Robot Manipulation and Mobility Final Project Report

Maksat Bekkuliyev, Saparkhan Kassymbekov. Azamat Kenzhekhan

Professor: Michele Folgheraiter

Nazarbayev University, Astana, Kazakhstan

November, 2023

Introduction

This final project report delves into the Direct and Inverse Kinematics analysis of the KUKA LBR 4+ serial manipulator. The initial phase involved the calculation of the forward kinematics utilizing the screw method. This method, grounded in the principles of Screw Theory, offers a systematic approach to describe the motion of spatial rigid body systems. The application of this technique enabled us to derive the mathematical representation of the end-effector pose.

Next, we solved the inverse kinematics. In the beginning, our team started from a manual derivation of the equations. However, it turned out to be too complicated to solve it by hand. Thats why, we harnessed the computational power of MATLAB to automate the inverse kinematics calculations.

To validate our theoretical findings, we conducted practical assessments using Coppelia Sim. This simulation served as a testing ground, allowing us to implement and evaluate the derived kinematic equations in a virtual representation of the KUKA LBR 4+. The outcomes of these simulations not only confirmed the validity of our analytical solutions but also provided insights into the practical implications of the derived kinematic models.

This report unfolds the intricacies of our journey through the Direct and Inverse Kinematics of the KUKA LBR 4+, blending theoretical derivations with computational implementations and practical validations.

Goals

- Formalize closed-form solutions for Direct and Inverse Kinematics (IK) of the KUKA LBR 4+ robot in CoppeliaSim.
- Utilize the Successive Screw Displacements Method to achieve these solutions.

Joint Configuration

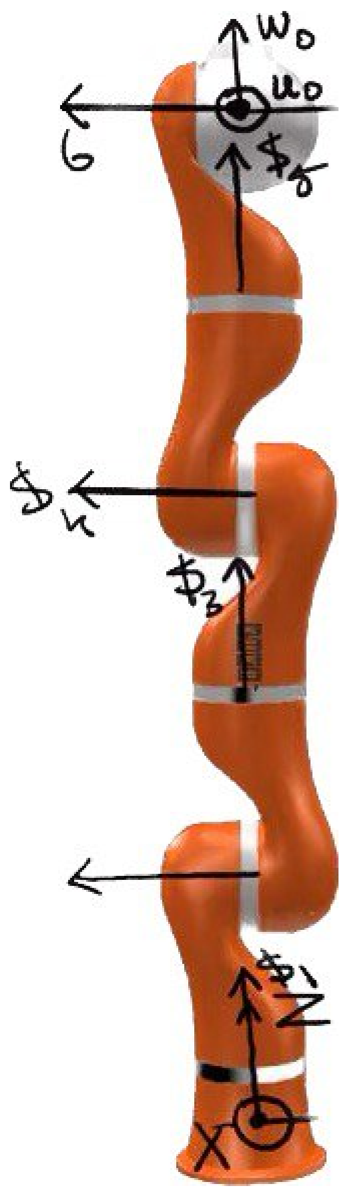

Figure 1. Joint configuration using screw-based method

Joint configuration has been configured according to CopeliaSim. Also the distances between joints were calculated from CoppeliaSim.

Forward Kinematics

$$^{0}A_1 = \begin{bmatrix} c\theta_1 & -s\theta_1 & 0 & 0 \\ s\theta_1 & c\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad ^{1}A_2 = \begin{bmatrix} c\theta_2 & 0 & -s\theta_2 & (d_0+d_1)s\theta_2 \\ 0 & 1 & 0 & 0 \\ s\theta_2 & 0 & c\theta_2 & (d_0+d_1)(1-c\theta_2) \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$^{2}A_3 = \begin{bmatrix} c\theta_3 & -s\theta_3 & 0 & 0 \\ s\theta_3 & c\theta_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad ^{3}A_4 = \begin{bmatrix} c\theta_4 & 0 & -s\theta_4 & s\theta_4(d_0+d_1+d_2+d_3) \\ 0 & 1 & 0 & 0 \\ s\theta_4 & 0 & c\theta_4 & (1-c\theta_4)d_0+d_1+d_2+d_3) \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$^{4}A_5 = \begin{bmatrix} c\theta_5 & -s\theta_5 & 0 & 0 \\ s\theta_5 & c\theta_5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad ^{5}A_6 = \begin{bmatrix} c\theta_6 & 0 & -s\theta_6 & s\theta_6(d_0+d_1+d_2+d_3+d_4+d_5) \\ 0 & 1 & 0 & 0 \\ s\theta_6 & 0 & c\theta_6 & (1-c\theta_6)(d_0+d_1+d_2+d_3+d_4+d_5) \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Figure 2. Homogeneous transform matrices

We calculate Forward Kinematics using Matlab, by multiplying A_h by q0.

```
q0 = [0, 0, d1+d2+d3+d4+d5]
```

Figure 3. Q0 is the position of q from the base-

Inverse Kinematics:

We made several attempts in order to calculate the inverse kinematics of the joint angles:

a) Calculation of θ_1 and θ_2:

In order to calculate the first two joint angles, the analytical method was attempted. We took the P_0 point on the joint 3 at $\left[0, 0,\ d_0 + d_1 + d_2 + d_3,\ 1\right]$. In order to transform this point into a P point, we need to perform 4 transformations as in Equation 1.

$$\left[p_x,\ p_y,\ p_z,\ 1\right]^T = A_1 \cdot\ A_2 \cdot A_3 \cdot A_4 \cdot \left[0, 0,\ d_0 + d_1 + d_2 + d_3,\ 1\right]^T \quad (1)$$

By multiplying both sides by A_1^{-1} we can obtain Equation 2:

$$A_1^{-1} \cdot \left[p_x,\ p_y,\ p_z,\ 1\right]^T = A_2 \cdot A_3 \cdot A_4 \cdot \left[0, 0,\ d_0 + d_1 + d_2 + d_3,\ 1\right]^T \quad (2)$$

Calculating both sides we obtain the following equations:

$$p_x \cdot cos(\theta_1) + p_y \cdot sin(\theta_1) = -sin(\theta_2) \cdot (d_2 + d_3) \quad (3)$$

$$p_y \cdot cos(\theta_1) - p_x \cdot sin(\theta_1) = 0 \quad (4)$$

$$p_z = d_0 + d_1 + (d_2 + d_3) \cdot cos(\theta_2) \quad (5)$$

From Equation 4, we can immediately calculate the joint angle θ_1 as in Equation 6:

$$\theta_1 = Atan2(p_y, p_x) \quad (6)$$

Combining Equations 3 and 5 we find the joint angle θ_2 as follows:

$$\theta_2 = Atan2\left(-\frac{p_x \cdot cos(\theta_1) + p_y \cdot sin(\theta_1)}{(d_2 + d_3)}, \frac{p_z - d_0 - d_1}{(d_2 + d_3)}\right) \quad (7)$$

This is the way we get the first two joint angles.

b) Calculation of θ_1, θ_2, θ_3, θ_4 and θ_5 using Matlab:

Following the method above requires extensive calculations, rather than that, we chose to use matlab. First we need 6 equations, for that reason we use 2 equations

$$A_1^{-1} \cdot [q_x, q_y, q_z, 1]^T = A_2 \cdot A_3 \cdot A_4 \cdot A_5 \cdot A_6 \cdot [0, 0, d_0 + d_1 + d_2 + d_3 + d_4 + d_5, 1]^T \quad (8)$$

$$R_1^{-1} \cdot [v_x, v_y, v_z]^T = R_2 \cdot R_3 \cdot R_4 \cdot R_5 \cdot [0, 1, 0]^T \quad (9)$$

In these equations the DH-decoupling method was used. In the second equation the v orientation was calculated by omitting the R_6, since R_6 doesn't affect the v orientation.

We have written a Matlab code for inverse kinematics, which computes the symbolic solution using the fsolve function, which uses 6 equations. The symbolic solution is then used to substitute the end effector pose, resulting in the determination of the corresponding joint angles.

```
function main()
    % Initial guess
    zg = zeros(6, 1);

    % Solve the system of equations
    z = fsolve(@myFunction, zg);

    % Display the solution
    disp('Solution:');
    disp(z);
end
```

Figure 4. Main function

```
function F = myFunction(z)
    syms t1 t2 t3 t4 t5 t6 vx vy vz qx qy d0 d1 d2 d3 d4 d5
    qz = sym('qz');

    % Extract variables from the input vector
    t1 = z(1);
    t2 = z(2);
    t3 = z(3);
    t4 = z(4);
    t5 = z(5);
    t6 = z(6);

    % Convert angles to radians
    t1_rad = deg2rad(t1);
    t2_rad = deg2rad(t2);
    t3_rad = deg2rad(t3);
    t4_rad = deg2rad(t4);
    t5_rad = deg2rad(t5);
    t6_rad = deg2rad(t6);

    % System of equations
    F(1) = vx*cos(t1_rad) + vy*sin(t1_rad) - (sin(t5_rad)*(sin(t2_rad)*sin(t4_rad)-cos(t2_rad)*cos(t3_rad)*cos(t4_rad))-cos(t2_rad)*cos(t5_rad)*sin(t3_rad));
    F(2) = vy*cos(t1_rad) - vx*sin(t1_rad) - (cos(t3_rad)*cos(t5_rad)-cos(t5_rad)*sin(t3_rad)*sin(t5_rad));
    F(3) = vz - (-sin(t5_rad)*(cos(t2_rad)*sin(t4_rad)-sin(t2_rad)*cos(t3_rad)*cos(t4_rad))-sin(t2_rad)*cos(t5_rad)*sin(t3_rad));

    F(4) = qx*cos(t1_rad) + qy*sin(t1_rad) - (-sin(t2_rad)*(d2+d3)-cos(t4_rad)*sin(t2_rad)*(d4+d5)-cos(t2_rad)*cos(t3_rad)*sin(t4_rad)*(d4+d5));
    F(5) = qy*cos(t1_rad) - qx*sin(t1_rad) - (-sin(t3_rad)*sin(t4_rad)*(d4+d5));
    F(6) = qz - (d0+d1+cos(t2_rad)*(d2+d3)+cos(t4_rad)*cos(t2_rad)*(d4+d5)-sin(t2_rad)*cos(t3_rad)*sin(t4_rad)*(d4+d5));

    % Substitute values for variables before converting to double
    F = subs(F, {vx, vy, vz, qx, qy, qz, d0, d1, d2, d3, d4, d5}, {0, -1, 0, 0.557, 0, 0.5, 0.051, 0.19998, 0.20002, 0.2, 0.2, 0.19});

    % Convert F to a double array
    F = double(F);
end
```

Figure 5. Function that finds the theta angles

```
Solution in degrees:
  101.2899
    2.7362
   76.7831
 -182.3351
    2.2488
         0
         0
```

Figure 6. Resultant theta angles

c) Calculation of θ_6:

After obtaining the first five joint angles, it was time to calculate the last joint angle θ_6. We calculated the W orientation by multiplying the W_0 in reference to the base frame (Equation 10).

$$\left[w_x,\ w_y,\ w_z\right]^T = R_1 \cdot R_2 \cdot R_3 \cdot R_4 \cdot R_5 \cdot R_6 \cdot [0, 0,\ 1]^T \quad (10)$$

Multiplying both sides by the transpose of the first five rotations we obtain the following:

$$\left(R_1 \cdot R_2 \cdot R_3 \cdot R_4 \cdot R_5\right)^T \cdot \left[w_x,\ w_y,\ w_z\right]^T = R_6 \cdot [0, 0,\ 1]^T \quad (11)$$

The result of the multiplication looks as follows:

$$k_1 = \ -\ sin\left(\theta_6\right) \quad (12)$$

$$k_2 = 0 \quad (13)$$

$$k_3 = \ cos\left(\theta_6\right) \quad (14)$$

The expressions $k_1,\ k_2\ and\ k_3$ have complicated forms and its from Matlab is shown below:

k1=wy*(sin(e)*(cos(a)*cos(c) - cos(b)*sin(a)*sin(c)) + cos(e)*(cos(d)*(cos(a)*sin(c) + cos(b)*cos(c)*sin(a)) - sin(a)*sin(b)*sin(d))) - wx*(sin(e)*(cos(c)*sin(a) + cos(a)*cos(b)*sin(c)) + cos(e)*(cos(d)*(sin(a)*sin(c) - cos(a)*cos(b)*cos(c)) + cos(a)*sin(b)*sin(d))) + wz*(cos(e)*(cos(b)*sin(d) + cos(c)*cos(d)*sin(b)) - sin(b)*sin(c)*sin(e))

k2 = wy*(cos(e)*(cos(a)*cos(c) - cos(b)*sin(a)*sin(c)) - sin(e)*(cos(d)*(cos(a)*sin(c) + cos(b)*cos(c)*sin(a)) - sin(a)*sin(b)*sin(d))) - wx*(cos(e)*(cos(c)*sin(a) + cos(a)*cos(b)*sin(c)) - sin(e)*(cos(d)*(sin(a)*sin(c) - cos(a)*cos(b)*cos(c)) + cos(a)*sin(b)*sin(d))) - wz*(sin(e)*(cos(b)*sin(d) + cos(c)*cos(d)*sin(b)) + cos(e)*sin(b)*sin(c))

k3 =
wx*(sin(d)*(sin(a)*sin(c) - cos(a)*cos(b)*cos(c)) - cos(a)*cos(d)*sin(b)) - wy*(sin(d)*(cos(a)*sin(c) + cos(b)*cos(c)*sin(a)) + cos(d)*sin(a)*sin(b)) + wz*(cos(b)*cos(d) - cos(c)*sin(b)*sin(d))

From Equations 12 and 14, we can get joint angle six:

$$\theta_6 = Atan2(-k_1, k_3)$$

d) Calculation of position P:
Position P is calculated for the purpose of delivering not the joint six to the desired position but the end effector. This P position is calculated using the formula:

$$^0\mathbf{p} = \overline{OP} = \begin{bmatrix} p_x \\ p_x \\ p_z \\ 1 \end{bmatrix} = \begin{bmatrix} q_x - d_6 w_x \\ q_x - d_6 w_y \\ q_z - d_6 w_z \\ 1 \end{bmatrix}.$$

Validation:

Simulation

We used Coppelia Sim software to test our inverse kinematics equations. To do so, we chose a desired coordinate reachable for the manipulator, and entered the values to our equations in MATLAB, which gave us the degrees of every joint. After that, we entered these configurations in CoppeliaSim and checked if the manipulator head achieved the desired pose.

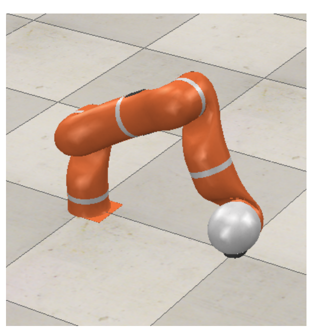

Figure 2. JManipulator reaching the target pose

Conclusion

In conclusion we have installed and configured the CopeliaSim simulation environment. We have used screw based methods coupled with Matlab to solve forward and inverse kinematics of KUKA LBR 4+ robots using. Moreover, we have tested the solutions with models available in CopeliaSim, so that the manipulator moves to the point we have given it.