BIPEDAL MOBILE ROBOT

PROJECT DESCRIPTION

The Bipedal Mobile Robot project served as a valuable learning experience in managing and iteratively improving a complex system. The robot features limbs with four revolute joints providing one degree of freedom each, and two revolute joints offering two degrees of freedom each. Throughout the build and testing phases, the primary goal was to apply theoretical knowledge in a practical setting.

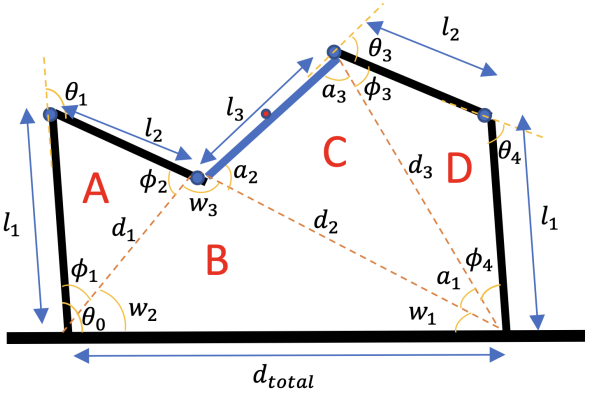
Concepts such as Forward Kinematics, Inverse Kinematics, and controller design, extensively studied in theory, were brought to life through this hands-on project.

Through the following sections, we present our work and what we have learned!

FORWARD KINEMATICS

In order to calculate locations of each members given joint angles $[\theta_1, \theta_2, \theta_3, \theta_4]$ there are multiple methods, we used Geometric Analysis and Denavit Hartenberg:

Geometric Analysis



This method involves the use of the geometry of the members and trigonometry to determine the required joint angles needed.

- First we divide the robot area into small triangles A,B,C,D.
- Then using sine and cosine rules, starting from A and D and known angles θ_1 and θ_4 we calculate all features.
- From triangle D we can get triangle C features.
- Finally, using both A and C we can calculate B, and use all the defining geometry to calculate joints and robot's pose.

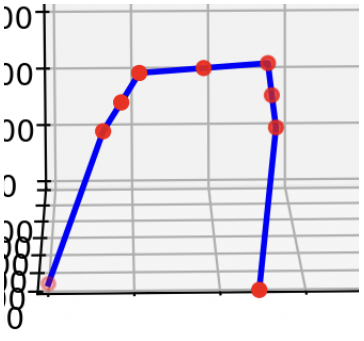
In summary, given angles $[\theta_1, \theta_2, \theta_3, \theta_4]$, all geometric features are calculated for this robot.

Denavit Hartenberg (D-H Method)

In robotics, a homogeneous transformation matrix combines rotation and translation into a single matrix to convert coordinates from one frame to another, typically from a local frame to a world or base frame. The Denavit-Hartenberg (DH) method is a standardized convention for modeling robot kinematics, where each joint's transformation is described using four parameters (θ, d, a, α) to systematically build the full transformation chain. We used these descriptions to visualize joint locations in 3D and implemented motion planning, given input angles $[\theta_1, \theta_2, \theta_3, \theta_4]$.

- Without dealing with trigonometric functions and purely using sine/cosine terms in linear matrix operations, this is another method to calculate joint locations from angles.

$$\mathbf{p}_{\text{world}} = \underbrace{\begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix}}_{\text{Transformation}} \cdot \underbrace{\begin{bmatrix} \mathbf{p}_{\text{local}} \\ 1 \end{bmatrix}}_{\text{Local Coordinate}} = \begin{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} & \begin{bmatrix} x_{\text{trans}} \\ y_{\text{trans}} \end{bmatrix} \\ \begin{bmatrix} 0 & 0 \end{bmatrix} & 1 \end{bmatrix} \cdot \begin{bmatrix} x_{\text{local}} \\ y_{\text{local}} \\ 1 \end{bmatrix}$$



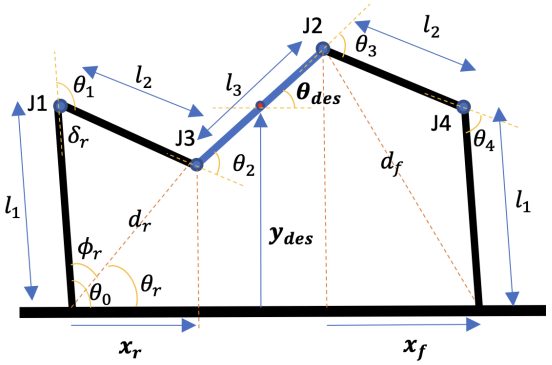
INVERSE KINEMATICS

In order to calculate joint angles required to get to positions $[y_d, x_r, x_f, \theta_d]$, we use geometric analysis:

$x_{j2} = x_r$ [1] $\quad y_{j2} = y_{des} - \frac{l_3}{2} sin\,\theta_{des}$ [2]

plug eq. 1 and 2: $d_r = \sqrt{x_{j2}^2 + y_{j2}^2}$ [3], $\theta_r = tan^{-1}\left(\frac{y_{j2}}{x_{j2}}\right)$ [4]

$\phi_r = cos^{-1}\left(\frac{l_2^2 - d_r^2 - l_1^2}{-2 d_r l_1}\right)$ [5] and $\delta_r = cos^{-1}\left(\frac{d_r^2 - l_1^2 - l_2^2}{-2 l_1 l_2}\right)$ [6]



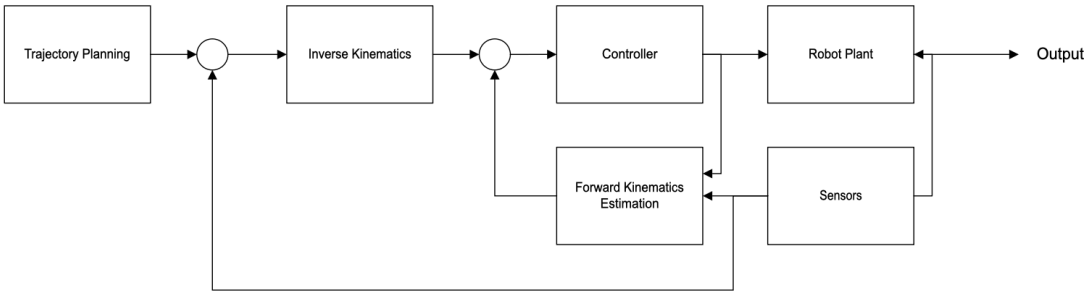
Plug in eq. 4-5 to solve θ_0 , eq. 6 to solve θ_1

$\theta_0 = \theta_r \pm \phi_r$ [7] $\quad \theta_1 = \pi - \delta_r$ [8]

finally use θ_0 and θ_1 to find: $\theta_2 = \theta_{des} - \theta_1 - \theta_0$ [9]

Use the same idea on the front side (right side in the front view) and solve for θ_3 and θ_4

SENSORS & FEEDBACK CONTROL



Here we have control logic of the robot for desired input $[y_d, x_r, x_f, \theta_d]$. Given desired input, Inverse Kinematics calculate input angles $[\theta_1, \theta_2, \theta_3, \theta_4]$ and after going through PID logic, servo position is set as input for the Robot Plant. The output is measured by the sensors, which with Forward Kinematics, unobserved states are estimated and input angles are used as feedback to controller logic. In addition, measured states are cascaded to trajectory planning such that a new trajectory is sent to achieve the full motion pre-determined.

ABOUT US

Chuba Oraka

Chuba is a self-taught programmer with a background in Civil Engineering. He has a passion for acquiring new skills through a project-based approach and has worked on projects ranging from Structural design and web design to robotics.

chubaoraka.github.io/assets/bin/Résumé.pdf

Kaan Beydüz

Kaan has a mechanical engineering background with a focus on robotics, and is currently planning to pursue a PhD. Currently working at an analog circuit design startup to validate customer specifications through Machine Learning.

www.kaanbeyduz.com