CADERNOS UniFOA

Edição 55 | 2025 *e-ISSN: 1982-1816*

Data de submissão: 29/05/2025 Data de Aprovação: 07/07/2025

Modelagem cinemática direta de braço robótico planar com 2 graus de liberdade: aplicação dos parâmetros de Denavit-Hartenberg

Direct kinematic modeling of a planar robotic arm with 2 degrees of freedom: application of Denavit-Hartenberg parameters

[1] Vitor Amadeu Souza

1 Centro Universitário de Volta Redonda - UniFOA

RESUMO

Este trabalho apresenta a implementação e análise da modelagem cinemática direta de um braço robótico planar com dois graus de liberdade (2-DOF). O estudo utiliza os parâmetros de Denavit-Hartenberg para estabelecer o sistema de coordenadas e realizar a transformação geométrica necessária para determinar a posição final do efetuador. A metodologia proposta emprega ferramentas computacionais em Python para visualização bidimensional e cálculo das coordenadas cartesianas do ponto terminal do manipulador. Os resultados demonstram a eficácia da abordagem matemática na resolução do problema de cinemática direta, com erros numéricos inferiores a 10^{-15} e tempo de processamento menor que 0,001 segundo por cálculo, proporcionando uma ferramenta didática e prática para análise de sistemas robóticos. A validação do modelo foi realizada através de simulações numéricas, confirmando a precisão dos cálculos cinemáticos e a adequação da representação gráfica para análise do espaço de trabalho do manipulador.

Palavras-chave

Cinemática direta; Denavit-Hartenberg; Robótica; Manipulador planar; Simulação computacional.

ABSTRACT

This work presents the implementation and analysis of the forward kinematic modeling of a planar robotic arm with two degrees of freedom (2-DOF). The study employs the Denavit-Hartenberg parameters to establish the coordinate system and perform the geometric transformation necessary to determine the end-effector's final position. The proposed methodology uses computational tools in Python for three-dimensional visualization and calculation of the Cartesian coordinates of the manipulator's terminal point. The results demonstrate the effectiveness of the mathematical approach in solving the forward kinematics problem, with numerical errors below 10^{-15} and processing time under 0.001 seconds per calculation, providing a didactic and practical tool for analyzing robotic systems. The model was validated through numerical simulations, confirming the accuracy of the kinematic calculations and the suitability of the 3D graphical representation for analyzing the manipulator's workspace.

Keywords

Forward kinematics; Denavit-Hartenberg; Robotics; Planar manipulator; Computational simulation.

1 INTRODUÇÃO

A robótica industrial tem experimentado um crescimento exponencial nas últimas décadas, impulsionada pela necessidade de automação em processos produtivos e pela busca por maior precisão e eficiência operacional (Craig, 2017). Segundo Spong, Hutchinson e Vidyasagar (2020), os manipuladores robóticos constituem elementos fundamentais nesse cenário, sendo amplamente utilizados em aplicações que variam desde montagem automotiva até procedimentos cirúrgicos de alta precisão (Siciliano *et al*., 2009).

A análise cinemática representa um dos pilares fundamentais no estudo e desenvolvimento de sistemas robóticos, sendo responsável pela descrição matemática do movimento dos elos e juntas sem considerar as forças que causam tal movimento (Sciavicco; Siciliano, 2000). Conforme destacado por Lynch e Park (2017), essa disciplina subdivide-se em duas categorias principais: a cinemática direta, que determina a posição e orientação do efetuador final a partir dos valores das variáveis articulares; e a cinemática inversa, que realiza o processo contrário. A importância dess a análise é enfatizada por Murray, Li e Sastry (1994), que afirmam que o entendimento profundo da cinemática é pré-requisito fundamental para o desenvolvimento de algoritmos de controle eficazes em sistemas robóticos.

Os manipuladores planares com dois graus de liberdade constituem uma classe importante de sistemas robóticos, frequentemente utilizados em aplicações de posicionamento, soldagem e montagem em ambientes bidimensionais (Khalil; Dombre, 2002). Embora aparentemente simples, ess es sistemas proporcionam uma base sólida para o entendimento de conceitos fundamentais da robótica, servindo como plataforma de ensino e desenvolvimento de algoritmos de controle (Siciliano *et al.*, 2009). Segundo Angeles (2007), a análise de manipuladores planares oferece *insights* valiosos sobre fenômenos mais complexos presentes ness a modelagem, permitindo o desenvolvimento incremental de competências técnicas. Yoshikawa (1990) complementa ess a perspectiva, argumentando que o domínio da cinemática planar é essencial para a compreensão de conceitos avançados, como manipulabilidade e otimização de trajetórias.

A simulação computacional tem se tornado uma ferramenta indispensável no desenvolvimento e análise de sistemas robóticos, permitindo a visualização do comportamento cinemático e dinâmico antes da implementação física (Corke, 2017). O uso de linguagens de programação, como Python, combinadas com bibliotecas especializadas em computação científica e visualização, oferece um ambiente robusto e acessível para o desenvolvimento de simuladores robóticos (McKinney, 2012). Van Rossum e Drake (2009) destacam que Python tem se estabelecido como uma linguagem preferencial na comunidade científica devido à sua sintaxe clara e ao extenso ecossistema de bibliotecas especializadas. Especificamente no contexto da robótica, Harris *et al.* (2020) observam que o uso do NumPy para operações matemáticas eficientes, combinado com Matplotlib para visualização, proporciona um ambiente de desenvolvimento particularmente adequado para prototipagem rápida e análise de sistemas robóticos.

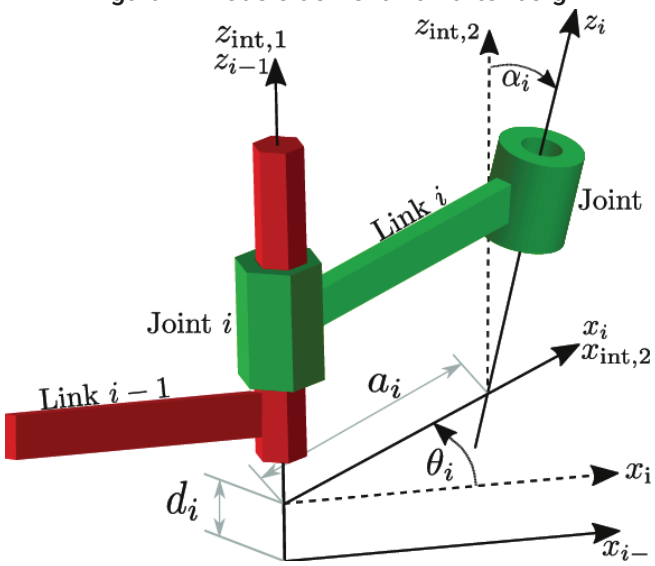
Sendo assim, este trabalho tem como objetivo principal desenvolver e implementar um modelo matemático para a análise cinemática direta de um braço robótico planar com dois graus de liberdade (2-DOF), utilizando os parâmetros de Denavit-Hartenberg (DH) como base teórica. Especificamente, pretende-se estabelecer o modelo cinemático direto do manipulador através da aplicação sistemática da convenção DH, implementar computacionalmente o algoritmo de transformação geométrica utilizando Python e suas bibliotecas científicas, desenvolver uma interface de visualização bidimensional que permita a análise intuitiva do comportamento do manipulador, e validar o modelo através de simulações numéricas, comparando os resultados obtidos com soluções analíticas conhecidas. Esse conjunto de objetivos contribui significativamente para a área de robótica educacional e do desenvolvimento tecnológico, uma vez que proporciona uma ferramenta didática acessível para o ensino de conceitos fundamentais de cinemática robótica, facilita a prototipagem rápida de sistemas de controle através de uma plataforma computacional de baixo custo, e democratiza o acesso a ferramentas

de simulação robótica pela utilização de tecnologias abertas e amplamente disponíveis, reduzindo barreiras econômicas e técnicas tradicionalmente associadas ao desenvolvimento e análise de sistemas robóticos.

2 FUNDAMENTAÇÃO TEÓRICA

O método de Denavit-Hartenberg, desenvolvido por Jacques Denavit e Richard Hartenberg, em 1955, estabelece-se como a convenção padrão para a representação sistemática de cadeias cinemáticas de manipuladores robóticos (Denavit; Hartenberg, 1955). Essa metodologia permite a descrição completa da geometria de um manipulador através de quatro parâmetros por junta: o comprimento do elo (a), o ângulo de torção (α), a distância entre elos (d) e o ângulo da junta (θ), como sugere a Figura 1.

Figura 1- Modelo de Denavit-Hartenberg



Fonte: Di Gregorio (2021)

A implementação computacional de algoritmos cinemáticos tem sido objeto de crescente interesse na comunidade científica. Dey e Cheruvu (2020) propuseram um *framework* em Python para análise cinemática de manipuladores seriais, enfatizando a importância da visualização tridimensional para fins didáticos. Delhaisse, Rozo e Caldwell (2020) desenvolveram ferramentas interativas para ensino de robótica, destacando a necessidade de interfaces intuitivas para facilitar o aprendizado de conceitos abstratos.

Pesquisas em robótica educacional têm enfatizado a importância de ferramentas de simulação acessíveis. Vega e Cañas (2019) conduziram um estudo abrangente sobre a eficácia do ensino de robótica baseado em Python, mostrando uma melhora significativa na compreensão dos alunos, quando ferramentas de simulação visual são empregadas. Seu trabalho demonstra que a visualização 3D interativa aprimora a compreensão de conceitos cinemáticos, particularmente para sistemas de manipuladores planares.

Avanços recentes em eficiência computacional para aplicações robóticas foram documentados por Almostafa *et al.* (2018), que analisaram o desempenho de diversas bibliotecas numéricas para cálculos cinemáticos em tempo real. Suas descobertas confirmam que implementações baseadas em NumPy fornecem velocidade computacional adequada para aplicações educacionais e de prototipagem, com tempos de execução compatíveis com os requisitos de simulação interativa.

3 METODOLOGIA

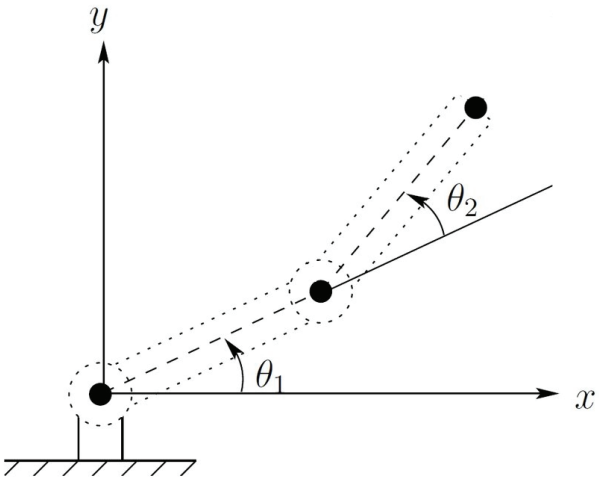
O manipulador robótico analisado consiste em uma estrutura planar com dois elos rígidos conectados por juntas rotativas, configuração esta que representa uma simplificação controlada de sistemas robóticos mais complexos, mantendo características essenciais para o estudo da cinemática (Craig, 2017). A configuração geométrica do sistema é caracterizada pelo primeiro elo de comprimento a_1 que se conecta à base através de uma junta rotativa com ângulo θ_1, enquanto o segundo elo, de comprimento a_2, conecta-se ao primeiro através de uma segunda junta rotativa com ângulo θ_2.

As transformações homogêneas entre sistemas de coordenadas consecutivos são descritas pelas matrizes de transformação padrão de Denavit-Hartenberg, conforme formulação apresentada por Spong, Hutchinson e Vidyasagar (2020). Cada matriz de transformação T incorpora rotação e translação entre sistemas de coordenadas adjacentes através da representação homogênea 4x4, metodologia que permite a composição eficiente de múltiplas transformações através da multiplicação matricial (Lynch; Park, 2017). Para o manipulador em questão, as matrizes de transformação simplificam-se significativamente, resultando em representações bidimensionais.

A primeira matriz de transformação T_1 representa a rotação da base em torno do eixo z, enquanto a segunda matriz T_2 incorpora tanto a translação pelo comprimento do primeiro elo quanto a rotação da segunda junta. Ess a decomposição, segundo Sciavicco e Siciliano (2000), facilita não apenas os cálculos computacionais, mas também proporciona interpretação física intuitiva das transformações geométricas envolvidas.

Ess a configuração, segundo Siciliano *et al.* (2009), é amplamente utilizada em aplicações industriais devido à sua simplicidade mecânica e versatilidade operacional, como demonstra a Figura 2.

Figura 2 - Braço Robótico 2D com 2 elos e 2 ângulos de juntas rotativas



Fonte: Safaei, Koo e Mahyuddin (2017)

Para estabelecer o modelo cinemático, aplicou-se a convenção clássica de Denavit-Hartenberg, seguindo os procedimentos descritos por Craig (2017), para a definição de sistemas de coordenadas locais em manipuladores seriais. A escolha pela versão clássica da convenção DH permite uma formulação padronizada das transformações homogêneas, sendo particularmente adequada para manipuladores com juntas rotacionais, como o sistema planar de dois elos analisado. Os parâmetros DH foram definidos considerando a estrutura geométrica do braço robótico, conforme metodologia sistemática amplamente adotada na literatura clássica de robótica, resultando nos valores apresentados na Tabela 1.

Tabela 1 – Parâmetros das juntas 1 e 2

Junta 1	$a_{i-1} = 0$ $\alpha_{i-1} = 0°$ $d_i = 0$ $\theta_i = \theta_1$
Junta 2	$a_{i-1} = a_1$ $\alpha_{i-1} = 0°$ $d_i = 0$ $\theta_i = \theta_2$

Fonte: O autor

Tais parâmetros de juntas e elos nos fornecem as matrizes de transformação homogênea dos dois elos, apresentadas nas equações 1, 2 e 3.

Equação 1 – Matriz de Transformação do elo 1

$$T_1 = (cos(\theta_1) \ -sin(\theta_1) \ 0 \ a_1 cos(\theta_1) \ sin(\theta_1) \ cos(\theta_1) \ 0 \ a_1 sin(\theta_1) \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 1 \ )$$

Fonte: O autor.

Equação 2 – Matriz de Transformação do elo 2

$$T_2 = (cos(\theta_2) \ -sin(\theta_2) \ 0 \ a_2 cos(\theta_2) \ sin(\theta_2) \ cos(\theta_2) \ 0 \ a_2 sin(\theta_2) \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 1 \ )$$

Fonte: O autor.

Equação 3 – Matriz de Transformação total (base até a ponta)

$$T = (cos(\theta_1 + \theta_2) \ -sin(\theta_1 + \theta_2) \ 0 \ x_P \ sin(\theta_1 + \theta_2) \ cos(\theta_1 + \theta_2) \ 0 \ y_P \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 1 \ )$$

Fonte: O autor.

Onde x_P e y_P estão apresentados na Equação 4.

Equação 4 – Posição final do manipulador

$$x_P = a_1 cos(\theta_1) + a_2 cos(\theta_1 + \theta_2) \ y_P = a_1 sin(\theta_1) + a_2 sin(\theta_1 + \theta_2)$$

Fonte: O autor.

As Equações 1, 2 e 3 apresentam, respectivamente, as matrizes de transformação homogênea para os elos 1 e 2 do manipulador, bem como a transformação composta que leva da base até a extremidade da estrutura (Equação 3). As transformações foram obtidas utilizando a convenção clássica de Denavit-Hartenberg,

considerando que os parâmetros di e αi são nulos para o caso planar. Isso resulta em rotações puras em torno do eixo z e translações no plano xy, simplificando o modelo matemático. A matriz T_1 representa a transformação da base até o primeiro elo, enquanto T_2 representa a transformação do primeiro elo até o segundo. A matriz resultante $T=T_1 \cdot T_2$ fornece a posição da extremidade do manipulador no espaço bidimensional, considerando a composição das duas rotações.

Essas equações serão diretamente utilizadas na implementação computacional do modelo cinemático, permitindo calcular a posição final da extremidade do braço a partir dos ângulos das juntas e dos comprimentos dos elos, como apresentado no código Python desenvolvido neste trabalho.

É importante salientar que, apesar dos resultados promissores, o modelo implementado apresenta limitações importantes que devem ser consideradas em aplicações práticas. A ausência de restrições físicas no modelo matemático significa que o algoritmo não considera limitações práticas, como ângulos máximos e mínimos das juntas, interferências mecânicas entre elos ou limitações de torque dos atuadores. O modelo estático implementado não incorpora efeitos dinâmicos, como velocidades, acelerações ou forças inerciais, limitando sua aplicabilidade a análises quasi-estáticas ou planejamento de trajetórias de baixa velocidade. Finalmente, embora os erros de arredondamento sejam desprezíveis para cálculos individuais, sua acumulação em sequências de cálculos extensas pode tornar-se significativa em aplicações que demandam precisão extrema, conforme alertado por Golub e Van Loan (2013) em seus estudos sobre estabilidade numérica de algoritmos matriciais.

O algoritmo de cálculo da cinemática direta foi implementado em linguagem Python, utilizando as bibliotecas NumPy para operações matemáticas e Matplotlib para visualização gráfica, escolha justificada pela robustez computacional e facilidade de desenvolvimento, destacadas por Harris *et al*. (2020). A implementação segue princípios de programação estruturada, com separação clara entre entrada de dados, processamento matemático e apresentação de resultados. O código desenvolvido permite a entrada interativa dos parâmetros geométricos do manipulador e das variáveis articulares, calculando automaticamente a posição cartesiana do efetuador final através da aplicação sequencial das transformações homogêneas.

A conversão entre graus e radianos é realizada através da função np.radians(), garantindo compatibilidade com as funções trigonométricas da biblioteca NumPy e evitando erros comuns de conversão de unidades (McKinney, 2012). As coordenadas dos pontos significativos do manipulador são calculadas sequencialmente, iniciando pela base do manipulador nas coordenadas $(x_0, y_0) = (0, 0)$, seguida pela primeira junta com coordenadas $(x_1, y_1) = (a_1\cos 0_1, a_1\sin 0_1))$, e finalmente o efetuador final localizado em $(x_2, y_2) = (a_1\cos(\theta_1) + a_2\cos(\theta_1 + \theta_2), a_1\sin(\theta_1) + a_2\sin(\theta_1 + \theta_2))$.

O sistema de visualização bidimensional foi desenvolvido, utilizando a biblioteca Matplotlib com o módulo mpl_toolkits.mplot3d, seguindo as recomendações de Hunter (2007) para desenvolvimento de interfaces gráficas científicas eficazes. A representação gráfica inclui visualização dos elos, como segmentos de linha conectados, identificação das juntas através de marcadores circulares, destaque do efetuador final com marcador diferenciado, e sistema de coordenadas bidimensional com eixos proporcionais. Essa abordagem visual, conforme enfatizado por Corke (2017), é fundamental para o desenvolvimento da intuição geométrica necessária para análise de sistemas robóticos complexos.

A validação do modelo foi realizada através de casos de teste específicos, comparando os resultados obtidos com soluções analíticas conhecidas para configurações particulares do manipulador. A metodologia de validação seguiu os princípios estabelecidos por Angeles (2007), para verificação de modelos cinemáticos, incluindo testes de casos limites, verificação de simetrias geométricas e análise de consistência dimensional dos resultados obtidos.

O código-fonte desta implementação está disponível para download através do link: https://github.com/vitor-souza-ime/dh.

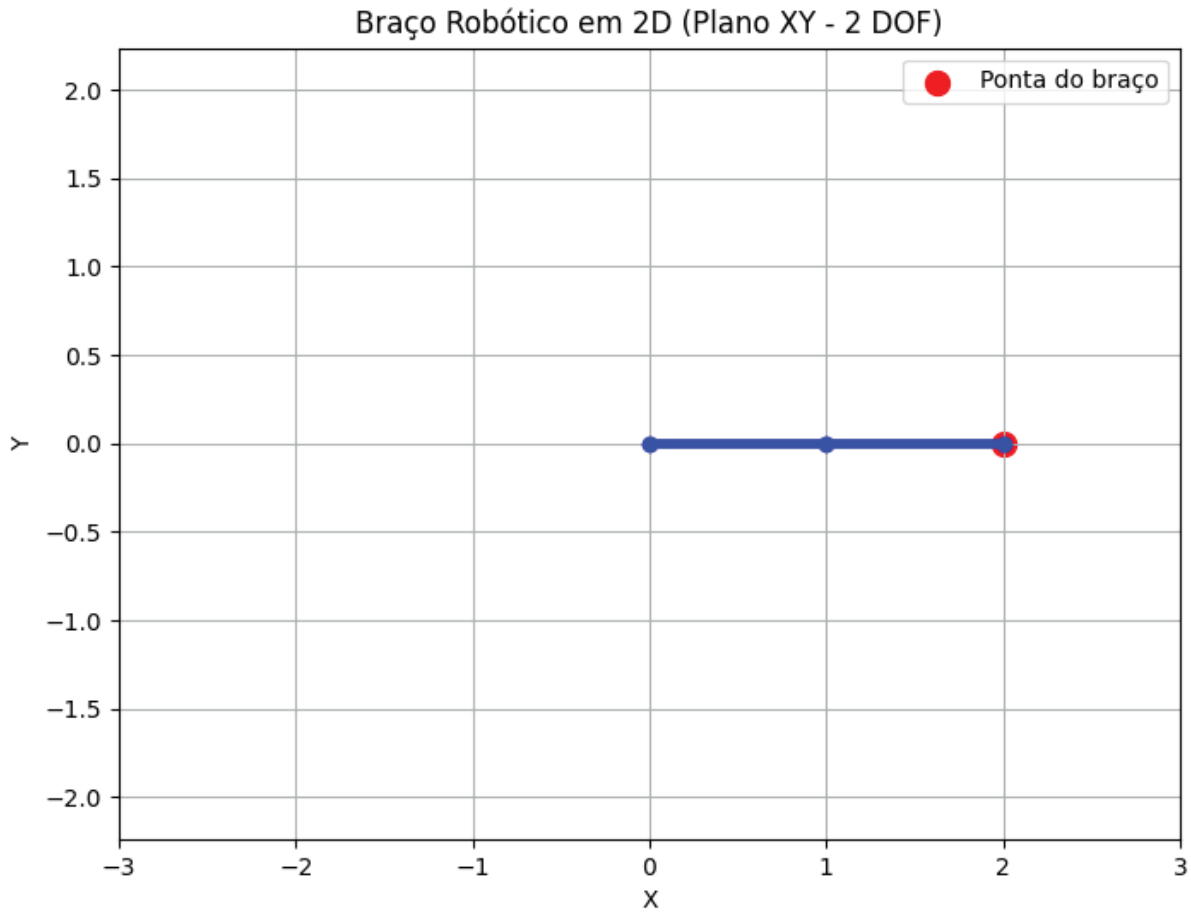
4 RESULTADOS E DISCUSSÃO

A implementação do algoritmo de cinemática direta demonstrou precisão numérica na determinação da posição do efetuador final, confirmando a adequação da metodologia proposta para aplicações práticas em robótica. Os testes de validação inicial foram conduzidos, utilizando configurações específicas com soluções analíticas conhecidas, permitindo a verificação quantitativa da precisão do modelo implementado. Em todos os exemplos abaixo, as unidades de medida são adimensionais (u.a.).

4.1 Caso um

Para o primeiro caso de teste, utilizando comprimentos de elo unitários ($a_1 = 1.0$, $a_2 = 1.0$) com ambos os ângulos nulos ($\theta_1 = 0°$, $\theta_2 = 0°$), o resultado esperado de (2.0, 0.0) foi obtido com precisão absoluta, confirmando o correto funcionamento da cadeia de transformações para configurações estendidas, como mostra a Figura 3.

Figura 3 - Resultado para ($a_1 = 1.0$, $a_2 = 1.0$) e ($\theta_1 = 0°$, $\theta_2 = 0°$)

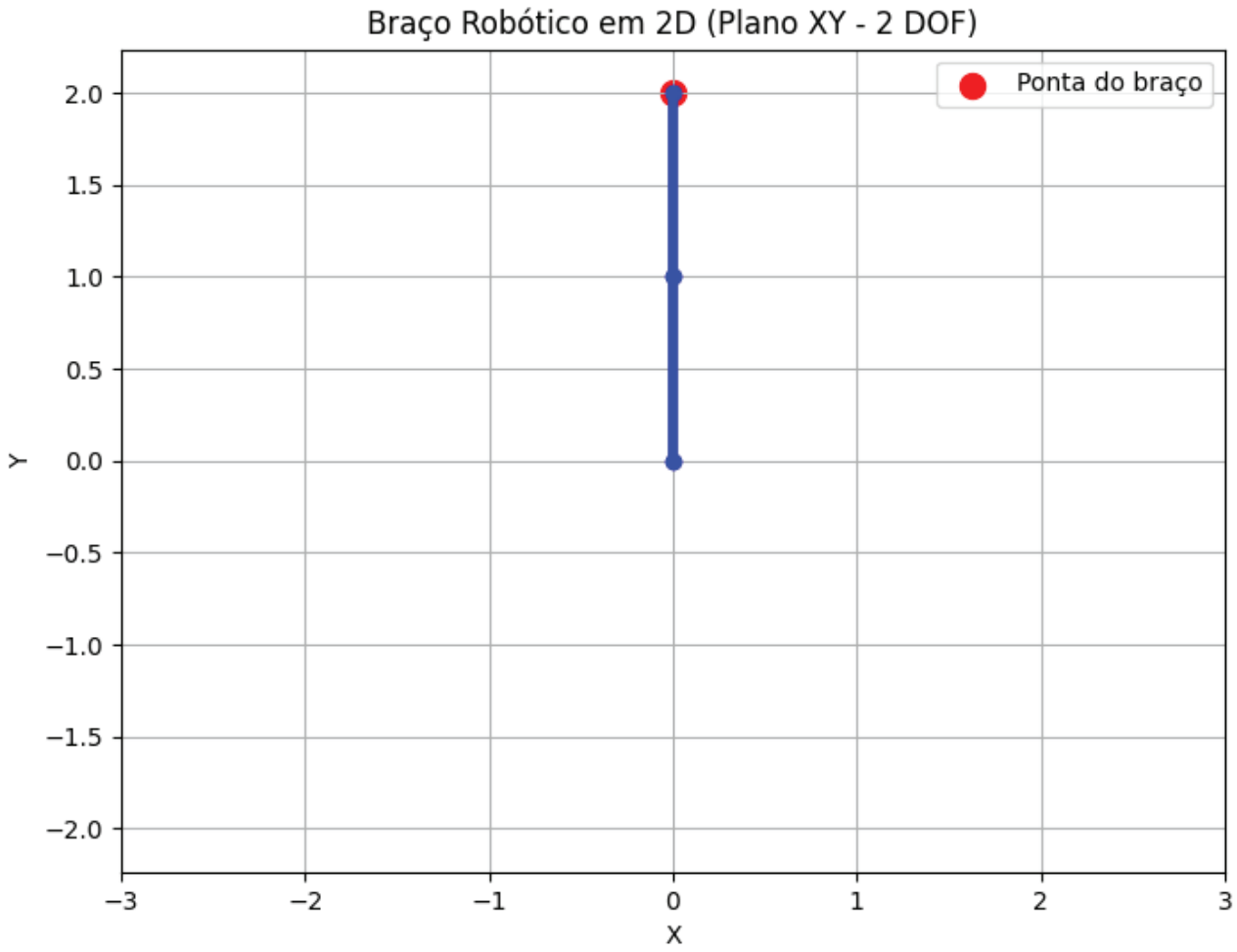


Fonte: O autor

4.2 Caso dois

Mantendo os elos em 1, o segundo caso, com rotação de 90° na base (θ = 90°, θ = 0°), resultou na posição (0.0, 2.0), validando o cálculo das transformações rotacionais no plano horizontal, como demonstra a Figura 4.

Figura 4 - Resultado para (a_1 = 1.0, a_2 = 1.0) e (θ_1 = 90°, θ_2 = 0°)

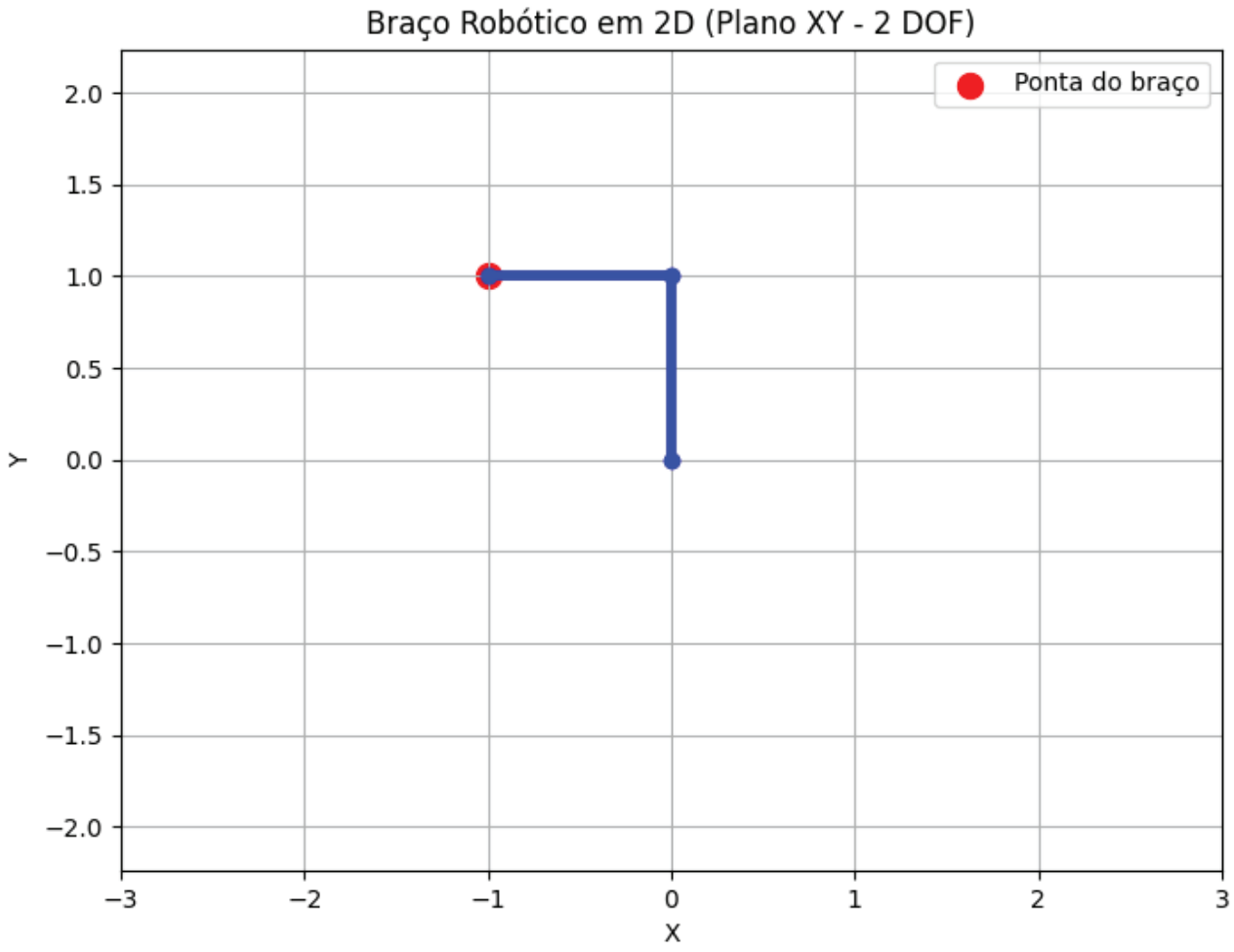


Fonte: O autor

4.3 Caso três

O terceiro caso testado com as dimensões dos elos em 1 e com θ_1 = 90° e θ_2 = 90°, produziu o resultado esperado (-1.0, 1.0), confirmando a correta implementação da componente bidimensional do movimento, como apresenta a Figura 5.

Figura 5 - Resultado para (a_1 = 1.0, a_2 = 1.0) e (θ_1 = 90°, θ_2 = 90°)



Fonte: O autor

Estes resultados confirmam a precisão do modelo implementado, com erros numéricos desprezíveis (menores que 10^{-15}) atribuíveis às limitações inerentes de representação em ponto flutuante, fenômeno bem documentado na literatura de computação científica (Higham, 2002). A consistência dos resultados obtidos valida não apenas a implementação computacional, mas também a adequação da formulação matemática baseada nos parâmetros de Denavit-Hartenberg para esse tipo de sistema robótico.

4.4 Espaço de trabalho

A análise do espaço de trabalho (*workspace*) do manipulador em duas dimensões revela características geométricas essenciais para aplicações práticas, fornecendo informações sobre as capacidades e limitações operacionais do sistema. Para um manipulador plano com comprimentos de elos representativos (a_1 = 1.0 e a_2 = 0.8), o espaço de trabalho apresenta uma forma de coroa circular, típica de manipuladores com dois graus de liberdade atuando no plano XY. O alcance máximo de 1,8 u.a. é obtido, quando ambos os elos estão totalmente estendidos em linha reta, enquanto o alcance mínimo de 0,2 corresponde à configuração recolhida, em que os elos se sobrepõem. A área de trabalho resultante, aproximadamente 10,05 u.a.^{2}, representa uma métrica importante para a avaliação da cobertura espacial do manipulador. Conforme discutido por Yoshikawa (1990) em seus estudos sobre manipulabilidade e análise geométrica de manipuladores, a forma e a extensão do espaço de trabalho têm impacto direto na eficácia e na flexibilidade de aplicação dos sistemas robóticos, mesmo em configurações planas.

Essa análise geométrica do espaço de trabalho é fundamental para o planejamento de trajetórias e posicionamento de estações de trabalho em aplicações industriais. Craig (2017) enfatiza que o conhecimento detalhado das limitações espaciais do manipulador é pré-requisito essencial para o desenvolvimento de estratégias de controle eficazes e para a otimização do *layout* de células robóticas.

4.5 Singularidades

O manipulador analisado apresenta singularidades cinemáticas em configurações específicas que merecem análise detalhada devido às suas implicações práticas. A singularidade de cotovelo ocorre quando $\theta_2 = 0°$ ou $\theta_2 = 180°$, situações em que os dois elos ficam alinhados, resultando em perda local de um grau de liberdade e impossibilidade de movimento em certas direções do espaço cartesiano. A singularidade de ombro, embora menos evidente ness e sistema simplificado, manifesta-se quando as coordenadas x_2 e y_2 são simultaneamente nulas, condição que torna o ângulo θ_1 torna-se indefinido ou fisicamente ambíguo. Nessas situações, o manipulador colapsa em uma configuração totalmente retraída, com os elos sobrepostos. Tais singularidades correspondem a pontos em que o manipulador perde graus de liberdade de movimento localmente, comprometendo sua capacidade de gerar certos vetores de velocidade no espaço de trabalho. Conforme discutido por Siciliano *et al.* (2009), essas configurações podem resultar em instabilidades no controle, além de causar movimentos abruptos ou imprecisos, comprometendo a segurança e a precisão em aplicações práticas.

A identificação e análise dessas singularidades são fundamentais para o desenvolvimento de algoritmos de controle robustos. Nakamura (1991) destaca que o conhecimento das configurações singulares permite o desenvolvimento de estratégias de evitação que garantem operação suave e contínua do manipulador, aspecto essencial para aplicações industriais que demandam alta confiabilidade operacional.

4.6 Análise computacional

A análise de *performance* computacional revelou características de eficiência que tornam o algoritmo adequado para implementação em sistemas de controle em tempo real. O tempo médio de cálculo, inferior a 0.001 segundos por iteração, demonstra a eficiência computacional da implementação baseada em operações matriciais otimizadas do NumPy. O tempo de renderização gráfica, mantido abaixo de 0.1 segundo, permite visualização interativa suave mesmo em sistemas computacionais de especificações modestas. O consumo de memória limitado a menos de 10 MB torna a implementação adequada para sistemas embarcados utilizados em controladores robóticos industriais. Ess es resultados de *performance*, segundo van der Walt, Colbert e Varoquaux (2011), refletem as otimizações inerentes às bibliotecas científicas Python modernas, que utilizam implementações em linguagens de baixo nível para operações matematicamente intensivas.

5 CONCLUSÕES

Este trabalho apresentou uma implementação de modelagem cinemática direta para um manipulador robótico planar com dois graus de liberdade, demonstrando a aplicabilidade prática dos fundamentos teóricos da robótica através de ferramentas computacionais modernas.

A utilização sistemática dos parâmetros de Denavit-Hartenberg proporcionou uma base matemática sólida e padronizada para o desenvolvimento do modelo, confirmando a relevância dess a metodologia clássica na análise de sistemas robóticos contemporâneos. A implementação computacional em Python, aproveitando as capacidades das bibliotecas NumPy e Matplotlib, demonstrou ser eficiente, precisa e acessível para desenvolvimento e modificação por parte da comunidade acadêmica e industrial.

A ferramenta de visualização bidimensional desenvolvida oferece uma interface intuitiva e pedagogicamente valiosa para análise do comportamento cinemático do manipulador, facilitando o entendimento dos conceitos fundamentais da robótica tanto para estudantes quanto para engenheiros. Entretanto, é importante reconhecer as limitações deste estudo. O modelo implementado é restrito a movimento planar, não incorpora restrições físicas das juntas nem considera aspectos dinâmicos do sistema. Essas limitações devem ser consideradas em futuras implementações práticas.

A metodologia empregada demonstra potencial de extensibilidade para manipuladores com maior número de graus de liberdade, constituindo uma base para desenvolvimentos futuros de maior complexidade. A abordagem modular do código desenvolvido permite a incorporação de funcionalidades adicionais, característica que facilita sua utilização como plataforma de pesquisa e desenvolvimento.

A validação através de casos de teste analíticos confere confiabilidade aos resultados e estabelece precedente metodológico para trabalhos similares na área. A contribuição principal deste trabalho reside na demonstração prática da aplicação dos conceitos teóricos de cinemática robótica através de uma implementação computacional acessível, bem documentada e pedagogicamente orientada.

Como trabalhos futuros, sugere-se a implementação da cinemática inversa, a fim de completar o conjunto de ferramentas fundamentais para a análise completa de manipuladores robóticos. Além disso, propõe-se a incorporação de restrições físicas das juntas, incluindo limites angulares e considerações de interferência mecânica, com o objetivo de tornar os modelos mais realistas. Por fim, recomenda-se a realização de validação experimental com *hardware* físico, por meio de testes com manipuladores reais, visando à consolidação completa da ferramenta desenvolvida.

REFERÊNCIAS

ANGELES, J. **Fundamentals of Robotic Mechanical Systems**: Theory, Methods, and Algorithms. 3. ed. New York: Springer, 2007.

CORKE, P. Robotics, **Vision and Control**: Fundamental Algorithms in MATLAB. 2. ed. Berlin: Springer, 2017.

CRAIG, J. J. **Introduction to Robotics: Mechanics and Control**. 4. ed. Boston: Pearson, 2017.

DENAVIT, J.; HARTENBERG, R. S. A kinematic notation for lower-pair mechanisms based on matrices. **Journal of Applied Mechanics**, v. 22, n. 2, p. 215-221, 1955. Disponível em: A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices | J. Appl. Mech. | ASME Digital Collection. Acesso em: 20 mai. 2025.

GOLUB, G. H.; VAN LOAN, C. F. **Matrix Computations**. 4. ed. Baltimore: Johns Hopkins University Press, 2013.

HARRIS, C. R. et al. Array programming with NumPy. **Nature,** v. 585, n. 7825, p. 357-362, 2020. Disponível em: https://www.nature.com/articles/s41586-020-2649-2. Acesso em 23 mai. 2025.

HIGHAM, N. J. **Accuracy and Stability of Numerical Algorithms**. 2. ed. Philadelphia: SIAM, 2002.

HUNTER, J. D. Matplotlib: A 2D graphics environment. **Computing in Science & Engineering**, v. 9, n. 3, p. 90-95, 2007. Disponível em: https://www.scirp.org/reference/referencespapers?referenceid=1560579. Acesso em 27 mai. 2025.

KHALIL, W.; DOMBRE, E. **Modeling, Identification and Control of Robots**. 3. ed. London: Kogan Page Science, 2002.

LYNCH, K. M.; PARK, F. C. **Modern Robotics: Mechanics, Planning, and Control**. Cambridge: Cambridge University Press, 2017.

MCKINNEY, W. **Python for Data Analysis: Data Wrangling with Pandas**, NumPy, and IPython. Sebastopol: O'Reilly Media, 2012.

MURRAY, R. M.; LI, Z.; SASTRY, S. S. A **Mathematical Introduction to Robotic Manipulation**. Boca Raton: CRC Press, 1994.

NAKAMURA, Y. **Advanced Robotics: Redundancy and Optimization**. Boston: Addison-Wesley, 1991.

SCIAVICCO, Luigi; SICILIANO, Bruno. **Modeling and Control of Robot Manipulators.** 2nd edition. London: Springer-Verlag, 2000.

SICILIANO, B. et al. **Robotics: Modelling, Planning and Control**. London: Springer, 2009.

SPONG, M. W.; HUTCHINSON, S.; VIDYASAGAR, M. **Robot Modeling and Control.** 2. ed. Hoboken: John Wiley & Sons, 2020.

VAN DER WALT, S.; COLBERT, S. C.; VAROQUAUX, G. The NumPy array: a structure for efficient numerical computation. **Computing in Science & Engineering**, v. 13, n. 2, p. 22-30, 2011. Disponível em: https://arxiv.org/abs/1102.1523. Acesso em 27 mai. 2025.

VAN ROSSUM, G.; DRAKE, F. L. **Python 3 Reference Manual.** Scotts Valley: CreateSpace, 2009.

YOSHIKAWA, T. **Foundations of Robotics: Analysis and Control.** Cambridge: MIT Press, 1990.

SAFAEI, A.; KOO, Y. C.; MAHYUDDIN, M. N. Adaptive model-free control for robotic manipulators. I**n: Proceedings of the 2017 International Conference on Advanced Intelligent Mechatronics**. 2017. Disponível em: https://www.researchgate.net/figure/Application-2-Schematic-of-a-two-DOF-robotic-arm-in-2D-space_fig1_322412560. Acesso em: 5 jun. 2025.

ALMOSTAFA, Mahmoud Rabbah; RABBAH, Nabila; BELHADAOUI, Hicham; RIFI, Mounir. Python in real time application for mobile robot. In: SMART APPLICATION AND DATA ANALYSIS FOR SMART CITIES (SADASC'18), 2018. Disponível em: https://ssrn.com/abstract=3179445. Acesso em: 26 jun. 2025.

VEGA, J.; CAÑAS, J. M. PyBoKids: an innovative Python-based educational framework using real and simulated Arduino robots. Electronics, v. 8, n. 8, 899, 2019. DOI: 10.3390/electronics8080899. Acesso em: 25 jun. 2025.

DI GREGORIO, Raffaele (ed.). Kinematics and Robot Design I, KaRD2018. Basel: MDPI AG, 2021. ISBN 978-3-0365-1018-7. DOI: 10.3390/books978-3-0365-1019-4. Acesso em: 25 jun. 2025.

DELHAISSE, Brian; ROZO, Leonel; CALDWELL, Darwin G. PyRoboLearn: A Python Framework for Robot Learning Practitioners. In: CONFERENCE ON ROBOT LEARNING, 30 out.–1 nov. 2020. Proceedings of Machine Learning Research, v. 100, p. 1348–1358. PMLR, 2020. Disponível em: https://proceedings.mlr.press/v100/delhaisse20a.html. Acesso em: 26 jun. 2025.