MSc Control and Robotics
Academic year 2023 - 2024

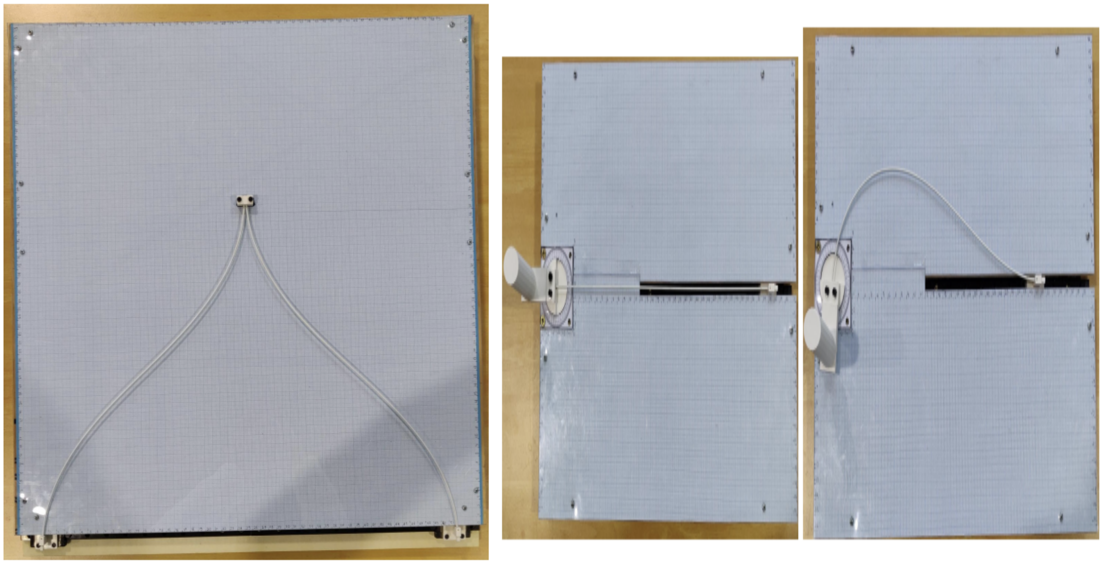
SOROMO Lab-3 : Constrained Elastic Systems

Raffaele Pumpo
Emanuele Buzzurro
Under the supervision of Andrea Gotelli

Contents

1 Introduction

In the third lab of the SOROMO course, we have analysed the constrained elastic systems. In this session, our focus centers on two distinct prototypes, each showcasing the intricate dynamics of soft robotics: the elastic slider-crank, featuring a rod between a revolute and a prismatic joint, and a continuum parallel robot shown in the figure below:



(a) Continuum Parallel robot

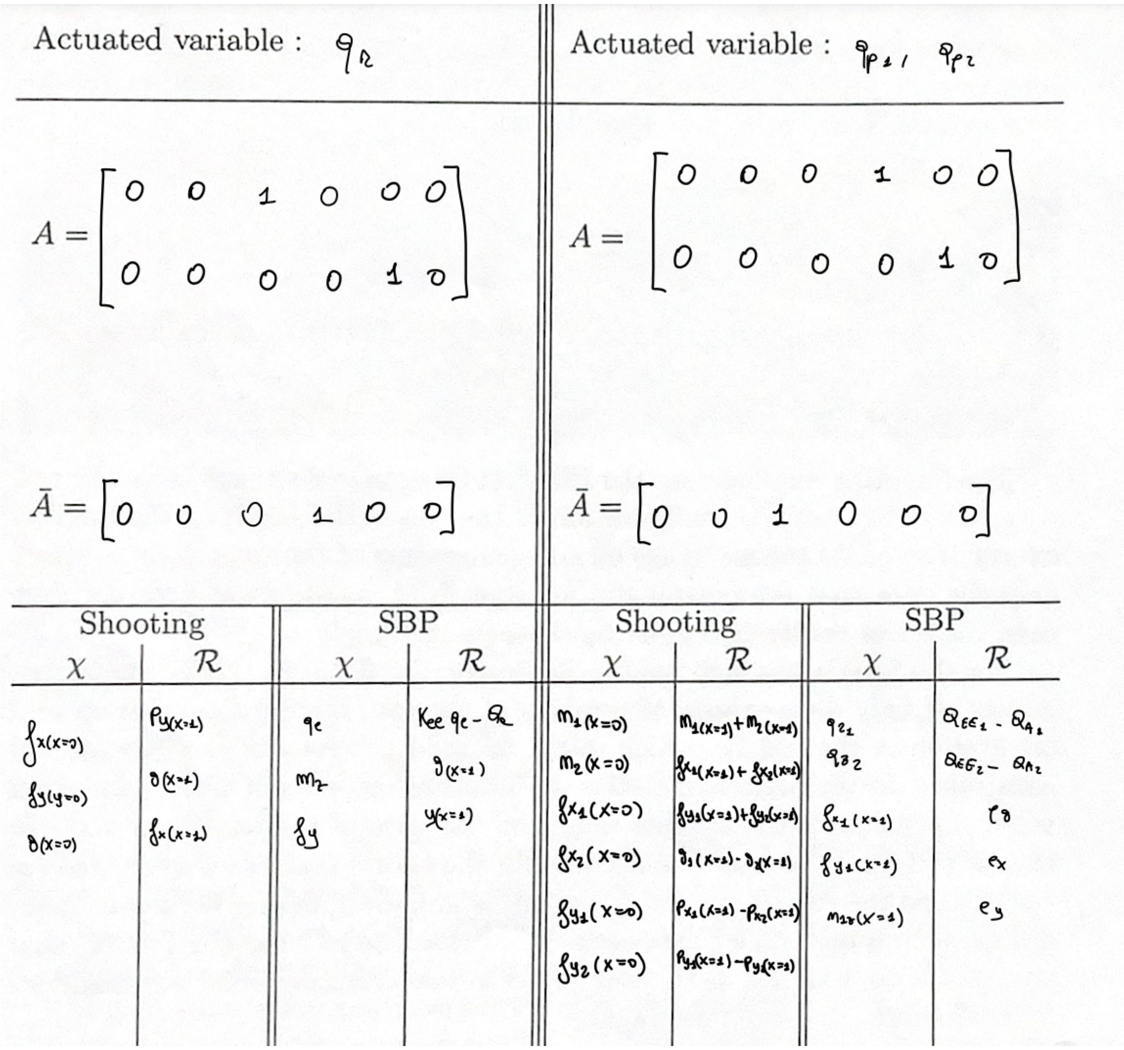
(b) elastic slider-crank

Building upon the knowledge gained in the previous lab, where we improved our knowledge regarding the modelization of Cosserat rods, using the Shooting method and Strain-Based Parameterization (SBP), the current lab elevates our exploration to the static modeling of parallel architectures incorporating elastic components. Just as in the real-world scenario where closing your hands forms a parallel architecture with your arms, these systems exhibit closed kinematic chains.

The initial section of the lab aims to provide insights into the methodology for modeling such intricate systems, offering a foundation for our subsequent case studies; in Section 2 we developed the numerical simulations for the two systems; While in Section 3, we delved into the specifics of the physical prototypes employed in the lab, by taking measurements of the behavior of teh real ones, and then comparing them with our computed model.

2 HomeWorks

The initial step in our lab exploration, we started with a preliminary assignment focusing on the completion of a table for two distinctive configurations: the elastic slider-crank and the continuum parallel robot. The objective was to provide a comprehensive overview of the matrices A and $\bar{A}$ associated with each configuration.



Actuated variable : q_R

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\bar{A} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Shooting		SBP	
$\mathcal{X}$	$\mathcal{R}$	$\mathcal{X}$	$\mathcal{R}$
$f_{x(x=0)}$	$P_{y(x=1)}$	q_e	$K_{ee}\,q_e - Q_R$
$f_{y(y=0)}$	$\theta(x=1)$	m_z	$\theta_{(x=1)}$
$\theta(x=0)$	$f_x(x=1)$	f_y	$y(x=1)$

Actuated variable : q_{p1}, q_{p2}

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\bar{A} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

Shooting		SBP	
$\mathcal{X}$	$\mathcal{R}$	$\mathcal{X}$	$\mathcal{R}$
$m_1(x=0)$	$m_{1(x=1)} + m_2(x=1)$	q_{e_1}	$Q_{EE_1} - Q_{A_1}$
$m_2(x=0)$	$f_{x_1}(x=1) + f_{x_2}(x=1)$	q_{e_2}	$Q_{EE_2} - Q_{A_2}$
$f_{x_1}(x=0)$	$f_{y_1}(x=1) + f_{y_2}(x=1)$	$f_{x_1}(x=1)$	e_θ
$f_{x_2}(x=0)$	$\theta_1(x=1) - \theta_2(x=1)$	$f_{y_1}(x=1)$	e_x
$f_{y_1}(x=0)$	$P_{x_1}(x=1) - P_{x_2}(x=1)$	$m_{1z}(x=1)$	e_y
$f_{y_2}(x=0)$	$P_{y_1}(x=1) - P_{y_2}(x=1)$		

Figure 2: Homework table

To comprehend the significance of these constraints, matrices A and $\bar{A}$ derive from the identity matrix, which encapsulates the total set of Degrees of Freedom (DoFs) of a joint, in our **planar case only 3 DoFs**. Matrix A includes all rows except the one corresponding to the joint DoF(s), while the complementary matrix $\bar{A}$ expresses the excluded row. Notably, the relationship between these matrices is defined by $A\bar{A} = 1$. For a fixed joint, A becomes the identity matrix, and $\bar{A}$ is an empty matrix ([]), indicating no additional constraints.

In our specific case, regarding the Elastic Slider-Crank:

1. A: The DoFs included are the rotation along z and the translation along y
2. $\bar{A}$: Means that there are no constraints on the translation along x

Regarding the Continuum Parallel robot:

1. A: The DoFs included are the translation along y and x
2. $\bar{A}$: Means that there are no constraints on the rotation around z

The second part to fill regarded the values of χ and residuals. In the context of deformable bodies, χ encompasses essential parameters for the geometric static modeling of the rod. For the Shooting method, these parameters include the wrench W(X=0), while for the Strain-Based Parameterization (SBP), the parameter q_e is considered. Additionally, χ accounts for the efforts applied to the rod by the cut joint.

In the Shooting method, W(X=0) represents the wrench exerted on the rod the base. For SBP, q_e characterizes the parameters essential for the strain-based representation of the rod, capturing its deformation characteristics and contributing to the overall static modeling.

Then we have Residuals, which are the differences between the predicted and actual values in a mathematical model, and will be explained in detail later. In this context, residuals signify the disparities between the expected behavior of the rod under specified parameters and the actual observations or measurements. They serve as indicators of how well the model aligns with the physical reality. Minimizing residuals is a key objective, as it signifies a closer match between the theoretical model and the practical behavior of the system.

3 Numerical Simulation

In this section, we outline the expected tasks for both the Shooting method and Strain-Based Parameterization (SBP) in the context of the elastic slider-crank and the Continuum Parallel Robot (CPR), all incorporated within the same main file to ensure uniformity.

3.1 Shooting Method

In the implementation of the Shooting method, we have designed the ShootingBVP function to address the critical task of solving the boundary value problem (BVP) integral to the geometric static modeling of our two systems. This function encapsulates a series of steps to derive the shape of the rods for a given actuation value.

The initialization phase involved extracting structural parameters such as rod lengths, initial configurations, and matrices A and $\bar{A}$. Subsequently, we establish the initial pose of the first rod by considering the position of the first joint as a reference point. Simultaneously for the second model, we initialize the second rod's pose, taking into account the distance between the two sliders.

A significant component of this process is the decomposition of the guess to retrieve planar wrenches applied at the base of the rods. This information is then utilized to compose the full se(3) wrenches for both rods, essential for the subsequent stages.

With the initial states defined, our attention turns to numerical integration. We numerically integrate the CosseratShootingODEs for both rods, a critical step in capturing the dynamic behavior of the system. The subsequent extraction of states at the tip further refines our understanding of the rods' positions and orientations.

Quantities such as position, orientation, and wrench at the tip are then computed for both rods. The geometrical error, a key metric, is determined by comparing the position and orientation of the two rod tips. Additionally, we ensure the balance of wrenches to maintain equilibrium within the system.

The end of the process resulted in the generation of a residual vector, denoted as R, encapsulating the computed geometrical error and the balance of wrenches. This residual vector serves as a indicator of the system's adherence to the defined constraints and equilibrium conditions.

In the first model it contains the position along y of the tip, which must always be zero, since it is constrained to move only along x axis; then we can make the same reasoning for the rotation of the tip, which has to be always zero, and last we have the force along x which must also be zero since the are no forces applied.

Regarding instead the case of the Residual for the continuum parallel robot, we have to consider the loop closure equation for the two rods, which means that at the tip they should have same angle and position, and also the sum between forces and moments applied should be zero for the equilibrium condition.

In subsequent stages, a root-finding method is applied within the Shooting function to identify the correct set of variables satisfying the BVP. This method, whether utilizing fsolve or a custom Newton-Raphson scheme with a numerical Jacobian, is integral in fine-tuning the system to yield the desired rod shapes in response to specific actuation values.

3.2 Strain Based Parametrization

The implementation revolves around the SBP function, is instrumental in determining the shape of the rods for a given actuation value. This section delves into the core aspects of the SBP methodology and the associated functions.

3.2.1 Initialization:

The function commences by extracting essential quantities and initializing the state vector components, encompassing generalized coordinates and planar wrenches. Meaningful initial values are assigned to certain components, setting the stage for the subsequent steps.

3.2.2 Finding the Solution:

The heart of the SBP method lies in finding the solution to the formulated problem. This involves computing the residual, achieved through the getResidual function. The residual encapsulates the deviation from the desired equilibrium state, and its determination marks a pivotal step in shaping the subsequent stages of the SBP workflow.

When examining the residual vector in Strain-Based Parameterization (SBP) method, there exists a critical term that plays a key role in modeling internal forces and achieving the internal balance between stress and elasticity. This term is expressed as $K_{\epsilon\epsilon}q_e = Q_a$.

- $K_{\epsilon\epsilon}$ represents the stiffness matrix associated with the elastic deformations of the system. It characterizes how internal forces, induced by deformations, relate to the system's elasticity.
- q_e denotes the vector of elastic generalized coordinates, representing the internal deformations within the system.
- Q_a signifies the internal generalized forces, reflecting the internal balance between stress and elasticity in the generalized coordinates vector space.

In essence, the term $K_{\epsilon\epsilon}q_e = Q_a$ captures the equilibrium condition within the elastic components of the system. It signifies that the internal balance between stress and elasticity is maintained by comparing the internal generalized forces (Q_a) with the internal elastic generalized forces ($K_{\epsilon\epsilon}q_e$). This comparison ensures that the system remains in a state of equilibrium, where internal forces are balanced by the elastic response of the system.

This relationship is crucial for understanding how SBP method models and maintains the internal balance within the simulated system.

3.2.3 Getting the Robot Shape:

Following the solution finding phase, the getRobotShapeSBP function is invoked to extract and present the shape of the rods. This function plays a important role in translating the obtained solution into a tangible representation, offering insights into the system's configuration.

3.2.4 Residual Computation:

The getResidual function, utilizes the Inverse Geometric Model (IGM) to obtain the pose at the rod tip and generalized forces at the base. It then navigates through the intricacies of the static model, involving the computation of generalized elastic forces, internal rod balance, and geometrical constraints.

3.2.5 Forward Kinematics:

Integral to the SBP methodology, the ForwardKinematics function computes the derivatives necessary for advancing the state vector during the numerical integration process. It involves the computation of strains, transformation matrices, and the derivatives of orientation and position.

3.2.6 Backward Statics:

The BackwardStatics function, a crucial component of the SBP methodology, computes the derivatives for the backward integration phase. It incorporates strains, external forces (such as gravity), and the derivatives obtained from the forward kinematics. The resulting derivatives contribute to the overall evolution of the state vector.

In summary, the SBP method offers a systematic and comprehensive approach to modeling the static behavior of the system. The integration of forward kinematics, backward statics, and residual computations provides a framework for understanding the interplay of forces and deformations within the rods.

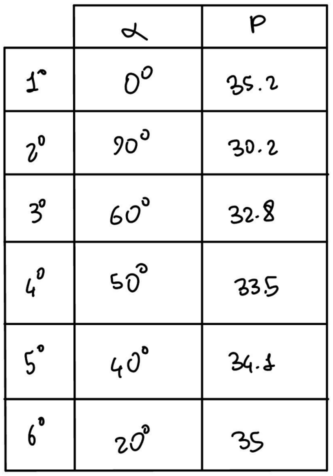
4 Working with Prototypes

The objective of this part is to compare the static simulations with some experimental data. Typically, when using nominal values for the beam parameters the model is not accurate and the parameters must be refined using the collected data.

4.1 Collecting data

Following the Experimental methodology for the two prototypes, a (small) dataset has been created. The dataset is composed by the input value (the rotation of the revolute joint, and the distance between the two prismatic joint, respectively for the two models) and the set of positions of the rod tip measured on the graph paper.

Starting with the prototype of the elastic slider-crank, the following dataset has been taken:



	α	p
1°	0°	35.2
2°	90°	30.2
3°	60°	32.8
4°	50°	33.5
5°	40°	34.1
6°	20°	35

Figure 3: Data Taken from phisical model

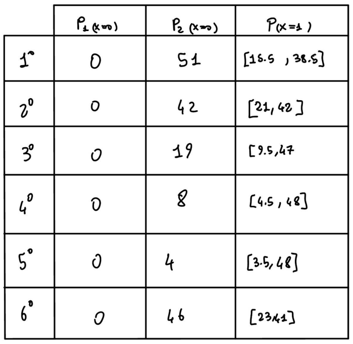
Where on the columns there are the values of α, which corresponds to the rotation of the revolute joint at the beginning of the structure, and then there is p, which is the position of the tip, that is only the x values of it since it can only translate along that direction.

And the following parameters have been taken into account for the simulation:

Name	Symbol	Value	Unit
Radius	r	$1e^{-3}$	m
Specific Weight	ρ	1900	Kg/m^3
Young Modulus	E	36.0	MPa
Shear Modulus	G	13.8	MPa

Figure 4: Nominal values for one fiberglass beam

While for the prototype of a Continuum Parallel robot we obtained the following values from the physical model:



	P_1 (x=0)	P_2 (x=0)	P(x=1)
1°	0	51	[15.5 , 38.5]
2°	0	42	[21, 42]
3°	0	19	[9.5,47
4°	0	8	[4.5 , 48]
5°	0	4	[3.5, 48]
6°	0	46	[23,41]

Figure 5: Data Taken from physical model

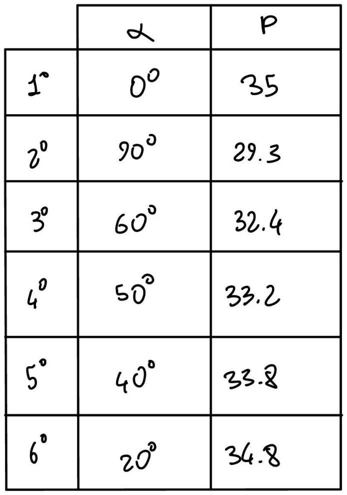
Where on the columns we have the positions of the two prismatic joint at x=0, and the resulting tip position for each configuration. The following parameters have been taken into account for the simulation:

Name	Symbol	Value	Unit
Length	ℓ	0.4	m
Radius	r	$1e^{-3}$	m
Specific Weight	ρ		Kg/m^3
Young Modulus	E		MPa
Shear Modulus	G		MPa

Figure 6: Parameters for the cantilever beam

4.2 Comparing Raw model

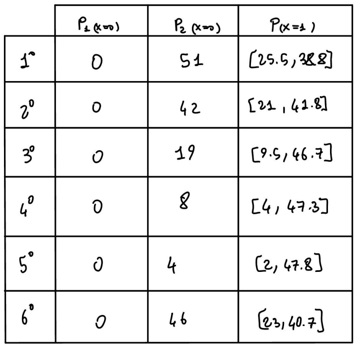
The values founded in the simulation for the Elastic Slider Crank are resumed in the table below:



	α	P
1°	0°	35
2°	90°	29.3
3°	60°	32.4
4°	50°	33.2
5°	40°	33.8
6°	20°	34.8

Figure 7: Data Taken from simulated model of Slider-Crank

While for the Continuum parallel robot we obtaines the following:



	$P_1 (x=0)$	$P_2 (x=0)$	$P(x=1)$
1°	0	51	[25.5, 38.8]
2°	0	42	[21 , 41.8]
3°	0	19	[9.5, 46.7]
4°	0	8	[4, 47.3]
5°	0	4	[2, 47.8]
6°	0	46	[23, 40.7]

Figure 8: Data Taken from simulated model of Continuum parallel robot

As we can see, there are few differences wich can depend by various factor that will be explained right below

4.3 Constitutive Laws and Deformation Modeling:

In the realm of soft robotics and the simulation of elastic structures, one critical aspect lies in defining how applied stresses deform the rod. This is achieved through the use of an operator $H \in R^{6\times 6}$, acting as a mapping from the space of deformations to the space of internal forces and moments. This operator is expressed in the local coordinate frame of the rod cross-section.

The operator H is decomposed into two matrices, $Hang$ and $Hlin$, each serving a distinct purpose. $Hang$ accounts for the effects of torsion and bending, while $Hlin$ captures the linear deformation behavior. These matrices are defined based on Hooke tensors along the beam and expressed in the local coordinates frame of the rod cross-section.

$$Hang = \begin{bmatrix} GJ & 0 & 0 \\ 0 & EI_{yy} & 0 \\ 0 & 0 & EI_{zz} \end{bmatrix}$$

$$Hlin = \begin{bmatrix} EA & 0 & 0 \\ 0 & GA & 0 \\ 0 & 0 & GA \end{bmatrix}$$

Here, G is the shear modulus, E is the Young modulus, A is the cross-sectional area, I_{yy} and I_{zz} are moments of inertia along principal axes, and J is the polar moment of inertia.

Using the matrix H, the relationship between internal wrenches $\Lambda(X)$ and strains $\xi(X)$ in the rod is expressed as:

$$\xi(X) = H^{-1}\Lambda(X) + \xi_c$$

In this equation, ξ_c represents the constrained strain, defining the shape of the rod when free from any internal or external wrench. For instance, if the rod is initially straight, ξ_c corresponds to a constrained strain that reflects a straight configuration.

By introducing these constitutive laws, you account for the material properties and deformation characteristics of the rods in the simulation. However, it's crucial to acknowledge that the constitutive laws rely on certain assumptions and approximations, and the differences between simulation and physical models may arise from these simplifications. Particularly, the matrix E in the model is an approximation that might not fully capture all bending and torsion effects observed in real-world scenarios. This distinction in the representation of material properties can contribute to the observed variations between simulated and physical outcomes.

4.4 Measurement Errors and Uncertainties:

Another potential source of discrepancies between simulated and physical outcomes arises from measurement errors and uncertainties. In practice, the accuracy of experimental measurements is susceptible to small uncertainties introduced during the measurement process. These uncertainties may stem from various factors, including the precision of the measurement devices, the scale used for measurements, or environmental conditions affecting the measurement apparatus.

Additionally, real-world materials are subject to inherent variations and deterioration over time. The properties of the model material may deviate from the assumed idealized values due to factors such as wear and tear, aging, or exposure to environmental conditions. These variations introduce uncertainties in the material properties used in the simulation, contributing to differences between the simulated and physical behaviors.

In summary, the observed disparities may, in part, be attributed to the unavoidable uncertainties associated with experimental measurements and the dynamic nature of real-world materials. These uncertainties underscore the importance of accounting for potential variations in physical conditions when comparing simulation results with experimental data.

4.5 Numerical Solution Challenges:

In the numerical solution of soft robotic systems, challenges may arise due to the intricacies of root-finding methods and the complexities associated with obtaining an optimal solution. However, several factors can contribute to potential errors in the numerical solution.

One notable challenge lies in the sensitivity of root-finding algorithms to initial guesses. The numerical methods employed, such as the Newton-Raphson scheme or fsolve, may struggle to converge to the correct set of variables if an inadequate initial guess is provided. This sensitivity becomes particularly pronounced when dealing with highly nonlinear or underdetermined systems, as is often the case in soft robotics.

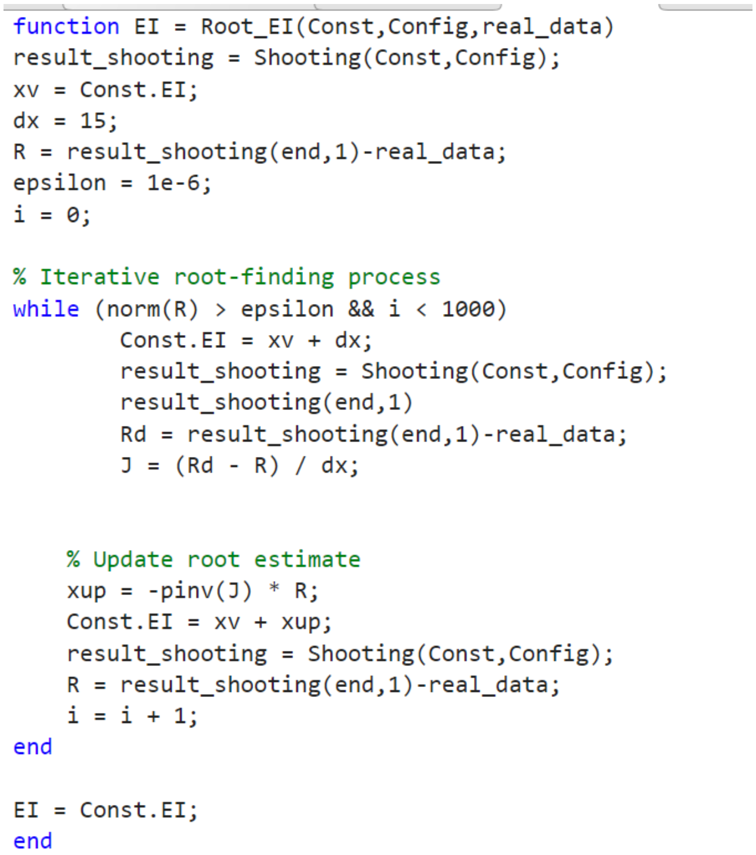
Additionally, the presence of multiple solutions or bifurcations in the system dynamics can pose challenges. Root-finding algorithms may converge to local minima or fail to explore the entire solution space, potentially leading to suboptimal or incorrect results. Robustness in handling such scenarios is crucial for ensuring the reliability of the numerical solution.

Furthermore, the numerical integration steps involved in solving differential equations, such as those governing the dynamic behavior of soft robotic systems, can introduce errors over time.
Considering different assumptions, as Kirchoff assumptions with no extensibility, we can rewrite the matrix as following:

$$H = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & EI_{yy} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Considering the main source of errors of the simulation,the nominal value assigned EI_{yy}, for this reason we have developed a root finding function for finding the right value of it.
This function takes in input the real value taken from the model, and update the value of

EI_{yy} until the position of the tip is almost the same of the real model in the same configuration. As we can se as following:



```
function EI = Root_EI(Const,Config,real_data)
result_shooting = Shooting(Const,Config);
xv = Const.EI;
dx = 15;
R = result_shooting(end,1)-real_data;
epsilon = 1e-6;
i = 0;

% Iterative root-finding process
while (norm(R) > epsilon && i < 1000)
        Const.EI = xv + dx;
        result_shooting = Shooting(Const,Config);
        result_shooting(end,1)
        Rd = result_shooting(end,1)-real_data;
        J = (Rd - R) / dx;


    % Update root estimate
    xup = -pinv(J) * R;
    Const.EI = xv + xup;
    result_shooting = Shooting(Const,Config);
    R = result_shooting(end,1)-real_data;
    i = i + 1;
end

EI = Const.EI;
end
```

Figure 9: Root Finding for the EI value

Unfortunately, the current function struggles to identify the correct value for EI_{yy}, seemingly oblivious to changes in the shooting function. Despite directly modifying the value in the rod's configuration file yielding the desired changes, we are actively working to resolve this anomaly.