Under-Actuated Manipulator Control (is hard)

& other revelations in manipulator kinematics

20.11.2025

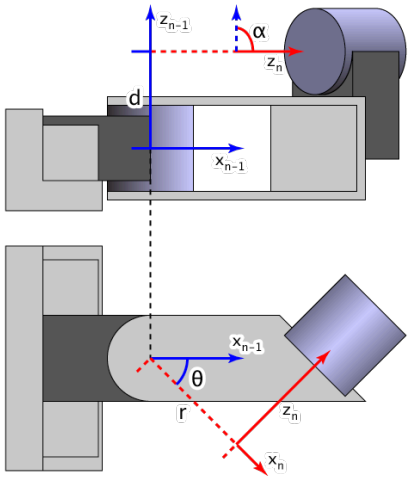
Vignesh Vembar & Krish Pandya

Contents

1 Representing the Arm

DH Parameters

Denavit Hartenberg (DH) parameters describe each joint using four values $a_i, \alpha_i, d_i, \theta_i$ and produce forward kinematics through a fixed transform pattern. They were popular because they give a compact, minimal representation.



Issues with DH Parameters

- Frame assignment rules are unintuitive and heavily constrained.
- Small mistakes in axis alignment break the model.
- Hard to insert tool frames, offsets, or calibration transforms.

Some people have tried to bridge the gap between modern methods like URDF and DH, but for this project we used a different approach.

ETS

- Elementary Transform Sequence (ETS) represents each joint as a separate transform.
- More intuitive frame assignment.
- Simply compose transforms to get forward kinematics.
- DH parameters might be only 4 parameters per joint, but when ETS has only 6 parameters per joint and is easier to work with, the tradeoff is not worth it.

$$ {}^0T_e = \prod_{\mathrm{i}=1}^{n} E_{i(\eta_i)} $$

> This tutorial [...] deliberately avoids the commonly-used Denavit-Hartenberg parameters which we feel confound pedagogy.
>
> — arxiv.org/abs/2207.01796

2 Metrics to Visualize Reachability

Manipulability

$$\omega = \sqrt{\det(J * J^{\perp})}$$

But for our 5-DOF arm, `J.shape() == (6, 5)`, so $J * J^{\perp}$ is not full rank and $\det(J * J^{\perp}) = 0$ always.

So instead, we use the singular values of `J` to define manipulability as:

$$\omega = \sqrt{\prod_{i=1}^{5} \sigma_i}$$

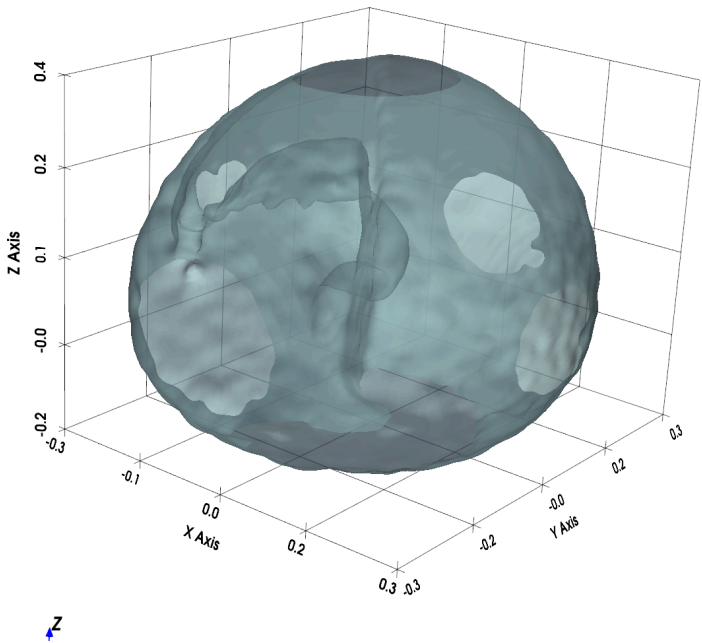
Condition

Another useful metric is the condition number of the Jacobian, defined as the ratio of the maximum and minimum singular values:

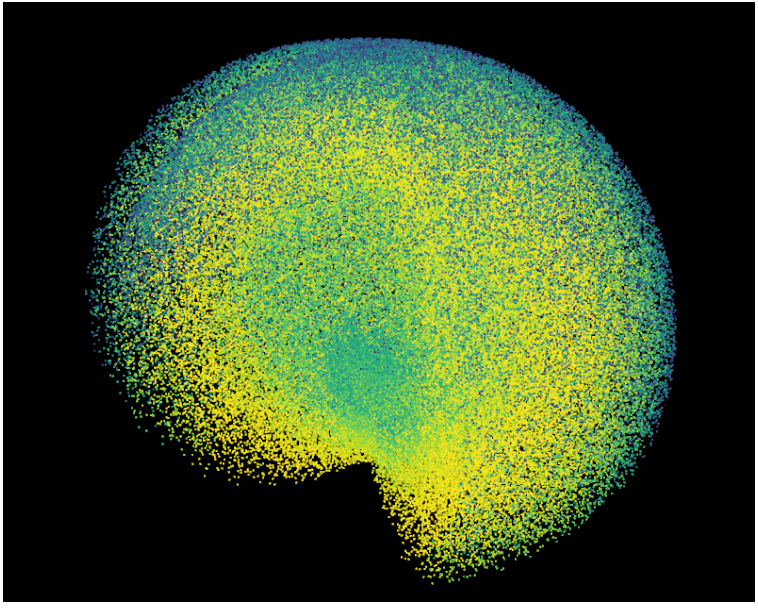
$$\kappa = \frac{\sigma_{\max}}{\sigma_{\min}}$$

$\kappa \gg 1$ implies the arm is close to a singularity, while $\kappa \approx 1$ implies good dexterity.

Visualizing the Reachable Workspace

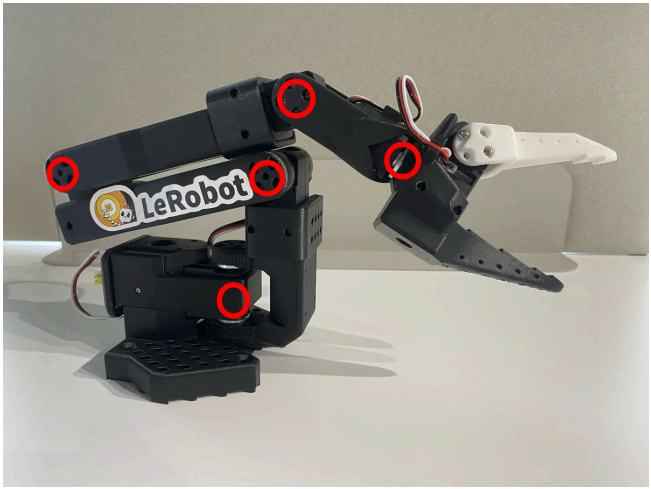


Visualizing the Reachable Workspace (ii)



3 The Problem and its Naive Solution

5-DOF, What are we missing?



There is no end-effector yaw.

Feasibility-Projected IK

A 5-DOF arm moves on a 5D manifold in 6D task space. We project the desired pose onto this feasible manifold and solve for the corresponding joint update.

Projection step: The achievable motion is the projection of the desired motion onto the range of the Jacobian:

$$\Delta x_{\text{achievable}} = JJ^{+}\Delta x$$

Joint update: We seek Δq satisfying $J\Delta q \approx \Delta x_{\text{achievable}}$ via damped least-squares:

$$\Delta q = \arg\min_{\Delta q} \| \, W_x(J\Delta q - \Delta x_{\text{achievable}}) \, \|^2 + \lambda^2 \, \|\Delta q\|^2$$

Derivation: Expanding the objective:

$$L(\Delta q) = (J\Delta q - \Delta x_{\text{achievable}})^{\top} W_x^{\top} W_x (J\Delta q - \Delta x_{\text{achievable}}) + \lambda^2 \Delta q^{\top} \Delta q$$

Taking the derivative with respect to Δq and setting to zero:

$$\frac{\partial L}{\partial \Delta q} = 2J^{\top} W_x^{\top} W_x (J\Delta q - \Delta x_{\text{achievable}}) + 2\lambda^2 \Delta q = 0$$

Feasibility-Projected IK (ii)

Rearranging:

$$J^\top W_x^\top W_x J \Delta q + \lambda^2 \Delta q = J^\top W_x^\top W_x \Delta x_{\text{achievable}}$$

Solution: Factoring and solving for Δq:

$$(J^\top W_x^\top W_x J + \lambda^2 I)\Delta q = J^\top W_x^\top W_x \Delta x_{\text{achievable}}$$

Substituting $\Delta x_{\text{achievable}} = J J^+ \Delta x$:

$$\Delta q = \left(J^\top W_x^\top W_x J + \lambda^2 I\right)^{-1} J^\top W_x^\top W_x J J^+ \Delta x$$

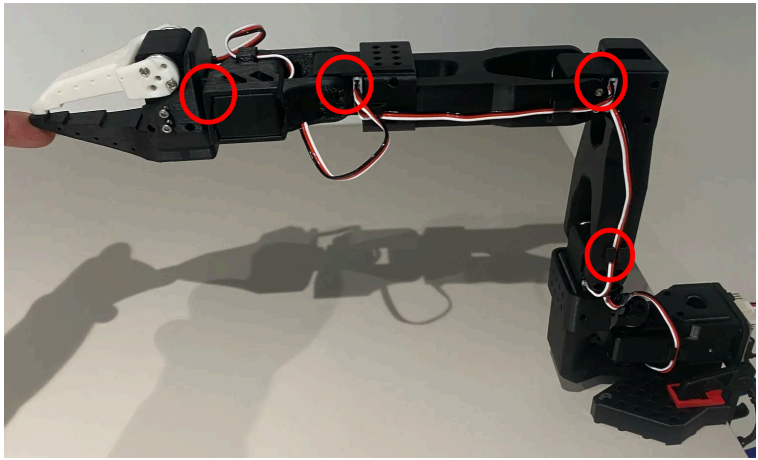
This directly computes joint motions that respect the kinematic constraints while minimizing weighted task-space error.

Where it Fails

- Because the yaw axis is missing, when the end effector moves off the 2D plane defined by the first joint axis, the IK solver continuously fails to find a solution.
- The “projected” solution is always just on (or very close to) the 2D plane.

4 The Workaround that Works

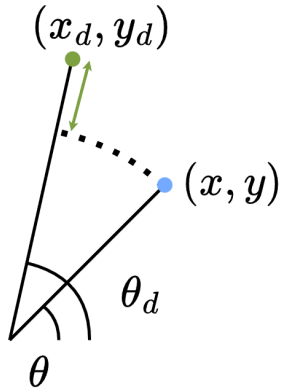
Fool the IK solver



The solver never gets end effector y-axis position or yaw updates, and never updates the first joint angle.

This way the solver doesnt continuously fail when we leave the 2D plane.

Handling the y-axis position



$$\theta_{\text{update}} = \arctan(y_d, x_d) - \arctan(y, x)$$

$$x_{\text{update}} = \sqrt{y'^2 + x'^2} - \sqrt{y^2 + x^2}$$

Limitations

- Only works for position control, not full 6D pose control.
- End effector yaw is completely uncontrollable, and pitch and roll are constrained to the rotating 2D plane.

5 Demo