TECHNICAL UNIVERSITY OF CLUJ-NAPOCA

ACTA TECHNICA NAPOCENSIS

Series: Applied Mathematics, Mechanics, and Engineering
Vol. 69, Issue Special I, February, 2026

ANALYTICAL AND COMPUTATIONAL MODELING OF AN SSC KINEMATIC ROBOT LIMB CONCEPT

Angela-Miruna NEACȘU-PAVEL, Ileana DUGĂEȘESCU, Elena CĂLIN, Mihaela-Elena ULMEANU, Alexandru-Ionuț NICOLESCU, Cristian-Vasile DOICIN

Abstract: *This paper presents the analytical and computational modeling of a robotic limb concept based on an SSC (Spherical–Spherical–Cylindrical) kinematic configuration, aimed at replicating the motion of an upper limb segment. The study focuses on the development of a Python-based simulation tool that facilitates both the validation of the mathematical model and graphical visualization of spatial trajectories in biomechanically inspired mechanisms. The modeled structure includes two spherical and one cylindrical joint, providing a simplified framework for studying articulated motion.*

The kinematic behavior of the system is described using position equations derived from forward kinematics, capturing the spatial orientation and displacement of the end-effector. The Python implementation integrates focused libraries such as NumPy and Matplotlib, employing a structured numerical approach to compute and visualize the limb's trajectory. The application supports user-defined input parameters, with preset configurations available for validation purposes. The results confirm that the SSC-based Python model reliably simulates the articulated motion of an upper limb, supporting its use in kinematic analysis, educational environments and biomechanical simulations.

Key words: *Python tools, spherical joints, kinematic analysis, robot limb concept.*

1. INTRODUCTION

The design of robotic limbs depends on the selection and modeling of kinematic architectures to ensure a balance between motion capability, control simplicity and mechanical feasibility. In recent years, focus has been placed on developing joint systems and limb structures that replicate or approximate biological movement strategies to improve both functionality and adaptability in robotics. Among these configurations, serial kinematic chains composed of spherical, pivotal or cylindrical joints have been used due to their capacity to provide multiple degrees of freedom for replicating human limb behavior [1, 2]. Cylindrical and spherical joints combine with translational motion and extend the functional versatility of robotic structures by allowing coupled position and orientation adjustments along constrained paths [3].

Research in robot kinematics modeling has proved the necessity for both accurate analytical frameworks and efficient computational tools to simulate and predict limb behavior. The development of soft robotics and compliant mechanisms have highlighted the importance of hybrid structures that integrate both rigid and flexible components. Such designs aim to increase adaptability and safety, particularly in domains involving human-robot interaction, rehabilitation and assistive technologies. Recent studies have demonstrated the potential of these hybrid approaches to enhance comfort and use without compromising precision or control fidelity [4, 5].

Xie et al. [6] investigated an equivalent kinematic model of the human upper limb for improving trajectory planning in rehabilitation exoskeletons. Their study obtained more accurate trajectory generation compared to traditional simplified limb models, enhancing the smoothness of simulated motion. However,

their framework relied primarily on revolute joint representations and did not incorporate hybrid spherical–cylindrical architectures, leaving unresolved challenges in modeling redundant DOFs and translational mobility within compact joint chains. Other researchers [7] developed a systematic synthesis of 3-DOF spherical hybrid mechanisms with fixed centers of rotation. Their approach provided novel mechanism topologies capable of achieving continuous rotational motion with improved structural compactness. Nevertheless, the study focused exclusively on spherical joint configurations and did not explore the integration of cylindrical translational motion, which limits the applicability of such mechanisms in tasks requiring extended reach or hybrid kinematic adaptability. Armanini et al. [8] provided a systematic overview of modeling strategies for soft and hybrid mechanisms, highlighting that accurate kinematic modeling remains a prerequisite for advancing bio-inspired robotics. While their work emphasized continuum structures, the integration of spherical–cylindrical hybrid joints within discrete chains remains underexplored. Similarly, Ryu [9] reviewed recent advances in parallel manipulators, focusing on the advantages of compactness and precision but also pointing out the limitations of existing architectures when translational adaptability is required. Fuente et al. [10] demonstrated adaptive robotic gait control strategies using inverse kinematics, which significantly improved trajectory smoothness. However, their approach was tailored to locomotion rather than upper-limb replication, leaving open the challenge of designing versatile joint configurations for articulated arm systems. Zheng et al. [11] further analyzed ABB-IRB2400 robot kinematics and trajectory planning, showing how industrial arms benefit from structured path generation. Yet, their reliance on revolute and prismatic joints did not address hybrid joint adaptability in constrained workspaces. Other authors [12] have explored multimodal control in upper-limb rehabilitation robots, achieving improvements in patient-specific adaptability. Nevertheless, their framework depended on conventional joint modeling, without examining the potential of SSC-type chains to simplify redundancy while extending reach.

Considering the abovementioned, the present research proposes an analytical and computational modeling of a novel SSC (Spherical–Spherical–Cylindrical) kinematic limb concept. The main objective is to determine whether this configuration, defined by a combination of cylindrical and spherical joints can serve as a viable structure for determining limb trajectories while simplifying control schemes. The hypothesis driving this study is that an SSC kinematic model offers enhanced versatility in modeling spatial trajectories compared to rotational chains, particularly in applications requiring reorientation and extension of the end-effector within a constrained workspace.

To validate this hypothesis, the analytical forward kinematics of the SSC mechanism is derived, and a computational model is implemented using Python as a platform for simulation and visualization. This choice reflects the increasing preference for open-source, accessible computational tools in robotics research and education. Compared to traditional environments like MATLAB, Python provides flexibility, community support and ease of integration with visualization libraries such as Matplotlib, making it a good choice for prototyping and analysis.

Beyond academic interest, hybrid kinematic models hold potential for diverse industrial and medical applications. In rehabilitation robotics, they can enable patient-specific exoskeletons with adaptable joint structures. In industrial manipulators, SSC-type mechanisms could provide compact and flexible motion. In humanoid and assistive robots, they can replicate human-like upper-limb mobility, improving safety and functionality in human–robot interaction.

SSC architecture remains underexplored compared to revolute or prismatic chains. Existing studies rarely integrate spherical–cylindrical combinations for simulating human-like motion in a computationally accessible manner. This study approaches the gap by developing an analytical SSC model and implementing a Python-based simulation tool,

demonstrating both theoretical feasibility and practical applicability.

2. ANALYSIS OF THE END-EFFECTOR, BOTH STRUCTURAL AND KINEMATIC

The kinematic analysis focuses on deriving the relationships between joint parameters and the end-effector's position and orientation. In parallel, the structural analysis studies how parameters affect stability and the position of the end-effector. These aspects must be considered together to ensure the system's functional accuracy, safety and durability.

In this context, the end-effector represents the terminal link of a robotic arm, serving as the interface between the robot and its environment for grasping, positioning or interaction tasks. Its performance is directly influenced by the design of the limb mechanism that actuates it, as well as the structural integrity and mobility characteristics of the system.

One of the aspects of end-effector analysis involves the determination of the system's Degrees of Freedom (DOF), a measure of the number of independent movements the end-effector can perform. In robotic systems, the DOF is directly tied to the kinematic chain's architecture and established the versatility of the manipulator. An SSC robot limb exhibits a redundant kinematic structure, meaning it has more DOFs than strictly necessary for a given task. This redundancy offers increased flexibility and adaptability, which is important for complex or constrained environments, but also demands advanced control and planning algorithms.

The degree of freedom (L) of a mechanism represents the number of independent motions it can perform. Calculating this parameter is essential to ensure adequate mobility and optimal functionality of the mechanism. In the case of a mechanism composed of multiple elements, the degree of freedom can be determined using a formula that accounts for the number of links, as well as the type and number of kinematic pairs involved [13]. The general formula used to compute DOF in planar mechanisms is given by equation (1).

$$L = 3n - 2i - s \quad (1)$$

where:

- n is the number of kinematic elements (links),
- i is the number of lower joints,
- s is the number of higher joints.

This formula provides a systematic method to evaluate the mobility of complex mechanisms, such as the SSC limb concept, which often integrates both conventional and compliant components.

Figure 1 presents the structural diagram of a humanoid robot concept, composed of 14 kinematic links interconnected through cylindrical and spherical joints.

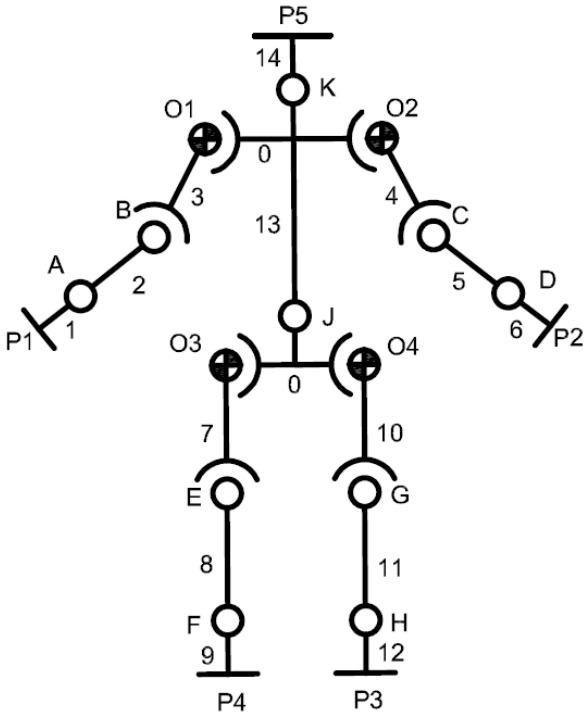

Fig. 1. Structural diagram of a bipedal robotic concept

The end-effectors, represented by points P1 through P5, serve as the terminal nodes, responsible for executing precise trajectories or manipulative tasks. These end-effectors represent the functional output of the kinematic chain and play an important role in evaluating task-space performance. Fixed spherical joints labeled O1 through O4 act as anchoring points within the system. These are immobile relative to the global reference frame and serve a dual role: they provide structural stability and act as the kinematic base, constraining global motion and defining the starting reference for mobility analysis.

The joints A, D, F, H, J, and K are cylindrical, meaning that the motion between the connected elements is restricted to rotation around a single axis. In contrast, the spherical joints B, C, E, and G allow rotation between two components around three axes, granting greater freedom of movement and enabling three-dimensional mobility. These joints are important in systems that require complex orientation of segments, closely resembling human joints such as the shoulder. Applying the data from the system presented in Figure 1 to the general formula (1), the degree of freedom is calculated as: $L = 3 \cdot 14 - 2 \cdot -8 = 22$.

This configuration, which incorporates both cylindrical and spherical joints, contributes to improved adaptability of the system in executing complex movements. More degrees of freedom allow for extended mobility and enhanced adaptability to diverse tasks and interactions, facilitating the integration of robots into various applications.

When developing a humanoid-inspired robotic system, it is necessary to draw parallels between the structure of the human body and that mechanical system. The human body is a complex biomechanical system, in which each segment of the upper limb has a well-defined role in mobility and control, replicating human functionality.

Kinematic elements 3 and 4 are responsible for flexibility and precision of movement and are functionally equivalent to the forearms. The arms, which provide stability and serve as support for the forearms, are represented by elements 2 and 5 in the robotic system. The hand, represented by elements 1 and 6, enables object manipulation and requires a high level of dexterity, which translates into the implementation of grasping and motion mechanisms. The kinematic elements that mimic lower-body movements such as the thighs (7, 10), shanks (8, 11) and feet (9, 12) contribute to ensure stability and maintaining postural equilibrium during motion.

The spinal column, represented by element 13, is considered a fixed reference point for coordinating movements. The fixed elements, labeled with 0, serve as structural support within the system.

After the analysis of structural diagram, it is necessary to concentrate on the kinematic analysis of a robotic system that emulates the human upper limb, examining specifically the motions occurring between its joints: the shoulder joints (O1 and O2), the elbow joints (B and C) and the wrist joints (A and D). These joints are modeled in the robotic system using a combination of cylindrical and spherical joints to mimic the mobility found in the human body.

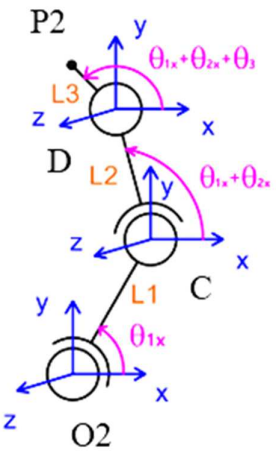

Fig. 2. Kinematic model of an SSC robot limb concept

The analysis presented in Figure 2 is structured around three key kinematic elements, each corresponding to a distinct anatomical segment:

- The upper arm (kinematic element L_1), which lies between the shoulder and the elbow, is represented by kinematic elements that ensure controlled lifting, rotation and positioning relative to the torso.
- The forearm (kinematic element L_2), located between the elbow and wrist, facilitates the extension and retraction of the hand, supporting intermediate adjustments during object manipulation or environmental interaction.
- The hand (kinematic element L_3), modeled as the terminal link in the kinematic chain. As the interface between the robotic system and its environment requires high dexterity and precision.

Each of these segments is analyzed individually within the system, highlighting their specific roles in generating movement and

enabling coordination. The shoulder (joint O2) provides a stable base and wide-range motion, the elbow (joint C) allows for directional adjustment and force control and the wrist (joint D) offers the fine motor capabilities necessary for precise interaction.

The model shown in Figure 2 includes three kinematic elements L_1, L_2, and L_3, representing the arm, forearm and hand. These are connected by joints O2 and C (spherical joints) and D (a cylindrical joint). The orientation of each segment is defined by the cumulative rotations relative to the fixed base, with the final orientation of the end-effector (point P2) expressed as the sum of sequential angles: $\theta_1x + \theta_2x + \theta_3$. Consequently, the concept (Figure 2) involves more complex joint interactions, requiring the model to account for complete joint mobility, which significantly influences the upper limb's overall motion and control strategy.

To determine the spatial configuration of the SSC robotic limb, it is needed to compute the Cartesian coordinates of anatomical reference points along the limb: the shoulder, elbow, wrist, and the end-effector.

- Position 1: Elbow Joint (joint C) marks the end of the first kinematic element L1 and the position is determined in terms of the first joint angle θ_{1x}, θ_{1y}, θ_{1z}.

$$X_1 = L_1 cos(\theta_{1x}) \quad (2)$$
$$Y_1 = L_1 sin(\theta_{1y}) \quad (3)$$
$$Z_1 = L_1 \sin(\theta_{1z}) \quad (4)$$

- Position 2: Wrist Joint (joint D) corresponds to the endpoint of the second kinematic element L2, extending from the elbow. It is influenced by angles θ_{1x}, θ_{1y}, θ_{1z} and θ_{2x}, θ_{2y}, θ_{2z}.

$$X_2 = X_1 + L_2 cos(\theta_{1x} + \theta_{2x}) \quad (5)$$
$$Y_2 = Y_1 + L_2 sin(\theta_{1y} + \theta_{2y}) \quad (6)$$
$$Z_2 = Z_1 + L_2 sin(\theta_{1z} + \theta_{2z}) \quad (7)$$

- Position 3: End-Effector (point P2), located at the end of the third kinematic element L3. Its position and orientation are influenced by the cumulative effects of all three joint rotations θ_{1x}, θ_{1y}, θ_{1z}, θ_{2x}, θ_{2y}, θ_{2z} and θ_3.

$$X_3 = X_2 + L_3 cos(\theta_{1x} + \theta_{2x} + \theta_3) \quad (8)$$
$$Y_3 = Y_2 + L_3 sin(\theta_{1y} + \theta_{2y} + \theta_3) \quad (9)$$
$$Z_3 = Z_2 + L_3 sin(\theta_{1z} + \theta_{2z} + \theta_3) \quad (10)$$

These equations serve as the foundation for forward kinematics in the SSC configuration, for planning trajectory motion simulation.

3. PYTHON APPLICATION

Python offers built-in data types, control structures and standard libraries that support quick development and automation [6]. Its dynamic changes and automatic memory management streamline the coding process, minimizing the complexity typically associated with lower-level programming languages. The implementation is based on a variety of well-established Python libraries, including NumPy, SciPy, Matplotlib, Pandas, SymPy, Scikit-learn, and GUI frameworks such as Tkinter, PyQt, or Streamlit [14, 15].

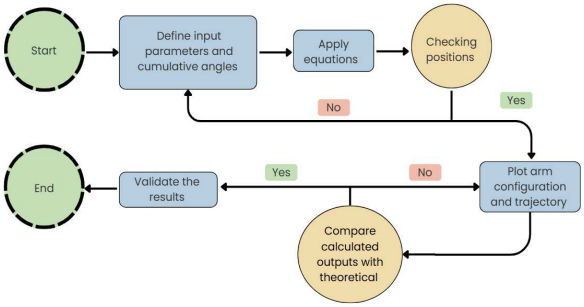

Fig. 3. Block diagram for the Python application

A block diagram (Figure 3) was developed to illustrate the logical structure of the simulation program designed in Python. The application focuses on simulating the motion of an SSC robot limb concept with one cylindrical and two spherical joints and three rigid links.

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3  from mpl_toolkits.mplot3d import Axes3D
4  L1 = 150
5  L2 = 130
6  L3 = 60
7  theta1x_0, theta2x_0 = np.pi/3, np.pi/4
8  theta1y_0, theta2y_0 = np.pi/6, np.pi/6
9  theta1z_0, theta2z_0 = np.pi/4, np.pi/6
10 theta3_0 = np.pi/6
11 t = np.linspace(0, 2*np.pi, 100)
```

```
theta1x = theta1x_0 + 0.2 * np.sin(t)
theta2x = theta2x_0 + 0.2 * np.sin(t + np.pi/3)
theta1y = theta1y_0 + 0.2 * np.sin(t + np.pi/6)
theta2y = theta2y_0 + 0.2 * np.sin(t + np.pi/2)
theta3 = theta3_0 + 0.2 * np.sin(t + np.pi/4)
theta1z = theta1z_0 + 0.2 * np.sin(t + np.pi/3)
theta2z = theta2z_0 + 0.2 * np.sin(t + np.pi/6)
x = (L1 * np.cos(theta1x) +L2 * np.cos(theta1x
+ theta2x) +L3 * np.cos(theta1x + theta2x +
theta3))
y = (L1 * np.sin(theta1y) +L2 * np.sin(theta1y
+ theta2y) +L3 * np.sin(theta1y + theta2y +
theta3))
z = (L1 * np.sin(theta1z) +L2 * np.sin(theta1z +
theta2z) +L3 * np.sin(theta1z + theta2z +
theta3))
fig = plt.figure(figsize=(8, 10))
ax = fig.add_subplot(111, projection='3d')
ax.plot(x, y, z, 'b-', linewidth=2, label='End-
effector')
num_positions = 10
indices = np.linspace(0, len(x) - 1,
num_positions, dtype=int)
for i in indices:
X1_i = L1 * np.cos(theta1x[i])
Y1_i = L1 * np.sin(theta1y[i])
Z1_i = L1 * np.sin(theta1z[i])
X2_i = X1_i + L2 * np.cos(theta1x[i]+
theta2x[i])
Y2_i = Y1_i + L2 * np.sin(theta1y[i] +
theta2y[i])
Z2_i = Z1_i + L2 * np.sin(theta1z[i]+
theta2z[i])
X3_i = X2_i + L3 * np.cos(theta1x[i]+
theta2x[i]+ theta3[i])
Y3_i = Y2_i + L3 * np.sin(theta1y[i]+
theta2y[i]+ theta3[i])
Z3_i = Z2_i + L3 * np.sin(theta1z[i]+
theta2z[i]+ theta3[i])
ax.plot([0, X1_i], [0, Y1_i], [0, Z1_i], 'r-',
alpha=0.5, linewidth=1.5, label="L1" if i ==
indices[0] else "")
ax.plot([X1_i, X2_i], [Y1_i, Y2_i], [Z1_i, Z2_i],
'g-', alpha=0.5, linewidth=1.5, label="L2" if i
== indices[0] else "")
ax.plot([X2_i, X3_i], [Y2_i, Y3_i], [Z2_i, Z3_i],
'm-', alpha=0.5, linewidth=1.5, label="L3" if i
== indices[0] else "")
ax.scatter([X1_i, X2_i], [Y1_i, Y2_i], [Z1_i,
Z2_i], c='pink', alpha=0.8, s=50)
ax.scatter(x[0], y[0], z[0], c='r', marker='o',
s=100, label="Start")
ax.scatter(x[-1], y[-1], z[-1], c='g', marker='o',
s=100, label="End")
ax.set_xlabel("X (mm)")
ax.set_ylabel("Y (mm)")
ax.set_zlabel("Z (mm)")
ax.legend()
fig.subplots_adjust(left=0.1, right=0.9,
bottom=0.1, top=0.9)
ax.set_zlabel("Z (mm)", labelpad=-30,
rotation=90)
plt.show()
```

The code simulates the kinematics of an SSC robot limb configuration, using sinusoidal functions to define the joint angle trajectories over time. At each step, the program calculates the forward kinematics by summing the joint angles to determine the positions of key anatomical reference points, including the elbow, wrist, and end-effector. The end-effector trajectory alongside the intermediate configurations of the arm is represented in simulation illustrating its motion within a 3D workspace.

4. RESULTS

The SSC robotic limb is parameterized to mimic human arm movement by specifying key inputs such as link lengths, joint limits, and the allowable stroke of the cylindrical joint. Based on these inputs, the model applies the established forward-kinematic formulation to determine the spatial configuration of the mechanism. In practice, this yields the positions of the key joint centers and the end-effector, along with the corresponding joint angles for the two spherical joints and the cylindrical translation. The resulting data provide a consistent pose description that can be used for visualization, analysis, or subsequent control design.

Table 1 lists the prescribed values for each link and joint, namely the link lengths and the initial joint orientation parameters. Together, these values define the baseline posture of the SSC chain (upper arm, forearm, and hand) and fix the reference configuration from which all poses are computed. They serve as the input set for the forward kinematic computations.

Table 1

Input Parameters

Parameters	Values
L_1	150 mm
L_2	130 mm
L_3	60 mm
θ_{1x}	60° = π/3
θ_{2x}	45° = π/4
θ_{1y}	30° = π/6
θ_{2y}	30° = π/6
θ_{1z}	45° = π/4
θ_{2z}	30° = π/6
θ_3	30° = π/6

By applying equations (2) to (10), the model's kinematic behavior was evaluated, allowing for the determination of joint configurations and end-effector positions under the given input conditions. The results are summarized in Table 2.

Table 2

Positions for elbow, wrist and end-effector

Positions [mm]		
Elbow (joint C)	x_1	75
	y_1	75
	z_1	106.07
Wrist (joint D)	x_2	41.35
	y_2	187.58
	z_2	231.64
End-effector (pont P2)	x_3	-1.08
	y_3	247.58
	z_3	289.60

Python code segments were developed to implement and validate the mathematical model to computationally apply the forward kinematic equations (2) through (6). These equations represent the positional relationships between the joint angles and the resulting spatial coordinates of the elbow, wrist and end-effector. The simulation computed the end-effector positions at ten distinct configurations, each corresponding to a unique set of joint angles, systematically varying to evaluate the model's response.

To support the computational and graphical components of the simulation, several Python libraries were employed [15]. The *NumPy* library was imported to facilitate efficient numerical operations, including matrix manipulation and trigonometric evaluations essential for forward kinematic calculations. In parallel, *Matplotlib.pyplot* was used to generate static and animated visualizations of the robotic arm's motion within the three-dimensional workspace. The *Mpl_toolkits.mplot3d* module was incorporated to allow for potential three-dimensional plotting, enabling future extensions of the model into spatial domains.

First, the robotic limb's three kinematic segments: L_1 (upper arm), L_2 (forearm) and L_3 (wrist) were assigned fixed length values representative of human anatomical proportions. To simulate smooth, biologically inspired motion, sinusoidal joint trajectories were generated for each of the revolute joints using the sine function from the *NumPy* library. This approach models the rhythmic, oscillatory nature of arm movements. Variables were initialized to store the three-dimensional coordinates of the intermediate joints and the end-effector along the motion path. The simulation iterated over a user-defined set of trajectory points, applying forward kinematic equations at each step to compute the spatial positions of the joints based on the cumulative joint angles. These positions were recorded in arrays corresponding to each joint and the end-effector, allowing for detailed tracking of the limb's configuration over time.

Graphical representation of the limb segments was achieved by plotting each link with distinct colours and a set transparency level to visually distinguish the anatomical components and illustrate the articulated nature of the arm. For clarity, joint positions were highlighted at discrete intervals along the trajectory, using slicing techniques to separate the segments corresponding to L_1 (shoulder to elbow), L_2 (elbow to wrist) and L_3 (wrist to end-effector). The resulting 3D plot (Figure 4) displays the dynamic motion of the robotic arm, with coloured lines tracing the intermediate segments and a continuous blue curve delineating the complete path of the end-effector. Start and end points of the trajectory are distinctly marked to provide clear reference landmarks for the simulated motion cycle.

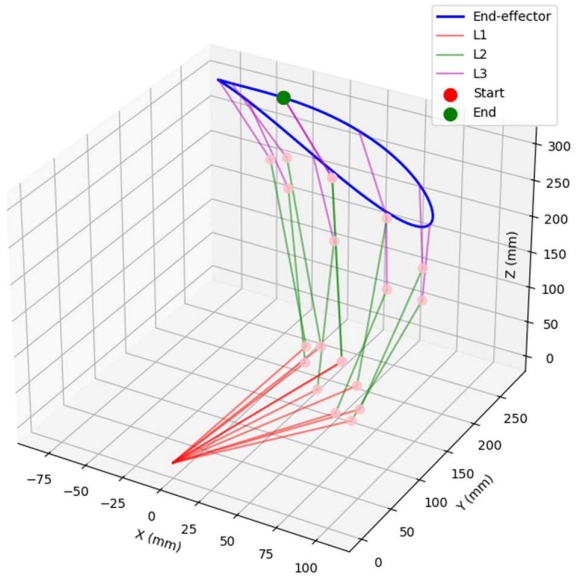

Fig. 4. End-effector trajectory in 3D workspace

Sinusoidal profiles were applied to the joint angles to mimic simple, repeatable motion cycles. Each angle varies producing smooth flex–extend and reach–return patterns similar to basic human gestures and repetitive robotic tasks. These periodic inputs give continuous velocities and accelerations (low jerk), make coordination between joints easy to tune (in-phase or out-of-phase), and provide a clear basis for testing joint limits, tracking, and the resulting end-effector paths.

Driving each joint with sine waves that use different frequencies and phase offsets creates varied timing across the chain. This produces simple superposition effects leading to richer, non-repeating coordination patterns and more realistic task cycles. The approach helps test coupling between joints, exposes a wider portion of the workspace, and makes it easy to study how timing choices affect the end-effector path and joint load. Specifically, the first joint oscillates through a single sine cycle, the second through two cycles, and the third through three cycles within the same time frame, producing a trajectory characterized by looping and non-linear spatial paths. This multi-frequency approach mimics the natural asynchrony and coordinated fluidity inherent to biological limbs, where joints rarely move in perfect synchrony but rather combine their motions to achieve smooth, naturalistic movement.

The end-effector coordinates (x_1, x_2, x_3) produced by the Python code align with the theoretical values derived from the closed-form forward kinematics equations (Equations 2–10). This validates that the mathematical model accurately represents the kinematic behaviour of the SSC robot limb concept under the simulated conditions.

The sinusoidal joint trajectories employed in the simulation were selected to replicate typical cyclic motions commonly observed in robotic manipulators and human-like arm gestures. Although the current validation emphasizes static numerical correspondence, subsequent research may incorporate real-time robotic testbeds or integrate the model within comprehensive physics-based simulation environments to evaluate its dynamic consistency and practical performance.

5. CONCLUSION

The state-of-the-art studies indicate a persistent gap: while many works advance spherical or cylindrical architectures independently, few integrate them into a combined SSC chain suitable for replicating human-like limb behaviour. This research directly addresses this limitation by proposing and validating a hybrid SSC kinematic model through both analytical derivations and a Python-based computational platform.

The analytical and computational modelling of the SSC robotic limb concept demonstrated in this study provides a framework for simulating spatial trajectories of humanoid-inspired limbs. The Python-based simulation effectively captures the forward kinematics of a three-link robotic arm with a hybrid joint configuration, offering clear visualization of the end-effector's 3D motions. Some advantages of this approach include the use of accessible open-source tools, flexibility in modeling biologically inspired joint movements, and the capacity to explore kinematic behaviors that mimic human limb functionality. The model currently assumes ideal joint behavior without considering dynamic effects, actuator constraints, or physical interaction forces, which restricts its applicability to static or quasi-static analysis.

The simulation relies on sinusoidal joint trajectories that, while useful for demonstrating motion patterns, do not reflect task-specific or optimized control strategies. It is important to note that the SSC configuration requires further validation in real robotic hardware or more sophisticated physics-based environments to assess its performance in practical scenarios. Expanding the model to incorporate dynamic modeling, real-time control, and integration with experimental platforms will significantly enhance its relevance for advanced robotic applications in rehabilitation, assistive devices and humanoid robotics.

Beyond validation, the study also lays the groundwork for future research into multi-DOF optimization, control strategies for redundancy resolution, and extensions toward compliant and continuum joint integration, thereby bridging a critical gap between theoretical kinematics and applied robotics. Future research may also focus on incorporating dynamic modelling and advanced control strategies, as well as validating the concept in real robotic concepts. Additional directions include integrating machine learning-based controllers, applying the model in rehabilitation exoskeletons and testing its performance in industrial case studies.

6. REFERENCES

[1] Armanini, C., Boyer, F., Mathew, A.T., Duriez, C., Renda, F., *Soft Robots Modeling: A Structured Overview*, IEEE Transactions on Robotics, 39(3), 1728-1748, 2023.

[2] Ryu, J.H., *Parallel Manipulators, New Developments*, ISBN 978-953-51-5825-7, Korea, University of Technology and Education, 2019.

[3] Fuente, L.A., Lones, M.A., Turner, A.P., Caves, L.S., Stepney, S., Tyrell, A.M., *Adaptive Robotic Gait Control using Coupled Artificial Signalling Networks, Hopf Oscillators and Inverse Kinematics*, ISBN 978-953-51-5825-7, Korea, University of Technology and Education, 2019.

[4] Zheng, Q., Jia, J., Zhu, P., Xiao, Y., Zhang, F., Ma, W., Zhao, S., *Kinematics Analysis and Trajectory Planning of ABB-IRB2400 Robot*, U.P.B. Scientific Bulletin, Series D: Mechanical Engineering, 84(2), ISSN 1454-2358, 2022.

[5] Ren, H., Liu, T., Wang, J., *Design and Analysis of an Upper Limb Rehabilitation Robot*

Based on Multimodal Control, MDPI, Sensors 2023, 23, 8801, 2023.

[6] Xie, Q., Meng, Q., Zeng, Q., Yu, H., Shen, Z, *An innovative equivalent kinematic model of the human upper limb to improve the trajectory planning of exoskeleton rehabilitation robot,* Mechanical Sciences, vol. 12, 661-675, 2021.

[7] Shi, S., Li, K., Ma, Z., Wang, H., Qiu, X., Zhou, Y., *Type Synthesis of 3-DOF Spherical Hybrid Mechanisms with Fixed Centers of Rotation*, Advances in Mechanical Engineering, vol. 16, issue 8, 2024.

[8] Armanini, C., Boyer, F., Mathew, A.T., Duriez, C., Renda, F., Soft Robots Modeling: A Structured Overview, IEEE Transactions on Robotics, 39(3), 1728–1748, 2023.

[9] Ryu, J.H., Parallel Manipulators, New Developments, ISBN 978-953-51-5825-7, Korea, University of Technology and Education, 2019.

[10] Fuente, L.A., Lones, M.A., Turner, A.P., Caves, L.S., Stepney, S., Tyrell, A.M., Adaptive Robotic Gait Control using Coupled Artificial Signalling Networks, Hopf Oscillators and Inverse Kinematics, in Bio-inspired Systems: Computational and Ambient Intelligence, Springer, 2009.

[11] Zheng, Q., Jia, J., Zhu, P., Xiao, Y., Zhang, F., Ma, W., Zhao, S., Kinematics Analysis and Trajectory Planning of ABB-IRB2400 Robot, U.P.B. Scientific Bulletin, Series D: Mechanical Engineering, 84(2), ISSN 1454-2358, 2022.

[12] Ren, H., Liu, T., Wang, J., Design and Analysis of an Upper Limb Rehabilitation Robot Based on Multimodal Control, Sensors, 23(8801), MDPI, 2023.

[13] Duca, C.D., Buium, F., Pârăoaru, G. *Mecanisme*, Editura "Gh. Asachi" Iași, *2023*.
[14] Python, https://www.python.org/downloads/
[15] Python Libraries, https://docs.python.org/3/library/index.html

Modelare analitică și numerică a unui concept cinematic de braț robotic de tip SSC

Această lucrare prezintă modelarea analitică și computațională a unui concept de membru robotic bazat pe o configurație cinematică SSC (Sferic–Sferic–Cilindric), având ca scop reproducerea mișcării unui segment de membru superior. Studiul se concentrează pe dezvoltarea unui instrument de simulare în Python care facilitează atât validarea modelului matematic, cât și vizualizarea grafică a traiectoriilor spațiale ale mecanismelor biomecanice inspirate. Structura modelată include două articulații sferice și una cilindrică, oferind un cadru simplificat pentru studiul mișcărilor articulate. Comportamentul cinematic al sistemului este descris prin ecuații de poziție, derivate din cinematica directă, stabilind orientarea spațială și deplasarea end-effectorului. Implementarea în Python integrează biblioteci precum NumPy și Matplotlib, utilizând o abordare numerică structurată pentru a calcula și vizualiza traiectoria elementelor cinematice care alcătuiesc conceptul de membru robotic. Aplicația permite introducerea parametrilor de intrare de către utilizator, având totodată configurații prestabilite pentru validarea modelului. Rezultatele confirmă că modelul Python bazat pe configurația SSC simulează în mod fiabil mișcarea articulată a unui membru superior, susținând utilizarea sa în analiza cinematică, în medii educaționale și în simulări biomecanice.

Angela-Miruna NEACȘU-PAVEL, PhD. Student, Assistant, National University of Science and Technology POLITEHNICA Bucharest, Faculty of Industrial Engineering and Robotics, Department of Manufacturing, angela.neacsu@upb.ro, Splaiul Independenței 313, București 060042.
Ileana DUGĂEȘESCU, PhD. Eng., Assoc. prof., National University of Science and Technology POLITEHNICA Bucharest, Faculty of Industrial Engineering and Robotics, Department of Manufacturing, ileana.dugaesescu@upb.ro, Splaiul Independenței 313, București 060042.
Elena CĂLIN, corresponding author, PhD. Student, National University of Science and Technology POLITEHNICA Bucharest, Faculty of Industrial Engineering and Robotics, Department of Manufacturing, elena.calin@upb.ro, Splaiul Independenței 313, București 060042.
Mihaela-Elena ULMEANU, corresponding author, PhD. Eng., Assoc. prof., National University of Science and Technology POLITEHNICA Bucharest, Faculty of Industrial Engineering and Robotics, Department of Manufacturing, mihaela.ulmeanu@upb.ro, Splaiul Independenței 313, București 060042; Academy of Romanian Scientists, Ilfov 3, 050044 Bucharest, Romania, +40 766 289 886.
Alexandru-Ionuț NICOLESCU, PhD. Student, University POLITEHNICA of Bucharest, Department of Manufacturing, anicolescu@upb.ro, Splaiul Independenței 313, București 060042.
Cristian-Vasile DOICIN, PhD. Eng., Professor, National University of Science and Technology POLITEHNICA Bucharest, Faculty of Industrial Engineering and Robotics, Department of Manufacturing, cristian.doicin@upb.ro, Splaiul Independenței 313, Bucharest 060042.