(-s_{\theta_s}) + (-s_{\phi_s})(0) + (c_{\phi_s} s_{\theta_s})(c_{\theta_s}) \\
&= -c_{\phi_s} s_{\theta_s} c_{\theta_s} + c_{\phi_s} s_{\theta_s} c_{\theta_s} = 0, \\
A31 &= A32 = A33 = 0,
\end{aligned}$$

which yields matrix A as:

$$A = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

Similarly, performing the product of $\frac{\partial R}{\partial \theta_s}$ and R^T gives the elements of matrix B as follow:

$$\begin{aligned}
B11 &= (-c_{\phi_s} s_{\theta_s})(c_{\phi_s} c_{\theta_s}) + (0)(-s_{\phi_s}) + (c_{\phi_s} c_{\theta_s})(c_{\phi_s} s_{\theta_s}) \\
&= -c_{\phi_s}^2 s_{\theta_s} c_{\theta_s} + c_{\phi_s}^2 s_{\theta_s} c_{\theta_s} = 0, \\
B12 &= (-c_{\phi_s} s_{\theta_s})(s_{\phi_s} c_{\theta_s}) + (0)(c_{\phi_s}) + (c_{\phi_s} c_{\theta_s})(s_{\phi_s} s_{\theta_s}) \\
&= -s_{\phi_s} c_{\phi_s} s_{\theta_s} c_{\theta_s} + s_{\phi_s} c_{\phi_s} s_{\theta_s} c_{\theta_s} = 0, \\
B13 &= (-c_{\phi_s} s_{\theta_s})(-s_{\theta_s}) + (0)(0) + (c_{\phi_s} c_{\theta_s})(c_{\theta_s}) \\
&= c_{\phi_s} (s_{\theta_s}^2 + c_{\theta_s}^2) = c_{\phi_s},
\end{aligned}$$

$$\begin{aligned}
B21 &= (-s_{\phi_s} s_{\theta_s})(c_{\phi_s} c_{\theta_s}) + (0)(-s_{\phi_s}) + (s_{\phi_s} c_{\theta_s})(c_{\phi_s} s_{\theta_s}) \\
&= -s_{\phi_s} c_{\phi_s} s_{\theta_s} c_{\theta_s} + s_{\phi_s} c_{\phi_s} s_{\theta_s} c_{\theta_s} = 0, \\
B22 &= (-s_{\phi_s} s_{\theta_s})(s_{\phi_s} c_{\theta_s}) + (0)(c_{\phi_s}) + (s_{\phi_s} c_{\theta_s})(s_{\phi_s} s_{\theta_s}) \\
&= -s_{\phi_s}^2 s_{\theta_s} c_{\theta_s} + s_{\phi_s}^2 s_{\theta_s} c_{\theta_s} = 0, \\
B23 &= (-s_{\phi_s} s_{\theta_s})(-s_{\phi_s}) + (0)(0) + (s_{\phi_s} c_{\theta_s})(c_{\theta_s}) \\
&= s_{\phi_s}(s_{\theta_s}^2 + c_{\theta_s}^2) = s_{\phi_s}, \\
B31 &= (-c_{\theta_s})(c_{\phi_s} c_{\theta_s}) + (0)(-s_{\phi_s}) + (-s_{\theta_s})(c_{\phi_s} s_{\theta_s}) \\
&= -c_{\phi_s}(c_{\theta_s}^2 + s_{\theta_s}^2) = -c_{\phi_s}, \\
B32 &= (-c_{\theta_s})(s_{\phi_s} c_{\theta_s}) + (0)(c_{\phi_s}) + (-s_{\theta_s})(s_{\phi_s} s_{\theta_s}) \\
&= -s_{\phi_s}(c_{\theta_s}^2 + s_{\theta_s}^2) = -s_{\phi_s}, \\
B33 &= (-c_{\theta_s})(-s_{\theta_s}) + (0)(0) + (-s_{\theta_s})(c_{\theta_s}) = 0,
\end{aligned}$$

which yields matrix B as:

$$B = \begin{bmatrix} 0 & 0 & c_{\phi_s} \\ 0 & 0 & s_{\phi_s} \\ -c_{\phi_s} & -s_{\phi_s} & 0 \end{bmatrix}.$$

Now, applying inverse skew-symmetric $\mathbb{V}_{\times}()$ operator to A , B , and C matrices yields:

$$a = \mathbb{V}_{\times}(A) = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad b = \mathbb{V}_{\times}(B) = \begin{bmatrix} -s_{\phi_s} \\ c_{\phi_s} \\ 0 \end{bmatrix}, \quad c = \mathbb{V}_{\times}(C) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix},$$

where $a, b, c \in \mathbb{R}^3$ are arbitrary vectors. Substituting in (3.51) leads to:

$$\omega_s = \begin{bmatrix} a & b & c \end{bmatrix} \begin{bmatrix} \dot{\phi}_s \\ \dot{\theta}_s \\ \dot{L}_s \end{bmatrix} = J_{s\omega}(q_s) \begin{bmatrix} \dot{\phi}_s \\ \dot{\theta}_s \\ \dot{L}_s \end{bmatrix},$$

in which the rotational part $J_{s\omega}(q_s)$ of the coordinates Jacobian matrix $J_s(q_s)$ is:

$$J_{s\omega}(q_s) = \begin{bmatrix} 0 & -s\phi_s & 0 \\ 0 & c\phi_s & 0 \\ 1 & 0 & 0 \end{bmatrix}. \quad (\text{B.9})$$

Combining both the translational and rotational Jacobian parts gives the full coordinate Jacobian matrix $J_s(q_s)$ of the individual segments as:

$$J_s(q_s) = \begin{bmatrix} \frac{s\phi_s(c\theta_s-1)L_s}{\theta_s} & \frac{c\phi_s(c\theta_s+\theta_s s\theta_s-1)L_s}{\theta_s^2} & \frac{c\phi_s(1-c\theta_s)}{\theta_s} \\ \frac{c\phi_s(1-c\theta_s)L_s}{\theta_s} & \frac{s\phi_s(c\theta_s+\theta_s s\theta_s-1)L_s}{\theta_s^2} & \frac{s\phi_s(1-c\theta_s)}{\theta_s} \\ 0 & \frac{(\theta_s c\theta_s - s\theta_s)L_s}{\theta_s^2} & \frac{s\theta_s}{\theta_s} \\ 0 & -s\phi_s & 0 \\ 0 & c\phi_s & 0 \\ 1 & 0 & 0 \end{bmatrix}. \quad (\text{B.10})$$

Appendix C

Hessian Tensor Verification Code (MATLAB)

```
% This code is to verify the derivation of the Hessian tensor via
% deriving the first and second time derivatives of the translation
% vector \tilde{t} and rotation matrix R of the homogeneous transformation
% matrix T.
%
% Written by Abdullah Al-Azzawi
% Dated 13.03.2023
% This code can be downloaded from https://github.com/abd-alaz/PhD-thesis

clc; clear;

% Define the variables of Eq(B.4) in Appendix B:

syms t                % time
syms p(t);           % phi, the rotation angle
syms h(t);           % theta, the bending angle
syms L(t);           % length of actuator segment

% Create a column vector for the three variables using vertical concatenate.
% The vector q is a function of time t and can be differentiate w.r.t time.
% Size of q is R^n where n=3
q = [p(t);h(t);L(t)];
assume(q,'positive') % all values of q are positive
% q = cat(1,p(t),h(t),L(t)); % another approach to generate q.

% Calculate the first and second time derivatives of q:
dq = diff(q);
ddq = diff(q,2);

t1 = L * (cos(p)*(1 - cos(h))) / h;
t2 = L * (sin(p)*(1 - cos(h))) / h;
t3 = L * sin(h) / h;

% Create the translation vector tv by concatenating vertically,
% size of tv is R^m where m=3 :
tv = [t1(t);t2(t);t3(t)];
% tv = cat(1,t1(t),t2(t),t3(t)); % another approach to generate tv.

% Calculate the second time derivate of tv
% it will be used later in the verification
ddtv = diff(tv,t,2);

% Now, Hessian of tv = grad(grad tv(q))
% the element-wise entry in Latex format is
% Htv(i,j,k) = \frac{\partial^2 tv_j}{\partial q_i \partial q_k}
% the inner partial is for q_k, and the outer partial is for q_i
```

```

n = size(q,1);      % size of the column vector q
m = size(tv,1);     % size of the column vector tv
Htv = sym(zeros(n, m, n));

for i = 1:n
    for j = 1:m
        for k = 1:n
            Htv(i,j,k) = functionalDerivative(functionalDerivative(tv(j), q(k)), q(i));
        end
    end
end

% Display the Hessian tensor as a sequence of Hessian matrices,
% MATLAB, by default, displays the (... , ... , k) slides but
% we need to display (... , j , ... ) slides which can be done by using
% permute and switch j & k as follow:
H1 = permute(Htv(:,1,:),[1 3 2]); % Hessian of f1
H2 = permute(Htv(:,2,:),[1 3 2]); % Hessian of f2
H3 = permute(Htv(:,3,:),[1 3 2]); % Hessian of f3

% Calculating the Jacobian of tv can be done either with element-wise (as above)
% or using concatenation of derivative vectors as per the following,
% using jacobian(tv) will generate a matrix that includes diff(q) in it
% as it is a derivative w.r.t time.
% instead, we need the derivative of vector tv w.r.t the vector q.
% Using the element-wise transpose ensures that the outcome is a row vector.
J1 = functionalDerivative(tv(1),q).';
J2 = functionalDerivative(tv(2),q).';
J3 = functionalDerivative(tv(3),q).';

% Concatenating vertically to create the Jacobian matrix, however
% using (t) will generate error as MATLAB will mix between the function of
% time parentheses versus index parentheses. So, it is simply:
Jtv = [J1;J2;J3];

% Confirming the Hessian tensor via the second time derivative equation
% ddtv = dJtv * dq + Jtv * ddq
% the term (dJtv) is a matrix calculated via n-mode product of Htv (which is
% a 3rd-order tensor) and the vector dq as below.
% assume A = ddtv - Jtv * ddq
% assume B = dJtv * dq
% the idea is to confirm that A = B
A = ddtv - Jtv * ddq;

% Now, for the second part (i.e. B)
% dJtv is the time derivative of Jtv. So, applying the chain rule gives:
% dJtv = JJtv (x_n) dq
% where (x_n) is the n-mode product, and JJtv is the partial derivative
% of Jtv w.r.t vector q, or the "Jacobian of the Jacobian" of tv(q).
%
% JJtv is a 3rd-order tensor and can be calculated via double transpose of Htv,
% first transpose is between dim1 and dim3 (outer gradient), while
% second transpose is between dim1 and dim2 (inner gradient) as follow:
% JJtv = permute(permute(Htv,[3 2 1]),[2 1 3]) = permute(Htv,[2 3 1])

% Another method to calculate JJtv is to use element-wise derivative,
% the element-wise entry (in Latex format) is:
% JJtv(i,j,k) = \frac{\partial^2 tv_i}{\partial q_k \partial q_j}
%
% the Jacobian = \frac{\partial tv_i}{\partial q_j} which is the inner
% partial, while the outer partial is for q_k.

JJtv = sym(zeros(m,n,n));
for i = 1:m
    for j = 1:n
        for k = 1:n
            JJtv(i,j,k) = functionalDerivative(Jtv(i,j), q(k));
        end
    end
end

```

```

% Also, to confirm that both methods (double transpose and element-wise) produce
% the same result ...
% JJtv - permute(permute(Htv,[3 2 1]),[2 1 3]) = [0]_{m x n x m}

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Now, the n-mode product ...

% The use of tensor n-mode product (tensor x vector) can be achieved via the
% library mytensorprod.m downloaded from the link
% https://au.mathworks.com/matlabcentral/fileexchange/116860-mytensorprod
% but it requires MATLAB 2020b or later.
% Also, the built-in MATLAB function 'tensorprod' requires MATLAB 2022a

% Let us do n-mode product (mode 3) in simple math steps
% that is the dim-3 of the tensor multiplied by the vector dq,
% the outcome is a matrix (or 2D tensor).
% This product can be expressed in element-wise entry as follow (in latex format):
%  $y_{ij} = \sum_{k=1}^K x_{ijk} * v_k$ 

% Initialising a matrix to hold the output of n-mode product ...
dJtv = sym(zeros(m,n));

k = 1:n;      % for mode-3 product
for i = 1:m    % for dim-1
    for j = 1:n % for dim-2
        aa = JJtv(i,j,k); % temporary variable, aa is still a 3D tensor

        % convert from a tensor to a vector, otherwise the product will
        % give error when multiplying a tensor by a vector directly
        % the function 'length' returns the largest dimension, while the
        % function 'size' returns two numbers.
        aa = reshape(aa,[length(k),1]);

        % finally, the n-mode product
        dJtv(i,j) = sum(aa .* dq);
    end
end

% Another solution for n-mode product is achieved by simply using the
% derivative of Jtv directly as follow:
%
% k = 1:n;      % for mode-3 product
% for i = 1:m
%     for j = 1:n
%         % the element-wise multiplication ensures every row of the vector
%         % is multiplied by the corresponding dq row
%         % (it is not a matrix multiplication)
%         dJtv(i,j) = sum(functionalDerivative(Jtv(i,j),q(k)) .* dq(k));
%     end
% end

% Anyway ...
%
% Calculate the other side of the second time derivative equation:
B = dJtv * dq;

% Finally, C should be [0;0;0] if the equations above are correct
% using 'simplify' is important to achieve the final answer
C = simplify(A - B);

```


Appendix D

Energy Modelling

As an alternative approach to length modelling, Section 4.1.2, geodesic theory can be applied to model the kinetic energy E_k of a massless particle moving along the sensor path. Thus, E_k can be interpreted as a specific kinetic energy [364] and computed as indicated by previous studies [149, 365]:

$$\begin{aligned} E_{ki} &= \frac{1}{2} \int_0^1 \|\dot{\gamma}_i(s)\|^2 dt \\ &= \frac{1}{2} \int_0^1 \dot{v}^2 R^2 + \dot{u}^2 (c - R \cos(v - \phi))^2 dt. \end{aligned} \quad (\text{D.1})$$

By substituting (4.17) into (D.1) and utilising the actuator curvature, i.e. $c = \frac{1}{\kappa}$, the expression becomes:

$$E_{ki} = \frac{1}{2} \int_0^1 R^2 (\Delta v_i + \psi)^2 + h^2 \left(1 - R\kappa \cos(v_{si} + t(\Delta v_i + \psi) - \phi) \right)^2 dt. \quad (\text{D.2})$$

This integral can be evaluated analytically. Solving and simplifying (D.2) yields the energy equation:

$$\begin{aligned} E_{ki} &= \frac{h^2}{2} \left[1 + \frac{\kappa^2 R^2}{2} \left(1 + \frac{\sin 2\alpha_1 + \sin 2\alpha_2}{2(\Delta v_i + \psi)} \right) - \frac{2\kappa R (\sin \alpha_1 + \sin \alpha_2)}{\Delta v_i + \psi} \right] \\ &\quad + \frac{R^2}{2} (\alpha_1 - \alpha_2)^2, \end{aligned} \quad (\text{D.3})$$

where $\alpha_1 = \Delta v_i + \psi - \phi + v_{si}$ and $\alpha_2 = \phi - v_{si}$.

Unlike the sensor length l_i , defined in (4.18), which requires numerical integration for evaluation due to its integral form, the energy E_{ki} is derived through direct mathematical operations, making its evaluation a simpler and more direct process. Nevertheless, No further analysis will be provided as it is beyond the scope of this thesis. Interested readers are encouraged to refer to the relevant literature cited above.