2022 IEEE International Conference on Engineering Veracruz (IEEE ICEV 2022®)

An adaptation of FABRIK algorithm for serial robot's inverse kinematics

1st Luis Antonio Orbegoso Moreno[a], 2nd Josmell Henry Alva Alcántara[b]
Department of Mechatronics Engineering
National University of Trujillo
Trujillo, Perú
lorbegoso@unitru.edu.pe[a], josalva@unitru.edu.pe[b]

2022 IEEE International Conference on Engineering Veracruz (ICEV) | 978-1-6654-5213-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICEV56253.2022.9959663

***Abstract*— Forward And Backward Reaching Inverse Kinematics (FABRIK) is an avant-garde heuristic method characterized by its simplicity and speed of convergence, mainly used in the field of computer graphics that provides a solution to the inverse kinematics (IK) for kinematic chains, formed mainly by spherical joints. However, FABRIK still has limitations for applications in robotics because robots generally use one-dimensional joints. For this reason, this paper develops a method to solve the IK of manipulator robots based on FABRIK, taking into account the hinge, pivot and prismatic joints that make up the robot's kinematic chain, while keeping the joint values within their limits. Thus, the resulting method turned out to solve the IK of a hypothetical serial redundant manipulator of 10 degrees of freedom (DOF) using much fewer iterations and expending less time of execution to get the smallest root-mean-square error (RMSE) between the end-effector and the target in comparison with the Damped Least Squares (DLS) method.**

Keywords*— *Inverse kinematics, FABRIK, Robotic manipulators, Joint Configuration.

I. Introduction

Inverse kinematics consists of finding the joint values of a kinematic chain that allow the final effector to reach a target [1]. The solution to this problem has many applications in the areas of computer graphics, video games, molecular biology and robotics [7,8]. For non-redundant kinematic chains, the IK solution can be analytical and direct, but, as the DOF increases, the problem becomes more complex, so new techniques are required [2].

In [3] is introduced the use of Jacobian Transpose, Jacobian Pseudoinverse and the DLS methods to solve the IK problem. All these methods are characterized by being based on Newton's method and avoiding the singularity problems of the Jacobian matrix; however, they have a high computational cost. To overcome this issue, in [4] is increased the speed of convergence of the DLS method by using dual quaternions. Meanwhile, in [5] is presented a constrained DLS method using the composition of constrained functions for joints to ensure that the joint values remain within the established joint limits.

On the other hand, heuristic methods offer simpler ways to solve the IK problem. One of the most popular is the Cyclic Coordinate Descent (CCD) algorithm, firstly introduced in [6], which has had applications in the video game industry [7] and in protein folding computation [8]. Another important heuristic method is FABRIK, first presented by Aristidou and Lasenby in [9]. It is characterized by having two stages: the forward stage and the forward stage, as is illustrated in Fig.1.

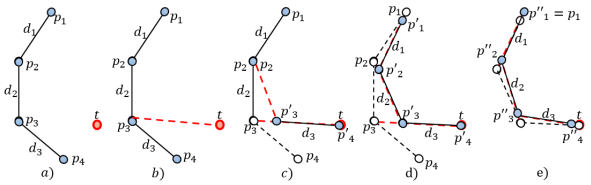

Fig. 1. Example of a FABRIK iteration for a planar 4-DOF manipulator; (a) the beginning of the forward stage defining the starting position and the target, (b) and (c) the links are repositioned in segments defined by the joints displacements, (d) the end of the forward stage once is reaching the kinematic's root; (e) execution of the backward stage that repeats the previous steps but targeting the original root position

FABRIK is characterized by its quick and simple solution of kinematic chains mainly composed of spherical joints Fig.2 (a), which differs from the morphology of robotic manipulators whose kinematic chains are mainly composed of hinge and pivot joints as shown in Fig.2 (b,c).

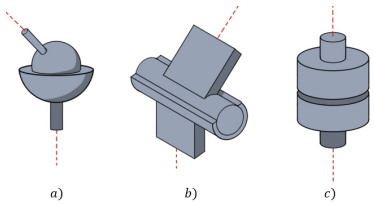

Fig. 2. (a) Spherical joint, (b) hinge joint and (c) pivot joint

One work that seeks to adapt the FABRIK algorithm for robotics is presented in [10] where FABRIK was able to work on a kinematic chain made up of only hinge joints described by Denavit-Hartenberg (DH) parameters. Another

978-1-6654-5213-7/22/$31.00 ©2022 IEEE

paper, presented in [11] introduces a variant of the algorithm called R-FABRIK which uses trigonometric equations to calculate the angle of twist of 1-DOF revolute joints to reposition the links of robotic kinematic chains in the forward and backward stage; however, this solution is not compared to other methods. For mobile manipulators, in [12] is presented another adaptation named M-FABRIK which allows the robot to be positioned according to various criteria, decreasing convergence time. To deal with obstacles, in [13] is extended the FABRIK algorithm to avoid obstacles in the task space. Recently, in [14] was developed a custom algorithm based on FABRIK and using the homogeneous transformation matrices with DH parameters to control the UR5 space manipulator robot.

This paper aims to extend the FABRIK algorithm to solve the IK of manipulator robots taking into account the types of joints that constitute its kinematic chain (hinge, pivot and prismatic joints) while keeping the joint values within their limits, by using common vector operations. In addition, the performance of this method was evaluated by comparing the results with the DLS method in iteration time, RMSE and the number of iterations for solving the IK problem for a hypothetical serial redundant manipulator of 10 DOF.

The remainder of the paper is organized as follows: in Section 2 the theories and mathematical expressions used to express the coordinates, the error and the kinematic model are explained. The explanation of the proposed algorithm is presented in Section 3. The comparative results and analysis of the study is shown in Section 4. And finally, the conclusions are given in Section 5.

II. Materials and Methods

A. Position, rotation and error measurements

3D Cartesian vectors and 3x3 rotation matrices were used to represent position and orientation in space, respectively. Thus, the absolute error in position E_P is just the difference between the target position P_t and the end-effector position P_f.

$$E_P = P_f - P_t \tag{1}$$

For the orientation error E_O between the target orientation O_t and the end-effector orientation O_f, the axis vector of the rotation matrix $M=O_f O^T_t$ was used to represent this orientation error, as is shown in the equation (2) where $m_{i,j}$ is an element of the M matrix.

$$E_O = [m_{3,2} - m_{2,3}, m_{1,3} - m_{3,1}, m_{2,1} - m_{1,2}] \tag{2}$$

If the error is just due to the position, the RMSE will be as follows.

$$RMSE = |E_P| \cdot \frac{1}{\sqrt{3}} \tag{3}$$

Now, if the error is due to the position and orientation, the $RMSE$ will be as is shown in (4).

$$RMSE = \sqrt{|E_P|^2 + |E_O|^2} \cdot \frac{1}{\sqrt{6}} \tag{4}$$

For this work, 200 is the limit number of iterations to reach an RMSE less than or equal to 10 for both the proposed method and the DLS method.

B. DLS method development

For the implementation of the DLS method, dual quaternions were used according to [4], in addition to the constrained differential IK method presented in [5] to keep the joint values within the limits.

C. Kinematic representation

The Screw Theory was used to describe the forward kinematics of a hypothetical redundant 10 DOF test robot shown in Fig 3, only defining a rotation and a translation axis relative to the coordinate frame of the previous link. This fact allowed to create a clear difference between pivot joints and hinge joints. For this work, all translations were done on the z-axis, whereas, pivot and hinge joint rotations were carried out around the z and x-axis respectively. The sequence of these rotations and translations of the test robot's kinematic chain is shown in Table I.

TABLE I
Parameter of the Forward Kinematics according to the Screw Theory

Link number	Rotation		Translation	
	Axis	*Value (rad)*	*Axis*	*Value (cm)*
1	[0, 0, 1]	[-π, π]	[0, 0, 1]	2
2	[1, 0, 0]	[-π⁄, π⁄]	[0, 0, 1]	2
3	[0, 0, 1]	[-π, π]	[0, 0, 1]	[7, 11]
4	[1, 0, 0]	[-π⁄, π⁄]	[0, 0, 1]	2
5	[0, 0, 1]	[-π, π]	[0, 0, 1]	[7, 11]
6	[1, 0, 0]	[-π⁄, π⁄]	[0, 0, 1]	2
7	[0, 0, 1]	[-π, π]	[0, 0, 1]	[6, 10]

As can be appreciated in Fig 3 (a), the articulations q_1, q_3, q_5 and q_7 belong to pivot joints; whereas q_2, q_4 and q_6 belong to hinge joints. Moreover, links l_3, l_5 and l_7 are considered prismatic joints for their variable length.

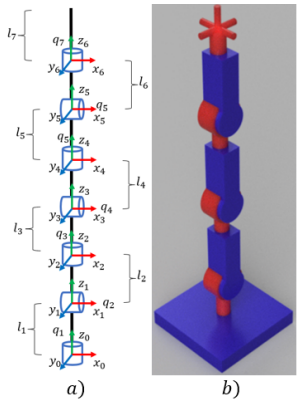

Fig. 3. (a) Kinematic chain model of the 10 DOF test robot and (b) CAD model of the test robot

The robot's starting pose is when all the revolute joint values are zero and all prismatic joints are at their minimum value by default; displaying as shown in Fig 3 (b). Because in this posture, the robot is in a configuration with singularity problems.

III. PROPOSE

The algorithm, whose flowchart is shown in Fig.4 (a), starts defining the target position-orientation. If the error between the target and the end-effector is smaller or equal to 1e-3 units, the algorithm ends; otherwise, it continues with the Forward step (so far, it's half an iteration). After that, if the error between the robot's fixed base and the new base obtained is smaller or equal to 1e-3 the algorithm will finish; otherwise, it returns with the Backward step (so far, it's a complete iteration) and repeats the previous steps. Unlike the original FABRIK algorithm, Fig.4 (b) , which only has a single condition clause for each iteration.

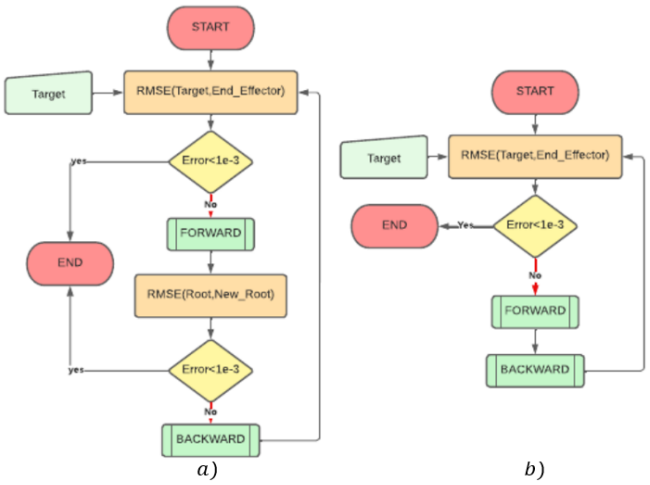

Fig. 4. (a) Proposed algorithm's flowchart and (b) Original FABRIK algorithm's flowchart

The backward stage and the forward stage do the same thing but starting from the opposite ends of the kinematic chain, the steps of the forward stage of the proposed algorithm will be explained below.

A. Solution for the Hinge Joint

First, the position P'_{i+1} and the orientation axes [x'_{i+1}, y'_{i+1}, z'_{i+1}] of a certain hinge joint $i+1$ are known (Fig. 5 (a)). The challenge is to determine the next position and orientation coordinates of the joint i. The solution of this first approximation is to define the next z'_i (Fig. 5 (b)) which can be calculated by projecting the distance vector $d_i=P'_{i+1} - P_i$ into the plane L which is orthogonal to x'_{i+1}, and then normalizing that projected vector as indicated in the following equation.

$$z'_i = \frac{d_i-(d_i\cdot x'_{i+1})x'_{i+1}}{|d_i-(d_i\cdot x'_{i+1})x'_{i+1}|} \quad (5)$$

In case the denominator equal to zero in (5), this means that z'_i can be whatever unit vector perpendicular to x'_i , so it is considered to take $z'_i=z'_{i+1}$. Now, once the axis z'_i is determined, the next position P'_i (Fig. 5 (c)) can be calculated with the following equation.

$$P'_i = P'_{i+1} - z'_i l_i \quad (6)$$

Then, knowing that for a hinge joint, the next joint inherits the x-axis of rotation, this is $x'_i=x'_{i+1}$ because the rotation is around the x-axis and remains static, so the y'_i axis can be determined by $y'_i=z'_i\times x'_i$. So far, both the position and the orientation of the joint i have been calculated, as is shown in Fig. 5 (d).

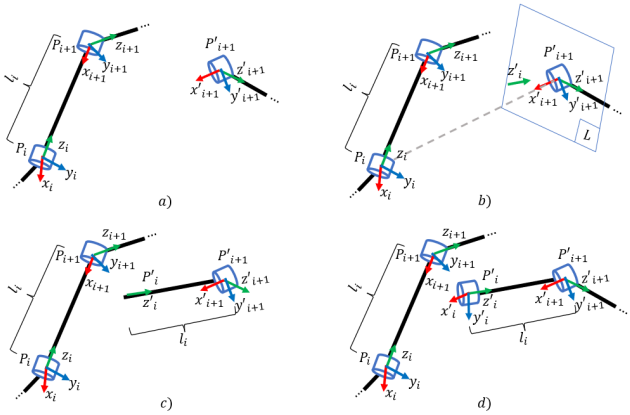

Fig. 5. Steps to find the solution of a hinge joint

However, if there is a restriction in the amplitude of the hinge joint's angle q_i, first this angle must be computed by the following relation.

$$q_{i+1} = cos^{-1}(z'_i \cdot z'_{i+1})sgn((z'_i \times z'_{i+1}) \cdot x'_i) \quad (7)$$

Now, if q_i is greater or less than a set limit q_L; the new z'_i will be as indicated below.

$$z'_i = z'_{i+1}cos(q_L) + y'_{i+1}sin(q_L) \quad (8)$$

B. Solution for the Pivot Joint

The position and orientation of a certain pivot joint $i+1$ is known (Fig. 6 (a)). As this joint rotates around the z-axis, this means that $z'_i=z'_{i+1}$. Nonetheless, z'_i is enough to determine the next joint position P'_i using equation (6) (Fig. 6 (b)). To compute the new x'_i, we can take advantage that usually before or after a pivot joint there must be a hinge joint. So, the next step is to find the position of another hinge joint (joint position P_{i-2} in Fig. 6) and then get the distance vector $d_i=P'_i-P'_{i-2}$. And as all translations are done on the z-axis, the unit vector of d_i must be the next z'_{i-1} axis. Therefore, as hinge joints the rotations are done around the x-axis, this new one will be as follows.

$$x'_i = \frac{z'_i\times z'_{i-1}}{|z'_i\times z'_{i-1}|} \quad (9)$$

If the denominator in (9) turns out to be zero, this means that x'_i can be whatever unit vector perpendicular to z'_i, so it is considered to take $x'_i=x'_{i+1}$. Another issue with the

expression (9) is that there is a problem with unexpected flipping of the new x-axis concerning its previous direction. To avoid this problem, then this new x-axis can be updated using $x'_i = x'_i sgn(x'_i \cdot x_i)$. Now, with the value of x'_i determined, the axis y'_i can be calculated and therefore all the coordinate axes are completed allowing us to define the next position P'_i and the orientation of the joint i as is shown in Fig. 6 (c).

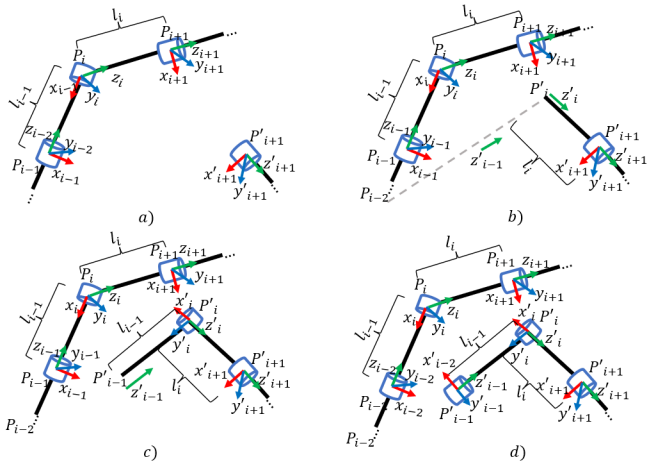

Fig. 6. Steps to find the solution of a pivot joint

In case there is a restriction in the amplitudes of the joint's angle, its value can be calculated as follows.

$$q_{i+1} = cos^{-1}(x'_i \cdot x'_{i+1}) sgn((x'_i \times x'_{i+1}) \cdot z'_i) \quad (10)$$

And if it exceeds an angle value q_L, so the axis x'_i will be corrected as it's indicated in the next equation.

$$x'_i = x'_{i+1} cos(q_L) - y'_{i+1} sin(q_L) \quad (11)$$

C. Solution for Prismatic Joints

The new length l'_i is straightforwardly computed by projecting the distance vector d_i into the z'_i axis previously computed (Fig. 7 (a)) using (12). Then the next position P'_i (Fig. 7 (b)) can be calculated using equation (6). If it is the case that l'_i exceeds a minimum or maximum allowable elongation value, which can be replaced directly by the respective limit value.

$$l'_i = d_i \cdot z'_i \quad (12)$$

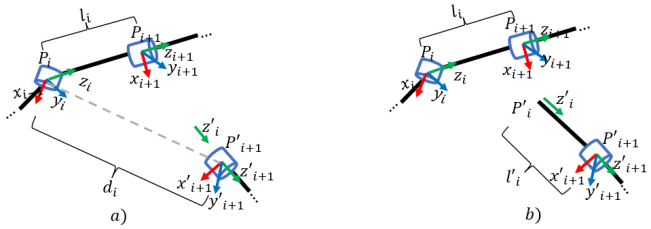

Fig. 7. Steps to find the solution of a prismatic joint

D. Solution for just target position

Firstly, it's necessary to estimate a target orientation along with the defined target position; then proceed with the same steps as above for the rest of the chain. A good estimation is to define a new z'_n, which is the unit vector of the distance between the target position and the position of the first hinge joint in the kinematic chain (Fig. 8 (a)). Once the new z-axis is defined, the other axes can be calculated using the simplified solution of Olinde Rodrigues around the rotor $k=(z'_n \times z_n)/|z'_n \times z_n|$ and angle $\theta = cos^{-1}(z'_n \cdot z_n)$, which leads to Fig.8 (b).

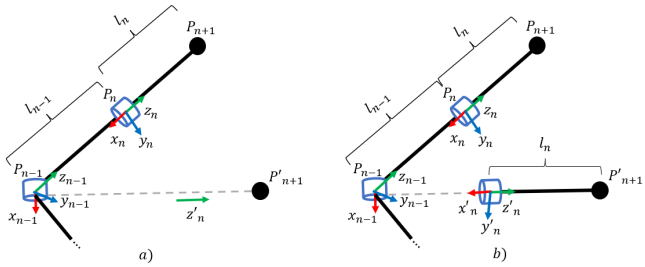

Fig. 8. Steps to estimate of the target orientation for a case with just target position

IV. Test and Results

Some random end-effector positions and orientations targets, shown in Table II, were used to test the proposed method to then be compared with the performance of the DLS method.

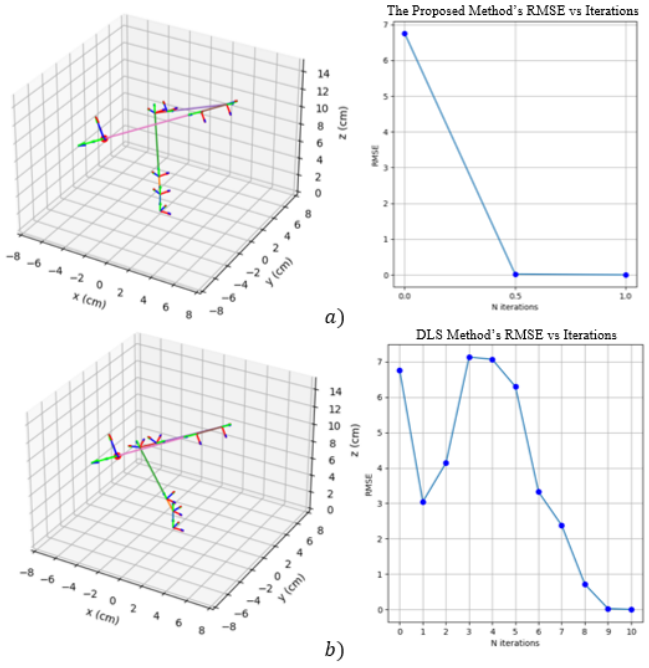

Fig. 9. Comparison between (a) the proposed method's result and (b) the DLS method's result for the same target position-orientation

The first target case is illustrated in Fig.9, where on the left side is shown the graphical representation of the solution and on the right side is shown a chart of the RMSE at its respective iteration number, proving that with both methods, the end-effector arrives at the target with a similar posture solution. Nevertheless, regarding the iterations, it's evident that the DLS method needs much more ones to arrive at the IK solution.

TABLE II
SOME TARGETS FOR THE ROBOT END EFFECTOR

Case N°	Target position (cm)			Target Orientation (rad)		
	x	y	z	θ	ϕ	ψ
1	1	-10	15	$-\pi\frac{3}{4}$	$-\pi\frac{1}{4}$	$\pi\frac{1}{4}$
2	5	0	12	$-\pi\frac{1}{4}$	$\pi\frac{2}{3}$	$\pi\frac{1}{2}$
3	3	-8	5	0	$\pi 1\frac{1}{4}$	π
4	0	-9	0	$\pi 1\frac{1}{5}$	0	$-\pi 1\frac{1}{3}$
5	-7	–5	6	$-\pi$	$-\pi\frac{1}{2}$	π
6	-4	-4	14	$\pi\frac{1}{5}$	π	0
7	-6	3	11	$\pi\frac{1}{2}$	$-\pi\frac{7}{8}$	$\pi 1\frac{1}{5}$
8	-2	7	7	$\pi\frac{2}{3}$	$-\pi\frac{2}{3}$	$\pi 1\frac{1}{3}$

For other targets, as is shown in Table III, the proposed method can perform the IK solution with a few numbers of iterations that require little time to accomplish, giving a RMSE below the threshold settled. Unlike the DLS method which in general took much more iterations and a longer time to get a result; even for case 6, it couldn't converge on a solution at the limit of 200 iterations.

TABLE III
COMPARISON OF METRICS BETWEEN THE PROPOSED METHOD AND THE DLS METHOD FOR SOME END-EFFECTOR TARGETS

N°	The Proposed Method			DLS Method		
	N Iterations	*Time (ms)*	*Final RMSE (E-3)*	*N Iterations*	*Time (ms)*	*Final RMSE (E-3)*
1	1	1.99	E-12	10	96.77	0.59
2	2.5	3.99	0.49	7	66.35	0.90
3	0.5	0.99	0.75	7	66.85	5.69E-2
4	1	1.99	1.5E-12	125	1100.2	0.98
5	1	2.03	3.1E-14	10	92.72	0.31
6	1	2.00	2.7E-12	199	1825.92	29.67
7	1	2.00	1.1E-12	6	56.88	0.12
8	1	1.99	1.9E-12	9	99.70	0.10

On the other hand, in Table IV can be found the joint values obtained from both methods of the first four cases shown in Table II. Comparing the results both methods can keep the joint values within their limits set. However, the difference is that the proposed method tends to take the values of the prismatic joints to extremes, which allows it to reach the target faster.

TABLE IV
COMPARISON OF THE ANGULAR VALUES OBTAINED FROM THE PROPOSED METHOD AND THE DLS METHOD FOR FOUR END-EFFECTOR TARGET CASES

Method	Joint	Case 1	Case 2	Case 3	Case 4
The Proposed Method	$q_1(^\circ)$	65.21	-1.50	54.97	0.00
	$q_2(^\circ)$	-4.59	-48.24	7.05	19.48
	$q_3(^\circ)$	-8.53E-7	-28.60	0.00	0.00
	$q_4(^\circ)$	61.14	97.26	73.17	75.12
	$q_5(^\circ)$	-91.02	-125.66	-34.17	0.00
	$q_6(^\circ)$	114.89	-113.11	133.73	121.40
	$q_7(^\circ)$	90.81	-109.35	38.50	120.00
	$l_1(cm)$	7.00	11.00	7.27	7.28
	$l_2(cm)$	9.09	7.00	11.00	11.00
	$l_3(cm)$	7.54	8.56	9.90	10.00
DLS Method	$q_1(^\circ)$	-13.84	-105.41	-1.56	-0.03
	$q_2(^\circ)$	21.79	53.36	35.51	32.80
	$q_3(^\circ)$	111.05	14.25	74.63	-179.95
	$q_4(^\circ)$	60.39	-87.54	84.45	-94.73
	$q_5(^\circ)$	-128.76	-10.22	-49.75	-180.00
	$q_6(^\circ)$	129.75	-78.83	128.35	88.47
	$q_7(^\circ)$	86.62	-30.77	67.75	120.02
	$l_1(cm)$	8.09	7.37	7.04	10.07
	$l_2(cm)$	7.45	7.02	7.26	7.13
	$l_3(cm)$	6.19	6.01	6.02	6.13

Now, regarding the IK problem with just target position (no orientation restriction at the end-effector). In Fig. 10 are shown the visual result of the IK problem's solution for the case 1 using the proposed method and the DLS method. It can be appreciated that whereas the method based on FABRIK could reach the target position in just a half of an iteration by just extending a single prismatic joint and rotating a hinge joint, the DLS method only reached the target position after five iterations where it was required to move many more joints of the robot than in the previous method.

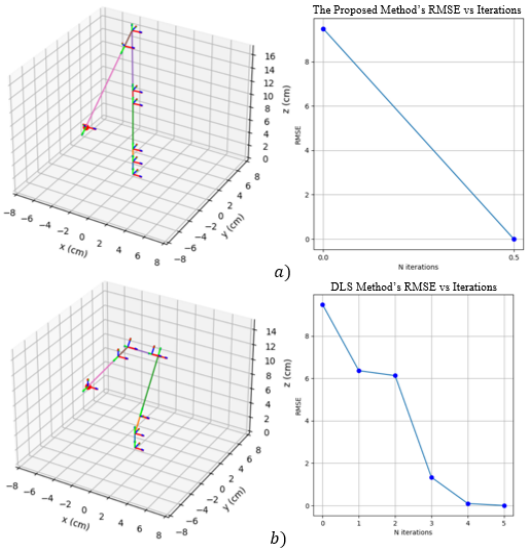

Fig. 10. Comparison between (a) the proposed method's result and (b) the DLS method's result for the same target position

Finally, the joint's values computed with our proposed method that solve the IK problem of the robot were exported to the Pybullet environment to validate those results using the URDF model of the robot. It can be appreciated in Fig. 11 that the robot's pose calculated by our method was recreated as expected in the Pybullet environment.

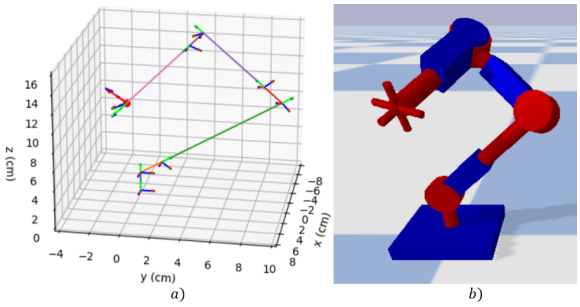

Fig. 11. Comparison between (a) our IK solution and (b) the validation in the Pybullet environment

V. Conclusions

After doing this work and having carried out the respective evaluations of the proposed method in comparison with the DLS method, the following conclusions were reached.

- The proposed method based on the FABRIK algorithm solves the IK problem of the test robot keeping the joint values within their limits, taking into account the particular joint type (pivot, hinge and prismatic joint) located in the kinematic chain.
- It was proved that the proposed method converges, reducing the RMSE, in a few step iterations in comparison to the DLS method. Also, it was proved that this proposal is computationally faster in time, arriving at the IK solution.
- The joint values obtained from the proposed method were exported to the robot's CAD model in the Pybullet environment, where it was validated that the robot's posture obtained in the simulation matched with the expected pose computed by the algorithm.
- The proposed algorithm works with position-only targets or targets with both position and orientation.

Finally, for future work, the proposed algorithm should be worked with the DH parameters, as well as to carry out a comparative study with other cutting-edge recursive methods such as metaheuristic methods and genetic algorithms. In addition to implementing the algorithm in an embedded system for a real redundant serial manipulator.

References

[1] Pasquale, C.; Stefano, C.; Lorenzo, S.; Bruno, S. Closed‐Loop Inverse Kinematics Schemes for Constrained Redundant Manipulators with Task Space Augmentation and Task Priority Strategy. Int. J. Robot. Res. 1991, 10,410–425.

[2] Kütük, M.; Taylan, M.; Canan, L. Forward and Inverse Kinematics Analysis of Denso Robot. In Proceedings of the International Symposium of Mechanism and Machine Science, Baku, Azerbaijan, 11–14 September 2017.

[3] Buss, Samuel. (2004). Introduction to inverse kinematics with Jacobian transpose, pseudoinverse and damped least squares methods. IEEE Transactions in Robotics and Automation. 17.

[4] Dantam, N. T. (2020). Robust and efficient forward, differential and inverse kinematics using dual quaternions. The International Journal of Robotics Research, 40(10–11), 1087–1105. https://doi.org/10.1177/0278364920931948

[5] Drexler, D. A., & Harmati, I. (2012). Joint Constrained Differential Inverse Kinematics Algorithm for Serial Manipulators. Periodica Polytechnica Electrical Engineering, 56(4), 95. https://doi.org/10.3311/ppee.7163

[6] Li-Chun Tommy Wang, Chih Cheng Chen, A combined optimization method for solving the inverse kinematics problems of mechanical manipulators, IEEE Transactions on Robotics and Automation 7 (4) (1991) 489–499.

[7] Jeff Lander, Making kine more flexible, Game Developer 5 (3) (1998)15–22

[8] Adrian A. Canutescu, Roland L. Dunbrack, Cyclic coordinate descent: a robotics algorithm for protein loop closure, Protein Science 12 (5) (2003) 963–972

[9] Aristidou, A., & Lasenby, J. (2011). FABRIK: A fast, iterative solver for the Inverse Kinematics problem. Graphical Models, 73(5), 243–260. https://doi.org/10.1016/j.gmod.2011.05.003

[10] Tenneti, R. A., & Sarkar, A. (2019). Implementation of modified FABRIK for robot manipulators. Proceedings of the Advances in Robotics 2019. https://doi.org/10.1145/3352593.3352605

[11] Santos, M. C., Molina, L., Carvalho, E. A. N., Freire, E. O., Carvalho, J. G. N., & Santos, P. C. (2021). FABRIK-R: An Extension Developed Based on FABRIK for Robotics Manipulators. IEEE Access, 9, 53423–53435. https://doi.org/10.1109/access.2021.3070693

[12] Santos, P. C., Freire, R. C. S., Carvalho, E. A. N., Molina, L., & Freire, E. O. (2020). M-FABRIK: A New Inverse Kinematics Approach to Mobile Manipulator Robots Based on FABRIK. IEEE Access, 8, 208836–208849. https://doi.org/10.1109/access.2020.3038424

[13] Tao, S., Tao, H., & Yang, Y. (2021). Extending FABRIK with Obstacle Avoidance for Solving the Inverse Kinematics Problem. Journal of Robotics, 2021, 1–10. https://doi.org/10.1155/2021/5568702

[14] Gangqi DONG, Panfeng HUANG, Yongjie WANG, Rongsheng LI, A modified forward and backward reaching inverse kinematics based incremental control for space manipulators, Chinese Journal of Aeronautics, 2021, ISSN 1000-9361, https://doi.org/10.1016/j.cja.2021.08.014.