UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

EBER DELGADO DE SOUZA

DESENVOLVIMENTO DE UM APLICATIVO NO AMBIENTE DO MATLAB PARA O ROBÔ PUMA 560

CAMPO MOURÃO

2022

EBER DELGADO DE SOUZA

DESENVOLVIMENTO DE UM APLICATIVO NO AMBIENTE DO MATLAB PARA O ROBÔ PUMA 560

Development of an application in Matlab nvironment for the PUMA 560 robot

Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Bacharel em Engenharia Eletrônica do Curso de Bacharelado em Engenharia Eletrônica da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Dr. Flávio Luiz Rossini

Coorientador: Me. Luiz Fernando Pinto de Oliveira

CAMPO MOURÃO

2022

EBER DELGADO DE SOUZA

DESENVOLVIMENTO DE UM APLICATIVO NO AMBIENTE DO MATLAB PARA O ROBÔ PUMA 560

Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Bacharel em Engenharia Eletrônica do Curso de Bacharelado em Engenharia Eletrônica da Universidade Tecnológica Federal do Paraná.

Data de aprovação: 17/Novembro/2022

Leandro Castilho Brolin
Doutor
Universidade Tecnológica Federal do Paraná(UTFPR)

Lucas Ricken Garcia
Doutor
Universidade Tecnológica Federal do Paraná(UTFPR)

Flavio Luiz Rossini
Doutor
Universidade Tecnológica Federal do Paraná(UTFPR)

CAMPO MOURÃO

2022

RESUMO

O presente trabalho consiste no desenvolvimento de um aplicativo *desktop* aplicado ao robô PUMA 560. O aplicativo foi desenvolvido no *software* Matlab® a partir da ferramenta *App Designer*. O trabalho teve foco na adequação das equações da cinemáticas direta e inversa e nas equações para a geração de trajetória do referido robô. Assim, o objetivo do trabalho foi o desenvolvimento do aplicativo e a utilização do Matlab® como forma de tornar o ensino da robótica menos abstrato e mais acessível aos alunos e profissionais da área. Apresentou-se a modelagem da cinemática direta do PUMA 560, a implementação da cinemática inversa no Matlab® e a geração de trajetória através do polinômio cúbico. Os requisitos do aplicativo foram o cálculo da cinemática direta e inversa, a geração de trajetória e a representação visual do robô.

Palavras-chave: app designer; cinemática; matlab; puma 560; robótica.

ABSTRACT

The present work consists of the development of a desktop application applied to the PUMA 560 robot. The application was developed in the software Matlab® from the App Designer tool. The work focused on fitting the equations for direct and inverse kinematics and the equations for the trajectory generation of the referred robot. Thus, the objective of the work was to develop the application and use Matlab® as a way to make the teaching of robotics less abstract and more accessible to students and professionals in the field. The modeling of the direct kinematics of the PUMA 560, the implementation of the inverse kinematics in Matlab® and the trajectory generation through the cubic polynomial were presented. The requirements of the application were the calculation of direct and inverse kinematics, trajectory generation, and visual representation of the robot.

Keywords: app designer; knimatics; matlab; puma 560; robot.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

s_1	$sen(\theta_1)$
s_2	$sen(\theta_2)$
s_3	$sen(\theta_3)$
s_4	$sen(\theta_4)$
s_5	$sen(\theta_5)$
s_6	$sen(\theta_6)$
s_{23}	$sen(\theta_{23})$
c_1	$\cos(\theta_1)$
c_2	$\cos(\theta_2)$
c_3	$\cos(\theta_3)$
c_4	$\cos(\theta_4)$
c_5	$\cos(\theta_5)$
c_6	$\cos(\theta_6)$
c_{23}	$\cos(\theta_{23})$
IRF	*International Federation of Robotics*
SR	Sistema de Referência

SUMÁRIO

1 INTRODUÇÃO

As historias, livros e filmes de ficção científica têm grande influencia no conceito em que as pessoas tem sobre o que é um robô e o que um robô pode fazer. A primeira vez que se tem o registro do uso do termo robô é de uma peça de ficção científica tcheca de 1920. Entretanto, essa tecnologia ainda não está no nível que é apresentada na ficção, mas está em constante evolução e a cada ano que passa está um passo mais perto (CORKE, 2017; CORTES, 2011).

Um robô pode ser definido como uma máquina orientada para um objetivo que pode detectar, planejar e agir. Ele capta informações do ambiente, planeja e realiza uma ação de acordo com um determinado objetivo. Essa ação pode ser o movimento de uma ferramenta de um robô manipulador ou o movimento de um carro autônomo. Neste contexto, robôs são dispositivos complexos e versáteis que envolvem diferentes áreas de estudo, tais como mecânica, elétrica, eletrônica, matemática, ciência da computação, entre outras (CORKE, 2017; JAZAR, 2007).

Existem diferentes tipos de robôs, e cada um possui uma aplicação e funcionalidade específica. Eles podem ser classificados em três grandes grupos: robôs móveis, humanoides ou industriais. Entre os industriais estão os manipuladores robóticos, que são objeto de estudo deste trabalho. Esses robôs são de grande importância para a automação industrial, pois geram aumento na produtividade, eficiência e rentabilidade das empresas (JAZAR, 2007; CORTES, 2012)..

Os manipuladores robóticos estão presentes nas indústrias há algumas décadas. O primeiro desse tipo comercializado foi o Unimates na década de 1960, criado pelo engenheiro americano Joseph F. Engelberger (CORTES, 2012).

Os robôs são utilizados em diferentes áreas e processos tecnológicos. Nas indústrias, eles realizam trabalhos perigosos, repetitivos, monótonos ou impossíveis de serem executados por humanos, pois, eles podem operar em ambientes extremos com precisão e repetibilidade. Além disso, em muitas tarefas os robôs têm desempenho superior a um humano, pois eles conseguem trabalhar ininterruptamente, aumentam a produtividade e podem ser utilizados em locais perigosos, que colocam em risco a vida de trabalhadores (CORKE, 2017).

A robótica pode ser considerada umas das áreas tecnológicas mais importantes do século. Máquinas inteligentes podem ser encontradas em diferentes lugares, carros autônomos já estão presentes no mercado, fábricas totalmente automatizadas são realidade em países desenvolvidos, robôs domésticos podem ser facilmente comprados pela internet (BULLER; GIFFORD; MILLS, 2018; CORKE, 2017).

Nos ambientes rurais os robôs estão sendo usados para automatizar as atividades agrícolas, eles contribuem para a melhora na qualidade dos alimentos e para o aumento da produção, auxiliam, deste modo, a suprir à crescente demanda por alimentos. E a cada ano novas tecnologias são criadas, e fomentam ainda mais essa ciência e auxiliam na evolução da humanidade (OLIVEIRA; SILVA; MOREIRA, 2020; OLIVEIRA *et al.*, 2020).

Neste presente trabalho realizou-se o desenvolvimento de um aplicativo para o robô manipulador PUMA 560. Para isso foram checadas e adequadas as equações do robô e, em seguida, desenvolveu-se um aplicativo com as ferramentas presentes no *software* Matlab®.

1.1 Objetivos

Nesta seção, são apresentados os objetivos, que podem ser divididos em objetivo geral e objetivos específicos, conforme serão descritos nos itens abaixo.

1.1.1 Objetivo geral

Realizar um estudo teórico sobre a cinemática de um manipulador robótico para a implementação e desenvolvimento de um aplicativo no ambiente do *software* Matlab®.

1.1.2 Objetivos específicos

- Modelar e checar matematicamente a cinemática direta do manipulador;
- Implementar a cinemática inversa;
- Realizar as simulações no *software* Matlab®;
- Desenvolver o aplicativo;
- Implementar as funcionalidades do aplicativo.

1.2 Justificativa

A automação de processos por meio de sistemas robóticos tem ganhado cada vez mais espaço no mercado. A diminuição do custo para a produção de um robô, o aumento do custo da mão de obra humana e a evolução dessa tecnologia são os principais fatores que contribuíram para o aumento da demanda por esses sistemas robóticos. Atualmente, essa tecnologia está presente na maioria das indústrias, e outros setores como comércio e serviço também a utilizam para automatizar seus processos. Portanto, a robótica está em constante evolução e constantemente surgem novas aplicações e ferramentas emergem para contribuir com o seu desenvolvimento e aprimoramento (CRAIG, 2012).

De acordo com o *International Federation of Robotics* (IFR), em 2020 foram instalados 1595 novos robôs no Brasil, um declínio de 13% em relação à 2015, o qual implica em um crescimento médio anual de 3%. Em escala global o crescimento para 2020 foi de 0,5%. A

indústria que mais cresceu nesse ano foi a indústria eletrônica com 29% das instalações (IFR, 2021a).

Na educação, a robótica pode ter um papel importante no processo de aquisição do conhecimento, pois com a robótica, o aluno desenvolve um aprendizado ativo e participativo. Ademais, a robótica reúne diversos campos tecnológicos, tais como engenharia e computação, de forma lúdica e interessante. Apesar da riqueza de possibilidades que a robótica oferece, ela ainda não está difundida nas escolas do Brasil (ZILLI, 2004).

Na história da robótica existem alguns robôs ícones, que foram de suma importância para o desenvolvimento da área. Dentre essses podem ser citados o Unimation, primeiro robô comercializado e utilizado em uma linha de montagem e o IRB 6, lançado em 1974, foi o primeiro robô industrial totalmente elétrico controlado por microprocessador (IFR, 2021b).

Em 1978 a Unimation junto com a General Motors lançam o PUMA, sigla para *Programmable Universal Machine for Assembly*, esse robô é considerado um arquetípico para robôs antropomórficos e foi desenvolvido para pequenos manuseios e ocupavam o mesmo espaço do que um operador humano. A modelagem do PUMA é utilizada como exemplo por diversos livros de robótica e ele se tornou um robô importante para o ensino (IFR, 2021b; GASPARETTO; SCALERA, 2019). Por esse motivo o modelo PUMA 560 foi o robô escolhido para o presente trabalho.

O *software* Matlab®, acrônimo para laboratório de matriz, teve sua primeira versão desenvolvida pelo professor e matemático Cleve Moeler em 1970. Na década seguinte, em 1984, Moeler fundou a empresa Mathworks que desenvolve e distribui o Matlab® até os dias atuais (CORTES, 2012). Neste trabalho utilizou-se o Matlab® R2021a para o desenvolvimento do aplicativo, para a modelagem da cinemática direta, para o cálculo da cinemática inversa e para a geração de trajetória.

O Matlab® é uma plataforma de programação e computação numérica usada para desenvolver algoritmos, análise de dados e criação de modelos. Ele possui diversas ferramentas e implementos, que possibilita a sua utilização em todas as áreas da engenharia. Umas dessas ferramentas permite a criação de aplicativos *desktop* com a programação desenvolvida no Matlab®, essa ferramenta é o *App Designer* (MATHWORKS, 2022a).

Aplicado à robótica, o Matlab® já possui diversas *Toolboxs* com o foco no desenvolvimento de robôs, fornecendo a modelagem, o desenvolvimento de controles, entre outros. Contudo, o Matlab® também possibilita que seus usuários desenvolvam suas próprias ferramentas e aplicativos. Neste trabalho realizou-se uma análise da utilização do Matlab® aplicado à robótica, com foco no uso educacional, e não foi feito uso dessas *Toolboxs*.

Com o desenvolvimento do aplicativo, busca-se providenciar uma forma menos abstrata de desenvolvimento da modelagem de um robô. Pois essa modelagem possui certa profundidade matemática, sendo necessário um grau de abstração para compreender o seu resultado, o que dificulta no aprendizado da robótica.

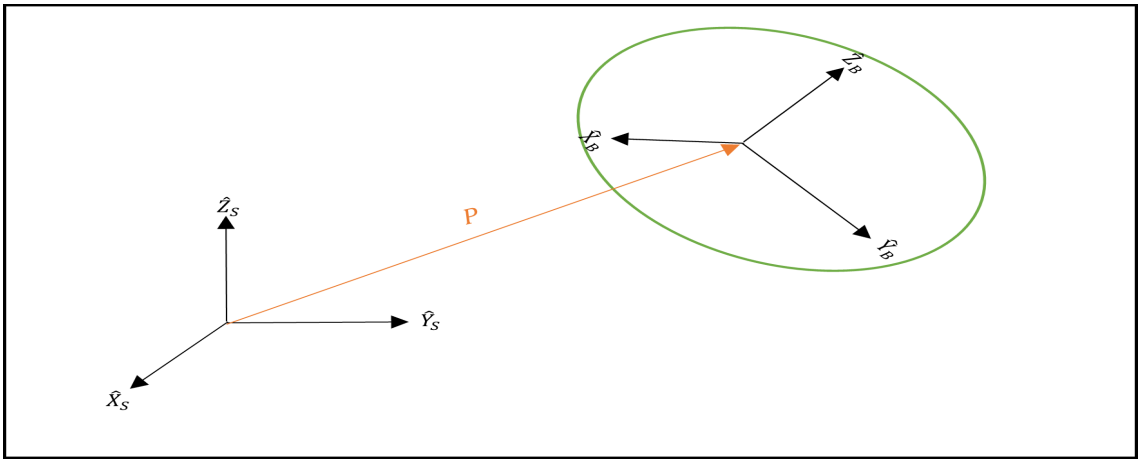
2 FUNDAMENTAÇÃO TEÓRICA

No estudo da manipulação robótica é essencial representar posições e orientações de peças, ferramentas e mecanismos em quantidades numéricas. Para isso é necessário estabelecer sistemas de coordenadas e desenvolver convenções, que formam a base do estudo e do desenvolvimento das manipulações robóticas (CRAIG, 2012).

2.1 Descrições espaciais e transformações

Quando adota-se um sistema universal de coordenadas é possível referenciar qualquer ponto no espaço a ele, ou seja, todas as posições e orientações de peças, ferramentas e corpo de um robô podem ser descritas em relação a esse sistema universal (CRAIG, 2012). Na Figura 1 se tem um exemplo dessa metodologia, em que é definido um sistema universal de referência $\{S\}$ e outro Sistema de Referência (SR) $\{B\}$ fixo em um corpo móvel. Com isso, é possível referenciar a posição e orientação desse corpo em relação ao sistema universal $\{S\}$ através do vetor de posição P, Equação (1), e da matriz rotacional ${}^S_B R$, Equação (2), respectivamente (LYNCH; PARK, 2017).

Figura 1 – Sistema de referência $\{\mathrm{B}\}$ em relação à um sistema universal $\{\mathrm{S}\}$



Fonte: Adaptado de Lynch e Park (2017, p. 68).

$$P = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} \tag{1}$$

$${}^S_B R = [{}^S\widehat{X}_B \ {}^S\widehat{Y}_B \ {}^S\widehat{Z}_B] = \begin{bmatrix} \widehat{X}_B \cdot \widehat{X}_S & \widehat{Y}_B \cdot \widehat{X}_S & \widehat{Z}_B \cdot \widehat{X}_S \\ \widehat{X}_B \cdot \widehat{Y}_S & \widehat{Y}_B \cdot \widehat{Y}_S & \widehat{Z}_B \cdot \widehat{Y}_S \\ \widehat{X}_B \cdot \widehat{Z}_S & \widehat{Y}_B \cdot \widehat{Z}_S & \widehat{Z}_B \cdot \widehat{Z}_S \end{bmatrix} \tag{2}$$

A representação da posição e da orientação pode ser feita em uma única matriz 4×4. Chamada de matriz de transformação homogênea, essa é a combinação de um vetor de posição com uma matriz rotacional e descreve a relação entre dois sistemas de referência. A matriz ${}^{S}_{B}T$, Equação (3), é uma matriz de transformação, em que foram combinados o vetor de posição P, Equação (1), com a matriz ${}^{S}_{B}R$, Equação (2), e representa a posição e orientação do SR$\{B\}$ em relação ao SR$\{S\}$, Figura 1 (LYNCH; PARK, 2017).

$$ {}^{S}_{B}T = \begin{bmatrix} {}^{S}_{B}R & P \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \widehat{X}_B \cdot \widehat{X}_S & \widehat{Y}_B \cdot \widehat{X}_S & \widehat{Z}_B \cdot \widehat{X}_S & P_x \\ \widehat{X}_B \cdot \widehat{Y}_S & \widehat{Y}_B \cdot \widehat{Y}_S & \widehat{Z}_B \cdot \widehat{Y}_S & P_y \\ \widehat{X}_B \cdot \widehat{Z}_S & \widehat{Y}_B \cdot \widehat{Z}_S & \widehat{Z}_B \cdot \widehat{Z}_S & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3} $$

Em um mapeamento a descrição de um vetor P pode ser conhecida em relação a um SR$\{B\}$ e desconhecida em relação a outro SR$\{A\}$. Para encontrar a relação desse vetor com o SR$\{A\}$ basta conhecer a matriz de transformação ${}^{B}_{A}T$, pois ao multiplicá-la pelo vetor ${}^{B}P$ encontra-se o vetor de interesse ${}^{A}P$, Equação (4) (CRAIG, 2012). Neste contexto, também é possível obter a matriz de transformação ${}^{B}_{A}T$, que descreve a relação do SR$\{C\}$ com o SR$\{A\}$, Equação (5), para isso basta conhecer a relação entre os SRs, ou seja, SR$\{C\}$ em relação ao SR$\{B\}$ e este em relação ao SR$\{A\}$ (JAZAR, 2007).

$$ {}^{A}P = {}^{A}_{B}T \cdot {}^{B}P \tag{4} $$

$$ {}^{A}_{C}T = {}^{A}_{B}T \cdot {}^{B}_{C}T \tag{5} $$

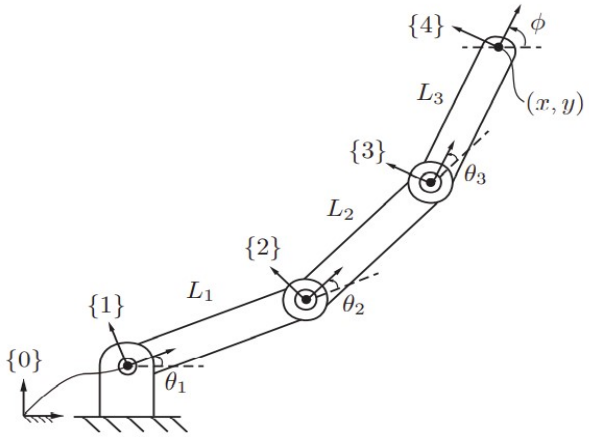
2.2 Cinemática direta

Segundo Craig (2012, p. 60) “Um manipulador robótico pode ser considerado como um conjunto de corpos conectados em cadeia por juntas. Esses corpos são chamados de elos e as juntas são as responsáveis pela conexão entre um par de elos vizinhos.” Essa definição é de suma importância para o estudo da cinemática direta de um manipulador, pois através dela é possível realizar o cálculo da posição e orientação do efetuador de um manipulador robótico, em função das coordenadas θ das suas juntas (LYNCH; PARK, 2017).

Um robô manipulador pode ter diferentes tipos de juntas, entretanto, as juntas mais comuns são as juntas rotativas e as juntas prismáticas. Em geral, os robôs manipuladores são construídos com juntas de apenas um grau de liberdade, mas existem juntas com mais graus. Em regra, um robô com n graus de liberdade possui n juntas e, para que se possa posicionar e orientar um efetuador de um manipulador em todas as posições e orientações no espaço tridimensional, são necessárias, no mínimo, seis juntas (CRAIG, 2012).

As transformadas homogêneas são ferramentas muito úteis para o cálculo da cinemática direta de um manipulador, pois com elas é possível encontrar a orientação e posição de um efetuador através das coordenadas das suas juntas. Para isso, é necessário fixar sistemas de referências no efetuador, nas juntas e na base desse manipulador. Na Figura 2 é mostrado um manipulador com três graus de liberdade, no qual foram fixados 5 SR: um no efetuador (SR $\{$ 4 $\}$), um em cada junta (SR$\{1\}$, SR$\{2\}$ e SR$\{3\}$) e um fixo na base (SR$\{0\}$). Dessa forma é possível escrever a cinemática direta desse robô como a matriz de transformação homogênea composta ${}^0_4T$, Equação (6) (LYNCH; PARK, 2017).

Figura 2 – Cinemática direta



Fonte: Lynch e Park (2017, p. 138).

$$ {}^0_4T = {}^0_1T \cdot {}^1_2T \cdot {}^2_3T \cdot {}^3_4T \tag{6} $$

2.3 Cinemática inversa

Segundo Jazar (2007, p. 263, tradução nossa) "A determinação das variáveis das juntas em termos da posição e orientação do efetuador é chamada de cinemática inversa.". Neste contexto a cinemática inversa é essencial na modelagem de um robô manipulador, pois objetos a serem manipulados são descridos em coordenadas do sistema fixo universal e o computador que controla esse robô precisa conhecer as variáveis das juntas para manipular esses objetos (JAZAR, 2007).

Há diferentes métodos para resolver a cinemática inversa, alguns utilizam a cinemática direta para encontrar as equações que descrevem a inversa. Um desses métodos utiliza a matriz de transformação X que representa a orientação e a posição de um ponto no espaço tridimensional, e a matriz da cinemática direta $T(\theta)$, onde θ representa os ângulos ou posições das juntas. Encontrando a solução de θ que satisfaz $T(\theta) = X$ é possível resolver o problema da cinemática inversa (LYNCH; PARK, 2017).

2.4 Geração de trajetória

Segundo Craig (2012, p. 192) “trajetória se refere a um histórico de posição, velocidade e aceleração em função do tempo, para cada grau de liberdade”. A trajetória é necessária para mover o efetuador de um ponto A para um ponto B, ou para movimentar um determinado objeto (CORKE, 2017). O problema da geração de trajetória envolve facilitar a descrição do movimento a ser feita pelo usuário, que, ao invés de escrever funções complexas de tempo e espaço, deve apenas especificar o movimento com posições e orientações desejadas. Deste modo, o sistema robótico calculará a trajetória, com detalhes como percurso, duração e velocidade (CRAIG, 2012).

A trajetória pode descrever diferentes movimentos de um manipulador, que vão desde movimentos simples com apenas um ponto inicial e final, à movimentos mais complexos que descrevem diferentes pontos de passagens com diferentes velocidades e acelerações (CRAIG, 2012).

Um movimento simples de um manipulador é uma trajetória que descreve uma linha reta, chamado de ponto a ponto, no qual o manipulador desloca-se de uma configuração inicial θ_i para uma configuração final θ_f, onde os θ representam as juntas desse manipulador. Utiliza-se o polinômio cúbico da Equação (7), para calcular todos os valores que as juntas de um manipulador devem assumir para realizar uma determinada trajetória ponto a ponto. Ao derivar esse polinômio encontra-se a velocidade das juntas em função do tempo, Equação (8), e com a segunda derivada a aceleração nessa trajetória, Equação (9) (LYNCH; PARK, 2017).

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \tag{7}$$

$$\dot{\theta}(t) = a_1 + 2a_2 t + 3a_3 t^2 \tag{8}$$

$$\ddot{\theta}(t) = 2a_2 + 6a_3 t \tag{9}$$

Ao resolver essas equações, obtêm-se as quatro constantes a_0, a_1, a_2 e a_3, Equações (10) a (13) (LYNCH; PARK, 2017).

$$a_0 = \theta_i \tag{10}$$

$$a_1 = 0 \tag{11}$$

$$a_2 = \frac{3}{t_f^2}(\theta_f - \theta_i) \tag{12}$$

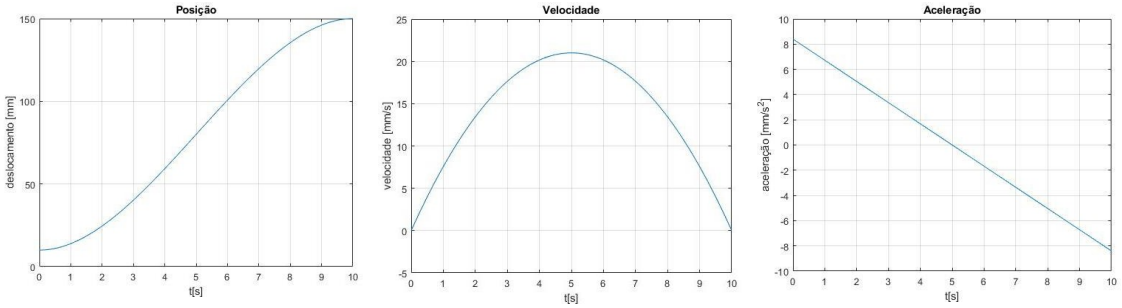
$$a_3 = \frac{3}{t_f^3}(\theta_f - \theta_i) \tag{13}$$

Com as Equações (7), (8) e (9) e com as informações de tempo, posição inicial θ_i e final θ_f das juntas é possível obter todas as posições, velocidades e acelerações de cada junta e

gerar uma trajetória ponto a ponto para um manipulador robótico (CRAIG, 2012). Nas Figuras 3a, 3b e 3c é possível analisar as funções de posição, velocidade e aceleração, respectivamente, de uma junta prismática para uma trajetória com a posição inicial da junta de $\theta_i = 0\ mm$ e final de $\theta_f = 150\ mm$.

Figura 3 – Gráficos da posição, da velocidade e da aceleração

(a) Posição **(b) Velocidade** **(c) Aceleração**



Fonte: Autoria própria (2022).

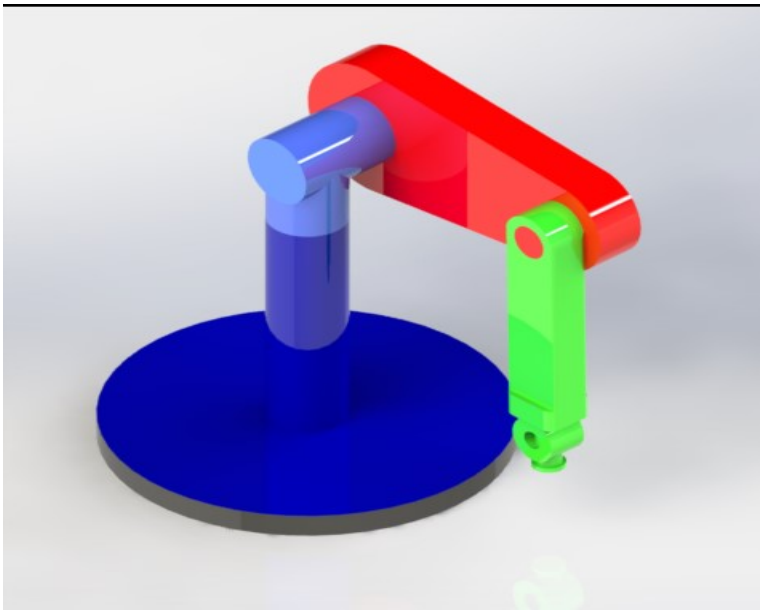
3 METODOLOGIA

O desenvolvimento desse trabalho teve foco no robô PUMA 560, e teve como base o modelo apresentado na terceira edição do livro Robótica de John J. Craig de 2012(CRAIG, 2012). Utilizou-se o *software* Matlab® para a implementação das modelagens das cinemáticas, direta e inversa, como também para a simulação das equações de geração de trajetória. No Matlab® também foi desenvolvido um aplicativo para o PUMA 560, esse aplicativo permite que o usuário insira todos os parâmetros necessários para a simulação do robô.

3.1 PUMA 560

O Unimation PUMA 560 é um braço robótico disseminado na indústria automobilística em processos de solda e pintura, além disso, ele possui simplicidade no manuseio, o que permite o seu uso em atividades didáticas. Por esses motivos foi escolhido para ser modelado neste trabalho. Ele possui seis graus de liberdade com todas as juntas rotacionais, Figura 4, que tornam o robô versátil (CRAIG, 2012).

Figura 4 – PUMA 560



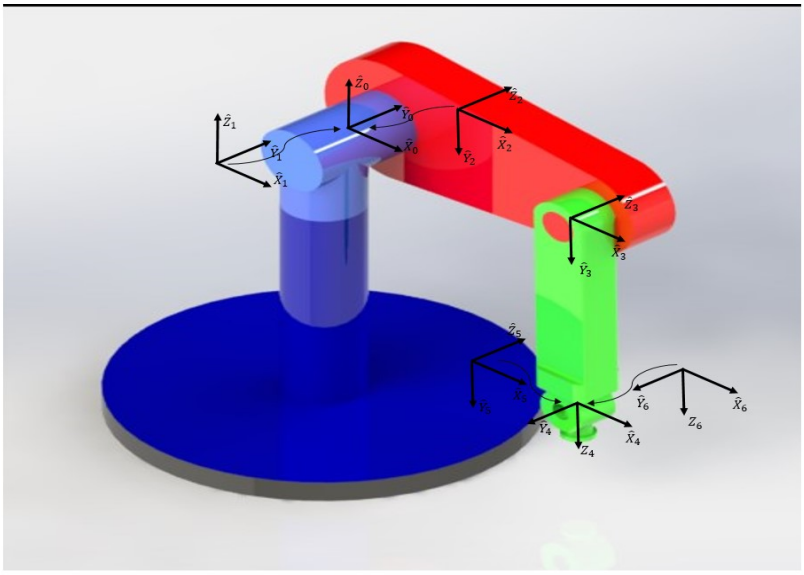
Fonte: Autoria própria (2022).

Na Figura 4 o robô está na posição inicial, com os ângulos de todas as juntas iguais a zero. Foi escolhida essa posição para que se possa ter uma comparação de resultados com os apresentados por CRAIG, possibilitou-se, deste modo, uma revisão bibliográfica.

3.2 Modelagem das cinemáticas

O Primeiro passo para a modelagem da cinemática direta é a definição dos sistemas de referência. O PUMA possui seis graus de liberdades, portanto se fez necessário a fixação de sete sistemas. Esses SR's são mostrados na Figura 5, sendo o SR {0} coincidente com o SR {1} quando o ângulo $\theta_0 = 0$, o SR {2} possui a origem em comum com os SR's {0} e {1}, esse ponto em comum também acontece com os SR's {4}, {5} e {6}. Os SR's {1} a {6} estão fixos em cada um dos eixos das juntas rotativas do robô (CRAIG, 2012).

Figura 5 – Sistemas de referência PUMA 560



Fonte: Autoria própria (2022).

Com os SR's definidos é possível construir a transformação que define o SR {i} em relação ao SR {i-1}. Com seis graus de liberdade precisou-se de seis transformações para o PUMA, Equação (14). Multiplicou-se essas seis transformações e encontrou-se a transformação do SR {6} ao SR {0}, Equação (15), sendo ela uma função de todas as seis variáveis de juntas θ do manipulador (LYNCH; PARK, 2017). Neste trabalho essa modelagem foi implementada no Matlab® e o resultado apresentado no capítulo posterior.

$$ {}^{0}_{1}T;\ {}^{1}_{2}T;\ {}^{2}_{3}T;\ {}^{3}_{4}T;\ {}^{4}_{5}T;\ {}^{5}_{6}T; \tag{14} $$

$$ {}^{0}_{6}T = {}^{0}_{1}T \cdot {}^{1}_{2}T \cdot {}^{2}_{3}T \cdot {}^{3}_{4}T \cdot {}^{4}_{5}T \cdot {}^{5}_{6}T \tag{15} $$

A Equação (15) é uma matriz 4×4 que é composta pela matriz 3×3 rotacional R e pela matriz 3×1 de posição P, Equação (16), que constitui a cinemática direta do PUMA 560. Portanto, com essa equação é possível encontrar a posição e orientação da ferramenta do manipulador através dos ângulos θ das juntas (CORKE, 2017).

$$ {}^{0}_{6}T = \begin{bmatrix} R & P \\ 0 & 1 \end{bmatrix} \tag{16} $$

A cinemática inversa do PUMA possui um grau de complexidade elevado e diferentes métodos matemáticos para encontrá-la. Neste trabalho optou-se por utilizar a modelagem apresentada por Craig, nas páginas 109 a 113 da terceira edição do livro Robótica. Dentre as soluções presentes nas bibliografias, as Equações (17) a (24) foram as escolhidas para serem implementadas no Matlab® (CRAIG, 2012).

Para os ângulos das três primeiras juntas essa modelagem utiliza-se das seguintes equações:

$$ \theta_1 = tan^{-1}\left(2 \cdot \frac{p_y}{p_x}\right) \tag{17} $$

$$ \theta_2 = \theta_{23} - \theta_3 \tag{18} $$

$$ \theta_3 = -tan^{-1}\left(\frac{K}{\sqrt{l_2^2 - K^2}}\right) \tag{19} $$

sendo:

$$ K = \frac{p_x^2 + p_y^2 + p_z^2 - l_2^2}{2 \cdot l_1} \tag{20} $$

e

$$ \theta_{23} = sen^{-1}\left(\frac{l_2 \cdot c_3 \cdot p_z + (c_1 \cdot p_x + s_1 \cdot p_y) \cdot (l_1 \cdot s_3 - l_2)}{p_z^2 - (c_1 \cdot p_x + s_1 \cdot p_y)^2}\right) \tag{21} $$

em que $l1$ e $l2$ são os comprimentos dos elos do robô. Para as últimas três juntas a modelagem é dada pelas Equações (22), (23) e (24):

$$ \theta_4 = tan^{-1}\left(2 \cdot \frac{-r_{13} \cdot s_1 + r_{23} \cdot c_1}{-r_{13} \cdot c_1 \cdot c_{23} - r_{23} \cdot s_1 \cdot c_{23} + r_{33} \cdot s_4}\right) \tag{22} $$

$$ \theta_5 = tan^{-1}\left(2 \cdot \frac{S_5}{C_5}\right) \tag{23} $$

$$ \theta_6 = tan^{-1}\left(2 \cdot \frac{S_6}{C_6}\right) \tag{24} $$

sendo:

$$ \begin{aligned} S_5 = -r_{13} \cdot (c_1 \cdot c_{23} \cdot c_4 + s_1 \cdot s_4) + r_{23} \cdot (s_1 \cdot c_{23} \cdot c_4 - c_1 \cdot s_4) \cdots \\ \cdots - r_{33} \cdot s_{23} \cdot c_4 \end{aligned} \tag{25} $$

$$C_5 = r_{13} \cdot ((-c_1) \cdot s_{23}) + r_{23} \cdot (-s_1 \cdot s_{23}) - r_{33} \cdot (c_{23}) \tag{26}$$

$$\begin{aligned} S_6 = -r_{11} \cdot (c_1 \cdot c_{23} \cdot s_4 - s_1 \cdot c_4) - r_{21} \cdot (s_1 \cdot c_{23} \cdot s_4 + c_1 \cdot c_4) \cdots \\ \cdots + r_{31} \cdot (s_{23} \cdot s_4) \end{aligned} \tag{27}$$

$$\begin{aligned} C_6 = r_{11} \cdot ((c_1 \cdot c_{23} \cdot c_4 + s_1 \cdot s_4) \cdot c_5 - c_1 \cdot s_{23} \cdot s_5) \cdots \\ \cdots + r_{21} \cdot ((s_1 \cdot c_{23} \cdot c_4 - c_1 \cdot s_4) \cdot c_5 - s_1 \cdot s_{23} \cdot s_5) \cdots \\ \cdots - r_{31} \cdot (s_{23} \cdot c_4 \cdot c_5 + c_{23} \cdot s_5) \end{aligned} \tag{28}$$

As Equações (17), (18), (19), (22), (23), e (24) resultam nos valores dos ângulos das seis juntas e formam a cinemática inversa do PUMA 560.

3.3 Geração de trajetória

Para a geração da trajetória foi desenvolvido um movimento simples ponto a ponto. Com as Equações (7), (8) e (9) calculou-se a posição, velocidade e aceleração, respectivamente, em função do tempo para cada uma das seis juntas do robô (LYNCH; PARK, 2017). No Matlab® foi implementado essas equações e desenvolvida uma simulação desse movimento.

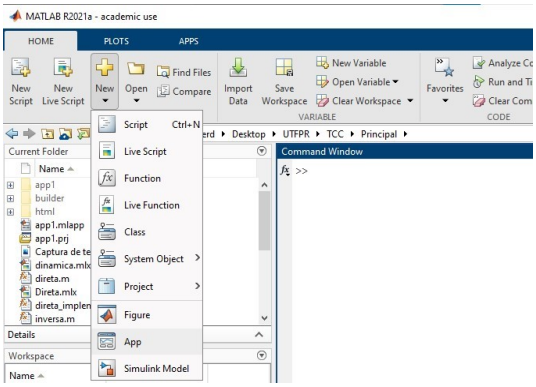
3.4 Aplicativo

O Matlab® é uma ferramenta robusta e oferece diversas funcionalidades com suas *Toolboxs*, sendo uma delas o *App Designer*, uma *Toolbox* que permite a criação de aplicativos *Desktop* implementando bibliotecas e funcionalidades do Matlab® (MATHWORKS, 2022b).

O aplicativo foi desenvolvido com a linguagem de programação Matlab para o sistema operacional Windows. O *software* Matlab® permite a compilação do aplicativo como um arquivo executável, o que permite a sua execução diretamente pelo sistema, não sendo necessário que o usuário tenha a licença do Matlab®. O desenvolvimento foi realizado em três passos, sendo o primeiro a criação da tela do aplicativo no *App Designer*, descrito abaixo nesta seção.

Para iniciar o *App Designer* o usuário deve escolher a opção *App* na aba *New* do menu superior da tela principal do Matlab®, Figura 6.

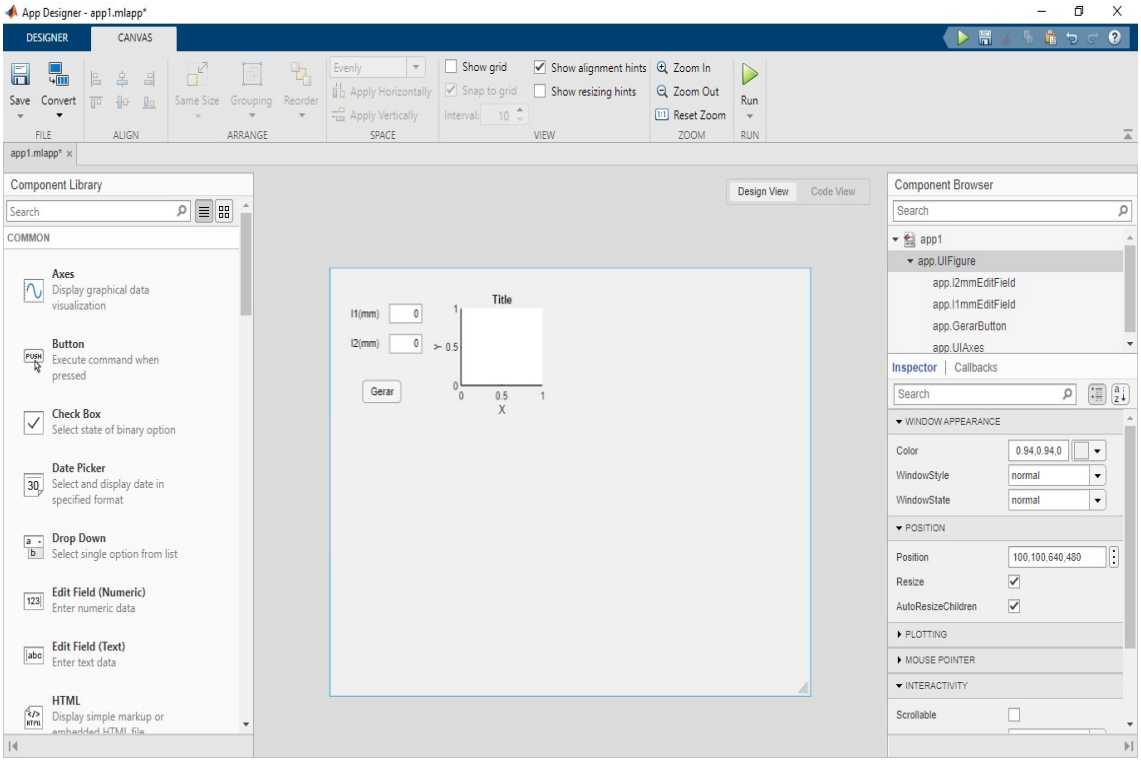
Figura 6 – Iniciar o App Designer



Fonte: Mathworks (2022a).

A Figura 7 contém uma captura de tela da interface do *App Designer*, na esquerda da imagem se tem a biblioteca de componentes visuais que a *Toolbox* oferece, na parte central se tem a pré-visualização da interface do usuário e é onde se constrói as telas do aplicativo, já na direita se tem o menu de propriedades dos componentes. O *App Designer* possui todas as funcionalidades e componentes necessários para o desenvolvimento do aplicativo para o PUMA 560, não sendo necessária a utilização de bibliotecas complementares.

Figura 7 – App Designer



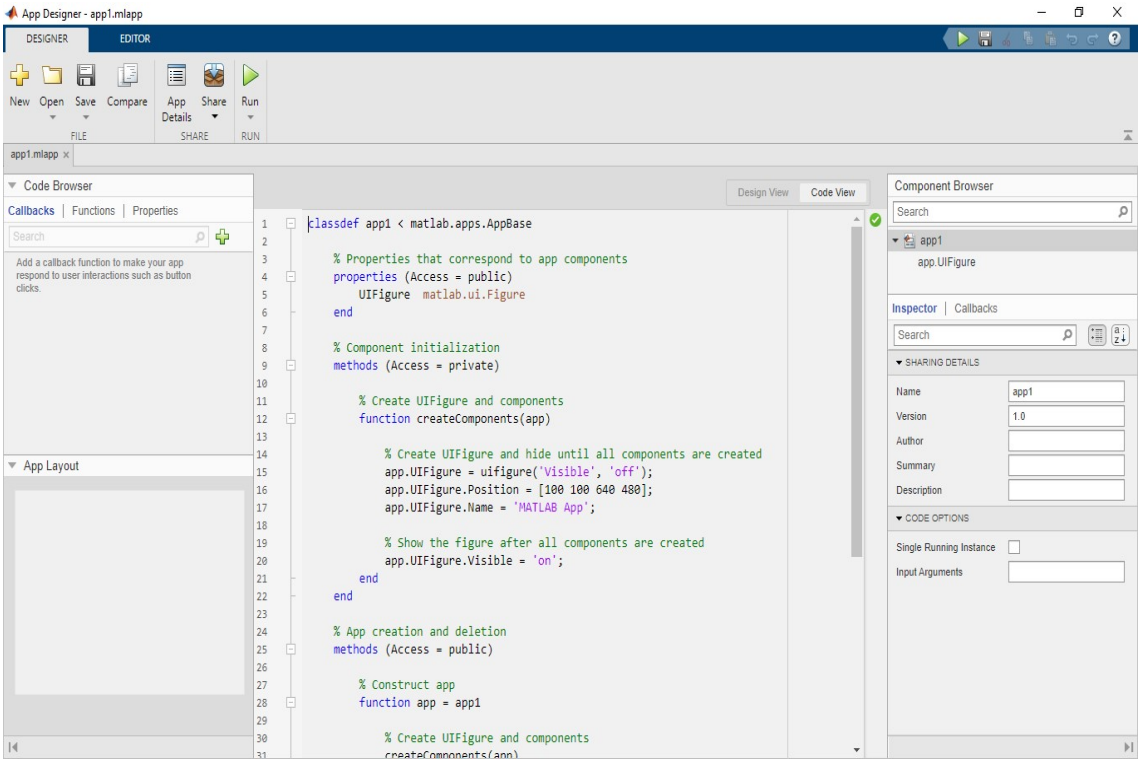
Fonte: Mathworks (2022b).

Para criar ícones no *App Designer*, o usuário deve "arrastar e soltar" os componentes do menu da esquerda para a pré-visualização do *designer* na parte central do *software*. Na Figura 7, também é apresentado um exemplo da criação do aplicativo, no qual criou-se três

componentes disponíveis no *App Designer*, *"Axes", "Button"* e *"Edit Field (Numeric)"*. O *Axes* permite o *plot* de gráficos no aplicativo, o *Button* é programado para agir com o *click* do usuário, e o *Edit Field* permite a inserção de valores numéricos pelo usuário, esses foram os principais componentes utilizados no desenvolvimento do aplicativo.

A programação do aplicativo no *App Designer* é feita através da aba *Code View*, Figura 8, e utiliza-se de *callbacks* para executar o código quando um evento acontece. Esse evento pode ser o *click* do *mouse* em um *Button* ou a inserção de um valor pelo usuário em um *Edit Field*.

Figura 8 – App Designer aba Code View



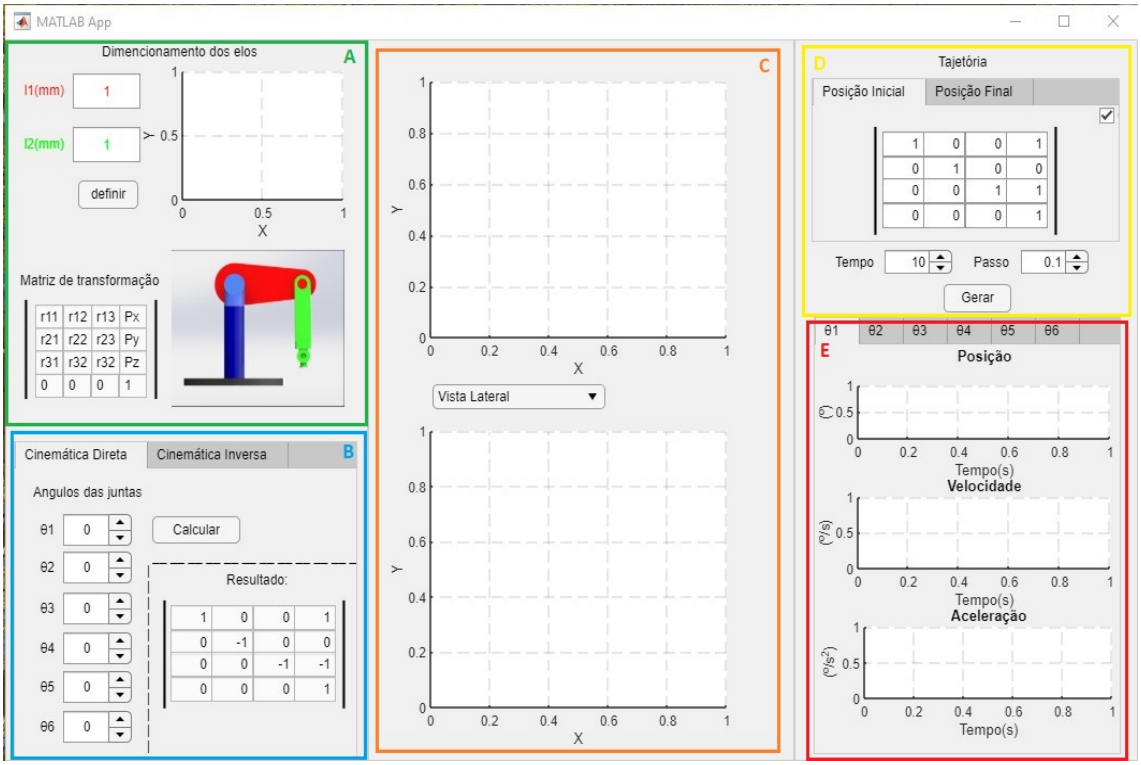
Fonte: Mathworks (2022b).

O aplicativo desenvolvido neste trabalho também manteve o foco em um modelo específico de robô, o PUMA 560. Como requisitos de desenvolvimento do aplicativo foram apresentados os seguintes pontos:

- Cálculo da cinemática direta;
- Cálculo da cinemática inversa;
- Geração da trajetória;
- Dimensionamento dos elos;
- Representação da posição do robô.

Para atender esses requisitos elaborou-se a tela apresentada na Figura 9. Na seção A, circulada em verde, se tem o dimensionamento dos elos do robô, onde se pode definir valores de 1 a 2000 mm para os elos $l1$ e $l2$, representados pelas cores vermelha e verde, respectivamente, na representação presente na seção. Após definido os elos, o gráfico presente na seção apresentará uma visualização do dimensionamento.

Figura 9 – Tela do aplicativo



Fonte: Autoria própria (2022).

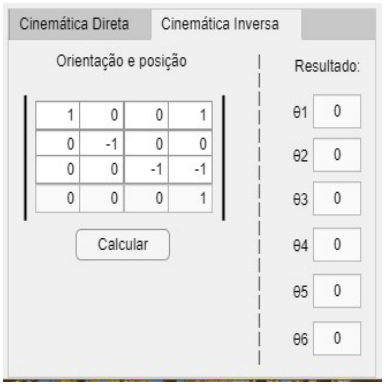
A seção B, marcada em azul, é a responsável por atender os requisitos do cálculo da cinemática direta e inversa, sendo dividida em duas abas. A primeira calcula a matriz de transformação para uma entrada com os valores dos ângulos das juntas e apresenta na seção C a representação desta posição.

Para a cinemática inversa se tem a segunda aba, Figura 10, nela a entrada é a matriz de transformação e a saída, ou seja, o resultado, os valores dos ângulos do PUMA em graus(°), para a posição e orientação dada pela matriz. Além disso, é gerado uma visualização desta posição na seção C.

Na seção C, marcada em amarelo, existe dois componentes gráficos fundamentais para a representação dos resultados, tanto do cálculo das cinemáticas quanto da geração de trajetória. Nesses gráficos são apresentadas as vistas da representação do robô para as posições obtidas nos resultados. Sendo que, em um dos gráficos é apresentado uma vista em três dimensões do robô e no outro gráfico é possível alternar entre a vista lateral ou superior do PUMA.

As seções D e E fazem parte do requisito da geração de trajetória, sendo a seção D a entrada da posição inicial e da posição final através de uma matriz de transformação com

Figura 10 – Aba cinemática inversa



Fonte: Autoria própria (2022).

posição e orientação. Já a seção E apresenta os resultados obtidos através dos gráficos das posições, velocidades e acelerações na trajetória gerada pelo aplicativo, para cada uma das seis juntas do manipulador. Para a geração de trajetória, a seção C apresenta uma animação do movimento realizado pelo PUMA.

O segundo passo do desenvolvimento do aplicativo foi a implementação das cinemáticas direta e inversa e da geração de trajetória como funções do Matlab®. A última etapa do desenvolvimento contou com a junção das funções com a tela para implementar as funcionalidades do aplicativo.

4 RESULTADOS

Neste capítulo serão apresentados e discutidos os resultados obtidos. O foco do trabalho foi dado para a implementação da modelagem do robô PUMA 560 no *software* Matlab®, com as cinemáticas direta e inversa e a geração de trajetória, e para o desenvolvimento de um aplicativo com o *App Designer*, ferramenta presente no Matlab® para o desenvolvimento de aplicativos *Desktop*.

Para o desenvolvimento deste trabalho utilizou-se os valores da Tabela 1 para as variáveis trabalhadas.

Tabela 1 – Parâmetros para simulação

Parâmetro	Valor
l1	20 mm
l2	20 mm
Tempo	10 s
Passo	0,1 s

Fonte: Autoria própria (2022).

4.1 Cinemática direta

A primeira etapa na modelagem da cinemática direta foi o cálculo das transformações dos SR ${}^{i}_{i+1}T$. Com sete SR foram calculadas seis transformações, Equações (29) a (34).

$$ {}^{0}_{1}T = \begin{bmatrix} cos(\theta_1) & -sin(\theta_1) & 0 & 0 \\ sin(\theta_1) & cos(\theta_1) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{29} $$

$$ {}^{1}_{2}T = \begin{bmatrix} cos(\theta_2) & -sin(\theta_2) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -sin(\theta_2) & -cos(\theta_2) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{30} $$

$$ {}^{2}_{3}T = \begin{bmatrix} cos(\theta_3) & -sin(\theta_3) & 0 & l1 \\ -sin(\theta_3) & cos(\theta_3) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{31} $$

$$ {}^{3}_{4}T = \begin{bmatrix} cos(\theta_4) & -sin(\theta_4) & 0 & 0 \\ 0 & 0 & 1 & l2 \\ -sin(\theta_4) & -cos(\theta_4) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{32} $$

$$ {}^{4}_{5}T = \begin{bmatrix} cos(\theta_5) & -sin(\theta_5) & 0 & 0 \\ 0 & 0 & -1 & 0 \\ sin(\theta_5) & cos(\theta_5) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{33} $$

$$ {}^{5}_{6}T = \begin{bmatrix} cos(\theta_6) & -sin(\theta_6) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -sin(\theta_6) & -cos(\theta_6) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{34} $$

O passo seguinte se deu com o cálculo da transformação composta ${}^{0}_{6}T$, que define o SR {0} em relação ao SR {6}, Equação (35).

$$ {}^{0}_{6}T = {}^{0}_{1}T \cdot {}^{1}_{2}T \cdot {}^{2}_{3}T \cdot {}^{3}_{4}T \cdot {}^{4}_{5}T \cdot {}^{5}_{6}T = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{35} $$

Os elementos da matriz ${}^{0}_{6}T$ estão descritos nas Equações (36) a (47) e constituem a cinemática direta do PUMA 560. As equações que formam a matriz de rotação da cinemática direta são as seguintes:

$$ \begin{aligned} r_{11} = c_1 \cdot (c_{23} \cdot (c_4 \cdot c_5 \cdot c_6 - s_4 \cdot s_6) - s_{23} \cdot s_5 \cdot c_6) \cdots \\ \cdots - s_1 \cdot (s_4 \cdot c_5 \cdot c_6 + c_4 \cdot s_6) \end{aligned} \tag{36} $$

$$ \begin{aligned} r_{12} = s_1 \cdot (c_{23} \cdot (c_4 \cdot c_5 \cdot c_6 - s_4 \cdot s_6) - s_{23} \cdot s_5 \cdot c_6) \cdots \\ \cdots - c_1 \cdot (s_4 \cdot c_5 \cdot c_6 + c_4 \cdot s_6) \end{aligned} \tag{37} $$

$$ r_{13} = -s_{23} \cdot (c_4 \cdot c_5 \cdot c_6 - s_4 \cdot s_6) - c_{23} \cdot s_5 \cdot c_6 \tag{38} $$

$$r_{21} = c_1 \cdot (c_{23} \cdot (-c_4 \cdot c_5 \cdot c_6 - s_4 \cdot c_6) - s_{23} \cdot s_5 \cdot c_6) \cdots \\ \cdots - s_1 \cdot (c_4 \cdot c_6 - s_4 \cdot c_5 \cdot s_6) \tag{39}$$

$$r_{22} = s_1 \cdot (c_{23} \cdot (-c_4 \cdot c_5 \cdot c_6 - s_4 \cdot c_6) - c_{23} \cdot s_5 \cdot c_6) \cdots \\ \cdots - s_1 \cdot (c_4 \cdot c_6 - s_4 \cdot c_5 \cdot s_6) \tag{40}$$

$$r_{23} = -s_{23} \cdot (-c_4 \cdot c_5 \cdot s_6 - s_4 \cdot c_6) - c_{23} \cdot s_5 \cdot s_6 \tag{41}$$

$$r_{31} = -c_1 \cdot (c_{23} \cdot c_4 \cdot s_5 + s_{23} \cdot c_5) + s_1 \cdot s_4 \cdot s_5 \tag{42}$$

$$r_{32} = -s_1 \cdot (c_{23} \cdot c_4 \cdot s_5 + s_{23} \cdot c_5) + c_1 \cdot s_4 \cdot s_5 \tag{43}$$

$$r_{33} = s_{23} \cdot c_4 \cdot s_5 - c_{23} \cdot c_5 \tag{44}$$

Já as Equações (45), (46) e (47) representam o vetor de posição nas direções x, y e z:

$$p_x = c_1 \cdot (c_2 \cdot l_1 - s_{23} \cdot l_2) \tag{45}$$

$$p_y = s_1 \cdot (c_2 \cdot l_1 - s_{23} \cdot l_2) \tag{46}$$

$$p_z = s_2 \cdot l_1 - c_{23} \cdot l_2 \tag{47}$$

em que

$$c_{23} = c_2 \cdot c_3 - s_2 \cdot s_3 \tag{48}$$

$$s_{23} = c_2 \cdot s_3 + s_2 \cdot c_3 \tag{49}$$

Ao comparar essa modelagem da cinemática direta com a apresentada no livro de referência notou–se algumas divergências, as Equações (39), (40), (41), (42) e (45) apresentaram sinais opostos, já a Equação (36) possui variáveis diferentes. Para a prosseguimento do trabalho optou-se por continuar com a modelagem descrita acima, que apresentou resultados coerentes.

O desenvolvimento da cinemática direta no Matlab® se deu com a criação de uma função, Apêndice A, que recebe como parâmetros de entrada os ângulos das juntas e calcula a posição e orientação do robô. Essa função também faz o *plot* de uma representação simplificada do manipulador, com vista 3d e lateral. A Equação (51) exibe o resultado obtido por essa

função para uma entrada com os ângulos das juntas $\theta_{1:6} = 0$, Equação (50), posição inicial do PUMA.

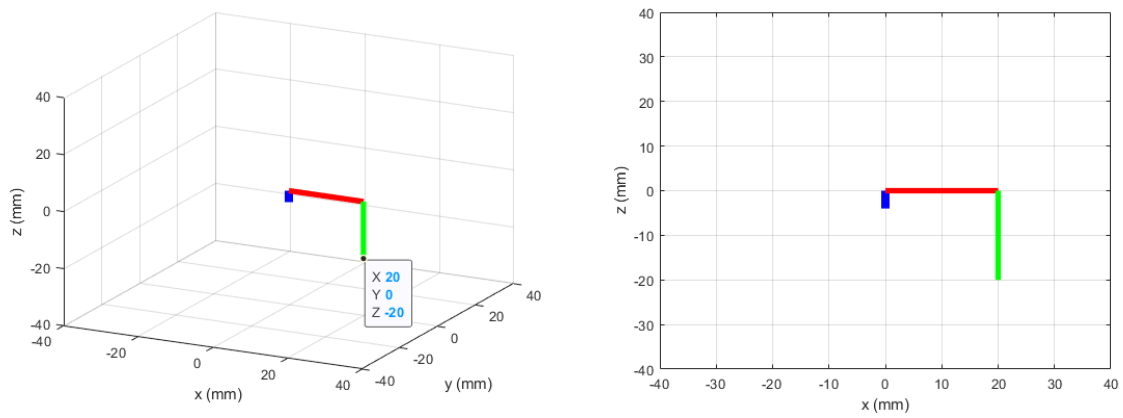
$$\theta_1 = 0;\ \theta_2 = 0;\ \theta_3 = 0;\ \theta_4 = 0;\ \theta_5 = 0;\ \theta_6 = 0; \tag{50}$$

$${}^{0}_{6}T = \begin{bmatrix} 1 & 0 & 0 & 20 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & -20 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{51}$$

As Figuras 11a e 11b são os *plots* obtidos com a cinemática direta da vista 3d e da vista lateral, respectivamente, da representação do PUMA.

Figura 11 – Plot posição inicial

(a) Vista 3d **(b) Vista Lateral**

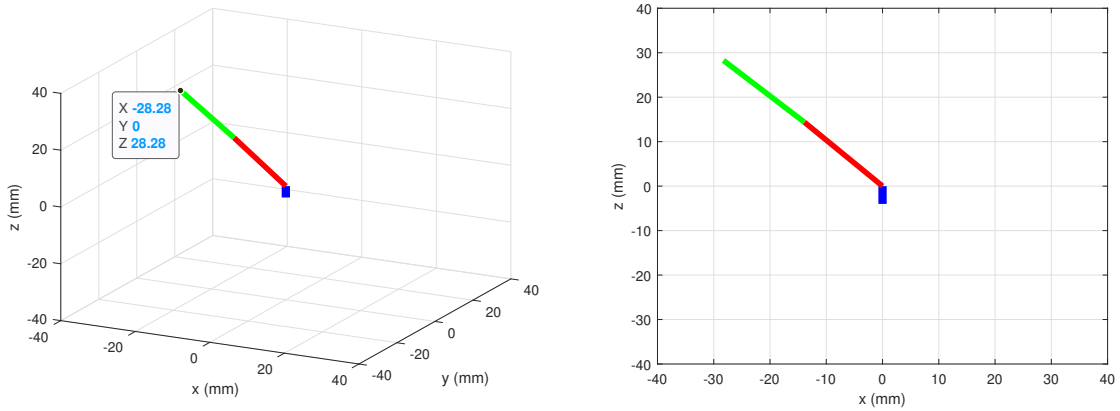


Fonte: Autoria própria (2022).

A Figura 12 representa o *plot* para uma segunda configuração de ângulos, apresentada na Equação (52), os quais foram a entrada da função da cinemática direta, e teve como resultado a Equação (53), matriz de transformação com posição e orientação do manipulador para essa configuração.

Figura 12 – Plot segunda posição

(a) Vista 3d **(b) Vista Lateral**



Fonte: Autoria própria (2022).

$$\theta_1 = 0;\ \theta_2 = -135;\ \theta_3 = -90;\ \theta_4 = 0;\ \theta_5 = 0;\ \theta_6 = 0; \tag{52}$$

$$ {}^{0}_{6}T = \begin{bmatrix} -0{,}707 & 0 & -0{,}707 & -28{,}28 \\ 0 & -1 & 0 & 0 \\ -0{,}707 & 0 & 0{,}707 & 28{,}28 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{53}$$

4.2 Cinemática inversa

Para a cinemática inversa implementou-se no Matlab® uma função, Apêndice B, que recebe como parâmetros de entrada uma matriz de transformação com a posição e orientação desejada. Ela calcula, com as equações da cinemática inversa, os ângulos que o robô deve assumir para chegar nessa posição e orientação.

Na Tabela 2 se têm alguns resultados obtidos com essa função, onde na primeira coluna estão representadas as entradas com a posição e orientação no formato de matriz de transformação ${}^{0}_{6}T$, já na segunda coluna a saída calculada pela função, com os ângulos das juntas em graus.

Tabela 2 – Resultado da cinemática inversa

Entrada (${}^0_6T$)	Saída (°)
$\begin{bmatrix} 1 & 0 & 0 & 20 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & -20 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\theta_1 = 0;\ \theta_2 = 0;\ \theta_3 = 0;\ \theta_4 = 0;\ \theta_5 = 0;\ \theta_6 = 0;$
$\begin{bmatrix} -0{,}707 & 0 & -0{,}707 & -28{,}28 \\ 0 & -1 & 0 & 0 \\ -0{,}707 & 0 & 0{,}707 & 28{,}28 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$\theta_1 = 180; \theta_2 = -45; \theta_3 = -90; \theta_4 = 180; \theta_5 = 0; \theta_6 = 0;$

Fonte: Autoria própria (2022).

As matrizes de transformações utilizadas para o cálculo da cinemática inversa, as entradas na Tabela 2, foram as matrizes obtidas na cinemática direta. Essa escolha foi feita para que houvesse uma análise entre as cinemáticas. Para a primeira entrada, Equação (51), resultou em uma mesma configuração de junta, Equação (50), nas duas cinemáticas.

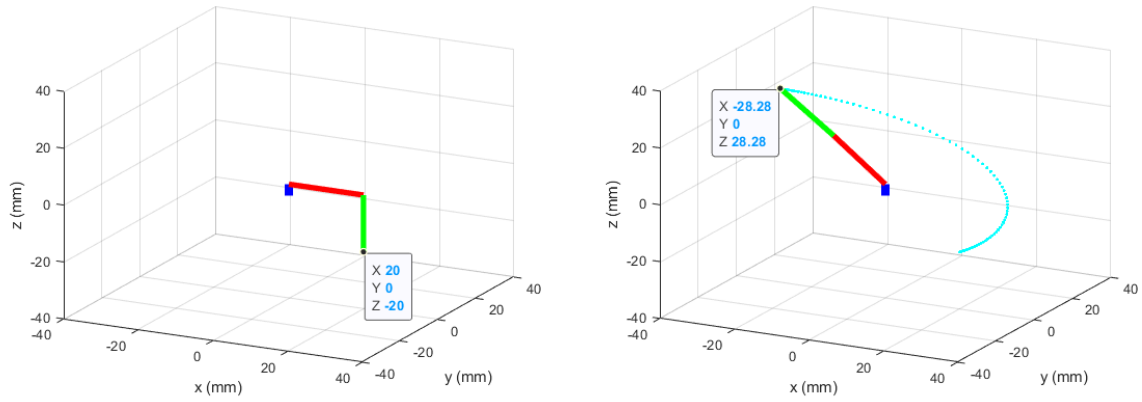
Já para segunda entrada, a cinemática inversa apresentou uma configuração diferente de ângulos da utilizada na cinemática direta, Equação (52), que gerou a matriz utilizada como entrada na inversa, Equação (51). Essa diferença deve-se ao fato de existirem diferentes configurações de juntas que o PUMA pode assumir para uma mesma posição e orientação.

4.3 Geração de trajetória

A geração de trajetória tem como entrada duas posições, a posição inicial e a final, o tempo que deve ser realizado o movimento e o passo para o cálculo do deslocamento que as juntas devem realizar para percorrer esse caminho. A função implementada no Matlab®, Apêndice C, recebe como parâmetro de entrada as posições como matriz de transformação e calcula através do polinômio cúbico o deslocamento das juntas, e tem como resultado as posições, velocidades e acelerações de cada junta para esse movimento, em um determinado tempo.

Figura 13 – Geração de trajetória

(a) Posição Inicial **(b) Posição Final**

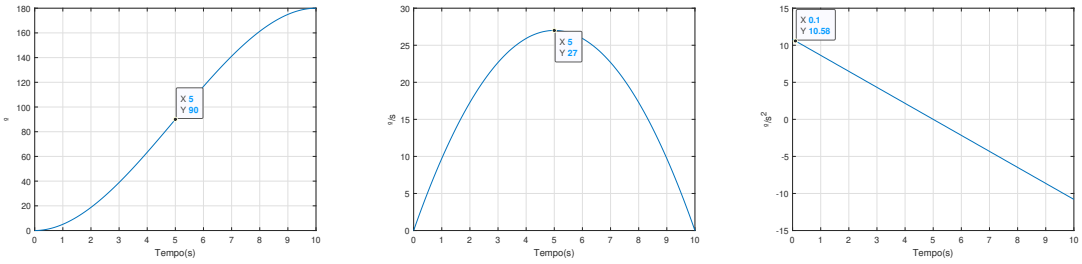


Fonte: Autoria própria (2022).

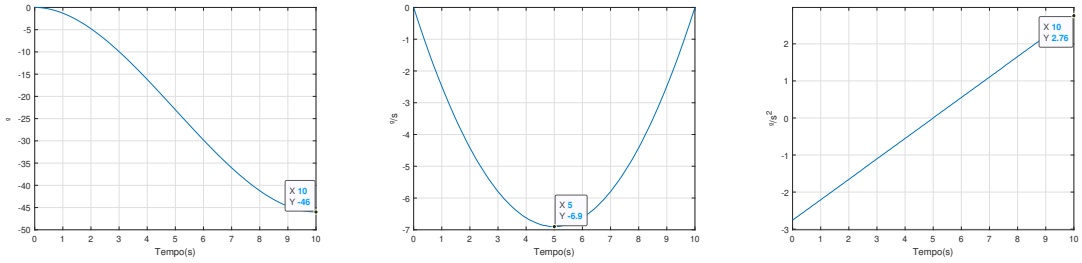
A Figura 13 mostra a posição inicial, Equação (51), e a posição final, Equação (53), para uma trajetória gerada por essa função. Em ciano, na Figura 13b, se tem todos os pontos calculados em que o robô irá passar nessa trajetória. Na Figura 14 estão os gráficos da posição, velocidade e aceleração obtidos para as três primeiras juntas.

Figura 14 – Trajetória dos θ_1, θ_2 e θ_3

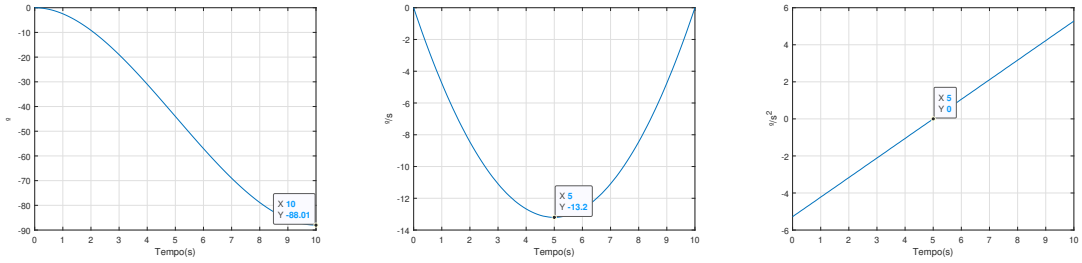
(a) Posiçãoθ_1 **(b) Velocidade θ_1** **(c) Aceleração θ_1**



(d) Posição θ_2 **(e) Velocidade θ_2** **(f) Aceleração θ_2**



(g) Posição θ_3 **(h) Velocidade θ_3** **(i) Aceleração θ_3**



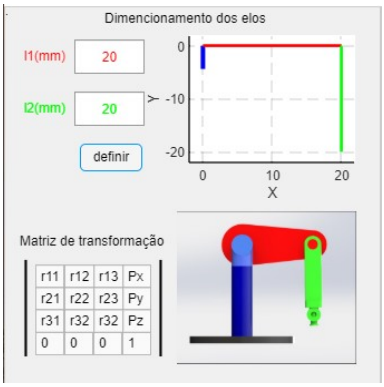
Fonte: Autoria própria (2022).

O polinômio cúbico fornece uma movimentação suave para a trajetória do robô, como é possível ver na Figura 14a, em que a junta θ_1 vai de 0° a 180° com uma partida e chegada à posição final de forma suave. Já a aceleração descreve uma forma linear, Figura 14c, sendo que na metade do tempo ela começa a desacelerar o movimento, o que faz com que a chegada à posição final seja suave.

4.4 Aplicativo

A etapa de desenvolvimento do aplicativo se deu com a junção das funções das cinemáticas e da geração de trajetória com a tela desenvolvida no *App Designer*. A primeira funcionalidade desenvolvida foi o dimensionamento dos elos, a Figura 15 apresenta o resultado para a entrada com $l_1 = 20\ mm$ e $l_2 = 20\ mm$.

Figura 15 – Dimensionamento dos elos

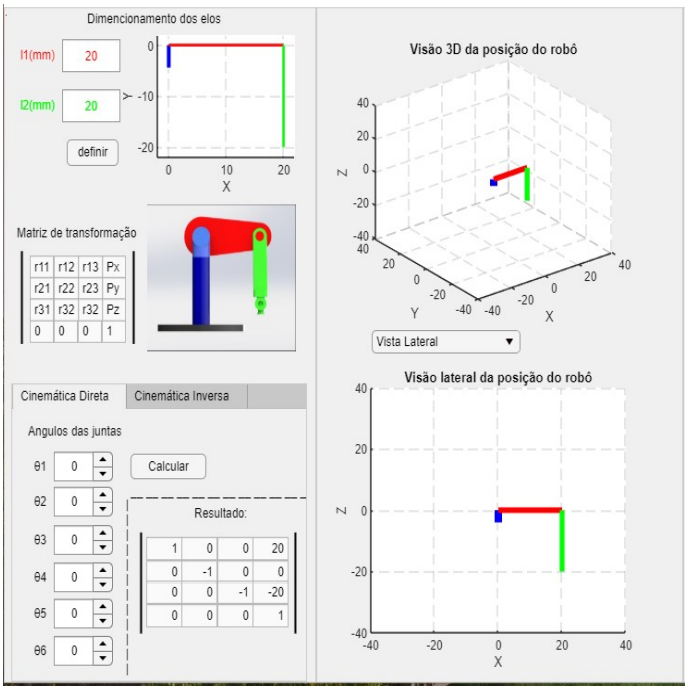


Fonte: Autoria própria (2022).

O dimensionamento permite a escolha do tamanho do robô a ser trabalhado, em que são aceitos valores no intervalo de 1 a 2000 mm para os dois elos do PUMA e tem como resultado uma visualização simplificada do robô com essas dimensões na posição inicial.

As cinemáticas foram implementadas no canto inferior esquerdo do aplicativo, separadas por abas, Figura 16. Na primeira aba está o cálculo da cinemática direta, que recebe os ângulos das juntas do robô como entrada e retorna a matriz de transformação com a posição e orientação do manipulador para essa configuração de ângulos. Além disso, o aplicativo apresenta uma representação da posição do PUMA para essa posição, nos gráficos do lado direito da Figura 16.

Figura 16 – Cinemática direta no aplicativo

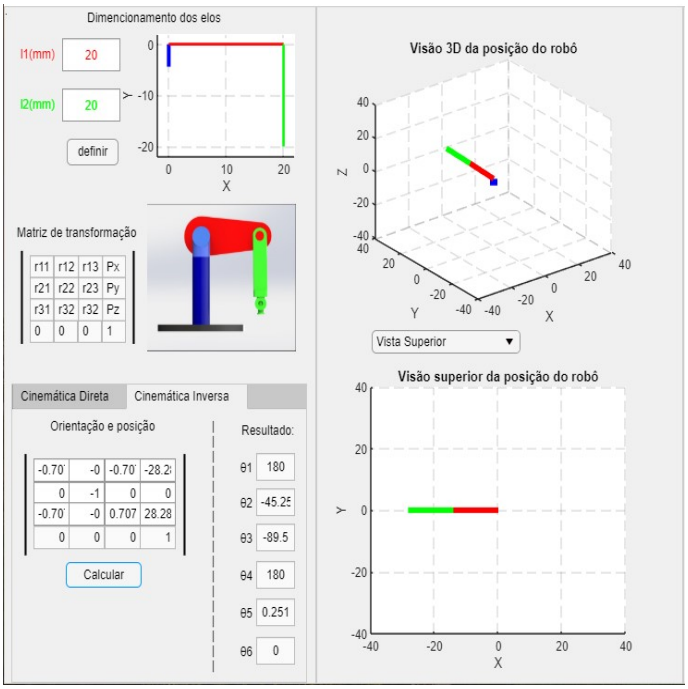


Fonte: Autoria própria (2022).

Para a cinemática inversa se tem a segunda aba, Figura 18, em que a entrada de dados são os parâmetros da matriz de transformação e o aplicativo calcula os ângulos que o PUMA

deve assumir para atingir essa determinada posição. O aplicativo também apresenta a representação dessa posição, a Figura 18 apresenta a vista 3d e a vista superior para o resultado da configuração de ângulos descritos na Equação (54), sendo a entrada a matriz de transformação da Equação (53).

Figura 17 – Cinemática inversa no aplicativo

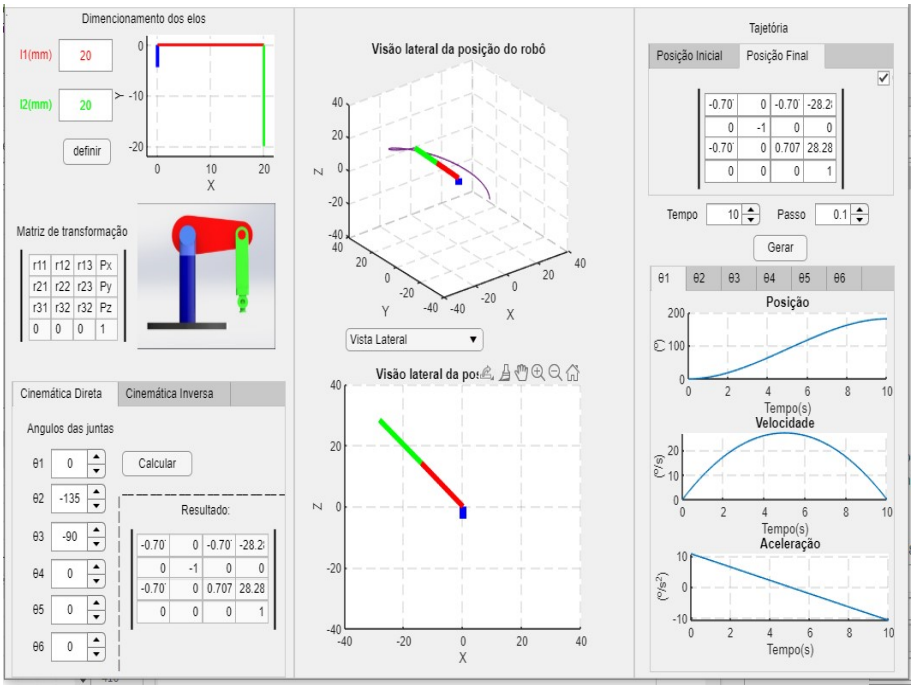


Fonte: Autoria própria (2022).

$$\theta_1 = 180;\ \theta_2 = -45{,}25;\ \theta_3 = -89{,}5;\ \theta_4 = 180;\ \theta_5 = 0{,}251;\ \theta_6 = 0; \tag{54}$$

A Figura 18 mostra a tela inteira do aplicativo desenvolvido, sendo que do lado direito está a geração de trajetória. No canto superior direito se tem as entradas necessárias, posição inicial e final, tempo do movimento e o passo a ser utilizado. Quando a trajetória é gerada, os gráficos na parte central da tela geram uma animação simplificada do movimento do PUMA, que foi calculado pelo aplicativo. Já no canto inferior direito é apresentado os gráficos da posição, da velocidade e da aceleração de todas as juntas, separados por seis abas.

Figura 18 – Geração de trajetória no aplicativo



Fonte: Autoria própria (2022).

O resultado obtido no aplicativo para a geração de trajetória para uma entrada com os valores da Tabela 3 são mostrados nas Figuras 18 e 19, sendo a segunda figura as abas do aplicativo para os gráficos das juntas.

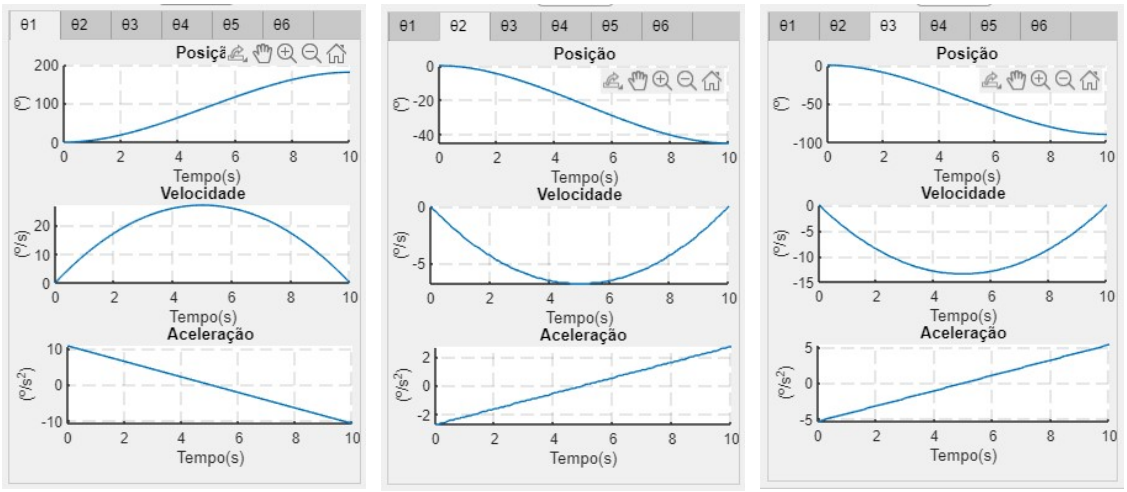
Tabela 3 – Entrada geração de trajetória no aplicativo

Posição inicial	$\begin{bmatrix} 1 & 0 & 0 & 20 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & -20 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
Posição final	$\begin{bmatrix} -0{,}707 & 0 & -0{,}707 & -28{,}28 \\ 0 & -1 & 0 & 0 \\ -0{,}707 & 0 & 0{,}707 & 28{,}28 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
Tempo	10 s
Passo	0,1

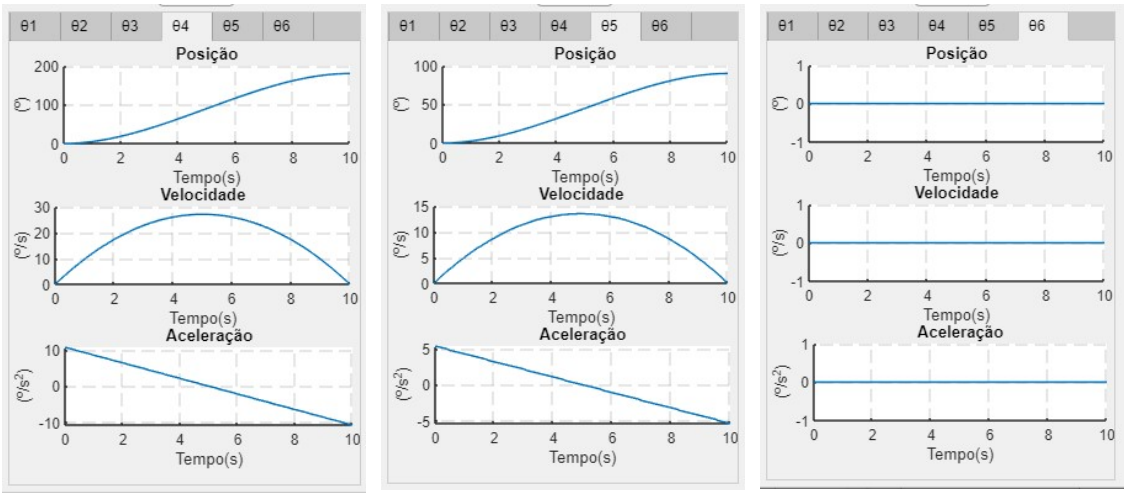
Fonte: Autoria própria (2022).

Figura 19 – Gráficos da posição, da velocidade e da aceleração

(a) θ_1 **(b)** θ_2 **(c)** θ_3



(d) θ_4 **(e)** θ_5 **(f)** θ_6



Fonte: Autoria própria (2022).

5 CONCLUSÃO

Este trabalho buscou desenvolver um aplicativo *desktop* para o modelo de robô PUMA 560, com o objetivo de analisar o uso do *software* Matlab® aplicado à robótica de forma genérica, sem a utilização de nenhuma *toolbox* de robótica. O aplicativo também buscou providenciar uma forma menos abstrata de realizar a modelagem de um robô em ambientes educacionais.

A primeira etapa do desenvolvimento se deu com a escolha do robô a ser trabalhado. O modelo PUMA 560 foi escolhido por sua versatilidade, importância para o desenvolvimento da robótica e por ser um modelo utilizado em alguns livros tradicionais de robótica como exemplo de cinemática.

Com o robô escolhido, seguiu-se para a modelagem da cinemática direta, a qual apresentou divergência com a cinemática apresentada no livro de referência Robótica de Jonh J. Craig(2012). Entretanto, para o prosseguimento do trabalho optou-se por continuar com a modelagem obtida neste trabalho, pois quando implementada no Matlab®, ela apresentou resultados coerentes. Com essa implementação foi possível obter uma representação gráfica da posição do robô.

Para a cinemática inversa utilizou-se a modelagem apresentada por CRAIG(2012), a qual foi implementada no Matlab® e apresentou resultados satisfatórios. A cinemática inversa foi a atividade que apresentou o maior grau de dificuldade para ser implementada, pois possui um sistema de equações não lineares.

A implementação da geração de trajetória foi feita com as equações de polinômio cúbico. Combinada com a cinemática direta, foi possível desenvolver uma animação para a trajetória gerada, desenvolveu-se, assim, uma representação visual do movimento do robô.

O aplicativo foi desenvolvido no *App Designer*, ferramenta disponível no Matlab®. Ele teve como requisito o cálculo das cinemáticas direta e inversa, a geração de trajetória e a representação visual do robô. Todos os requisitos e funcionalidades foram desenvolvidos no Matlab® e implementados no *App Designer*.

No desenvolvimento deste trabalho o Matlab® se mostrou ser uma ferramenta poderosa aplicado à robótica. Desde a modelagem, simulação das cinemáticas e do cálculo da trajetória até a criação do aplicativo *desktop* atendeu à todas as demandas.

Portanto a utilização do Matlab® para o ensino da robótica se provou importante. Entretanto é preciso ter conhecimento em programação de alto nível, deste modo, o Matlab® é mais recomendado ao ensino de nível superior de robótica.

Para futuros trabalhos, tem-se a modelagem da dinâmica e o desenvolvimento de um sistema controle para o PUMA 560. A implementação de outros modelos de robô também pode ser um caminho a ser seguido. Além disso, o Matlab® oferece *Toolbox* desenvolvidas especificamente para a robótica, que podem ser objetos de estudos.

REFERÊNCIAS

BULLER, L.; GIFFORD, C.; MILLS, A. **ROBOT**. New York: DK, 2018. 160 p. ISBN 9781465475848.

CORKE, P. **Robotics, Vision and Control**: Fundamental algorithms in matlab. 2nd.. ed. Gewerbestrasse: Springer, 2017. 693 p. ISBN 9783319544137.

CORTES, F. R. **Robótica**: Control de robots manipuladores. Ciudad de México: Alfaomega, 2011. 592 p. ISBN 9786077071907.

CORTES, F. R. **MATLAB aplicado a Robótica y Mecatrónica**. Ciudad de México: Alfaomega, 2012. 460 p. ISBN 9786077073574.

CRAIG, J. **Robøtica**: Tradução heloísa coimbra de souza. 3. ed. São Paulo: Pearson Education do Brasil, 2012. 379 p. Título original: Introduction to robotics: mechanics and control. ISBN 9783319544137.

GASPARETTO, A.; SCALERA, L. From the unimate to the delta robot: the early decades of industrial robotics. *In*: **Explorations in the History and Heritage of Machines and Mechanisms**. [*S.l.*]: Springer, 2019. p. 284–295.

IFR. **Executive Summary World Robotics 2021 Industrial Robots**. 2021. Disponível em: https://ifr.org/img/worldrobotics/Executive_Summary_WR_Industrial_Robots_2021.pdf. Acesso em: 05 out. 2022.

IFR. **Robot History**. 2021. Disponível em: https://ifr.org/robot-history. Acesso em: 07 out. 2022.

JAZAR, R. N. **Theory of Applied Robotics**: Kinematics, dynamics, and control. New York: Springer, 2007. 693 p. ISBN 9780387324753.

LYNCH, K. M.; PARK, F. C. **Modern Robotics**: Mechanics, planning, and control. Cambrigde: Cambridge University, 2017. 624 p. ISBN 9781107156302.

MATHWORKS. **Math. Graphics. Programming**. 2022. Disponível em: https://www.mathworks.com/products/matlab.html. Acesso em: 06 out. 2022.

MATHWORKS. **MATLAB App Designer**. 2022. Disponível em: https://www.mathworks.com/products/matlab/app-designer.html. Acesso em: 21 set. 2022.

OLIVEIRA, L.; SILVA, M.; MOREIRA, A. Agricultural robotics: A state of the art survey. *In*: **23rd International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines (CLAWAR)**. Moscow, Russia: [*s.n.*], 2020. p. 279–286.

OLIVEIRA, L. F. P. *et al.* Modeling, simulation and implementation of locomotion patterns for hexapod robots. *In*: **2020 IEEE Congreso Bienal de Argentina (ARGENCON)**. [*S.l.*: *s.n.*], 2020. p. 1–1.

ZILLI, S. d. R. A robótica educacional no ensino fundamental: perspectivas e prática. Florianópolis, SC, 2004.

APÊNDICE A – Função escrita em Matlab para a cinemática direta, usada pelo aplicativo

```
function td = direta(plotCima,plotBaixo,l1,l2,T1, T2, T3, T4, T5, T6,select);
% plotCima e plotBaixo apontam para onde os gráficos serão plotado
% Tn – parâmetro de entrada da função, ângulos das juntas em graus

   close all;
% a2,a3,d3,d4 – parâmetros de dimensionamento do PUMA
   a2 = l1;
   a3 = 0;
   d3 = 0;
   d4 = l2;
% tn – ângulos das juntas em rad
   t1 = T1 *pi/180;
   t2 = T2 *pi/180;
   t3 = T3 *pi/180;
   t4 = T4 *pi/180;
   t5 = T5 *pi/180;
   t6 = T6 *pi/180;
   P6 = [0 0 0 1]';
   R6 = [1 0 0; 0 1 0; 0 0 1;0 0 0];
   PR = [R6 P6];

% Tin – matriz de transformação i a n
   T56 = [cos(t6), -sin(t6), 0, 0; 0, 0, 1, 0; -sin(t6), -cos(t6), 0, 0; 0, 0, 0, 1];
   T45 = [cos(t5), -sin(t5), 0, 0; 0, 0, -1, 0; sin(t5), cos(t5), 0, 0; 0, 0, 0, 1];
   T34 = [cos(t4), -sin(t4), 0, a3; 0, 0, 1, d4; -sin(t4), -cos(t4), 0, 0; 0, 0, 0, 1];
   T23 = [cos(t3), -sin(t3), 0, a2; sin(t3), cos(t3), 0, 0; 0, 0, 1, d3; 0, 0, 0, 1];
   T12 = [cos(t2), -sin(t2), 0, 0; 0, 0, 1, 0; -sin(t2), -cos(t2), 0, 0; 0, 0, 0, 1];
   T01 = [cos(t1), -sin(t1), 0, 0; sin(t1), cos(t1), 0, 0; 0, 0, 1, 0; 0, 0, 0, 1];

   P03 = T01*(T12*(T23*P6));
% P – cinemática direta do PUMA
   P = T01*(T12*(T23*(T34*(T45*(T56*PR)))));
   aux = [0 P03(1,1) P(1,4);0 P03(2,1) P(2,4);0 P03(3,1) P(3,4)];
% td - retorno da função com a matriz de transformação
   td = round(P,3,'decimals');

% Plots das figuras 3d do aplicativo
   plot3(plotCima,[0,0],[0,0],[-l1*0.2,0],'LineWidth',6,'Color','blue');
   hold(plotCima);
```

```
    plot3(plotCima,aux(1,1:2),aux(2,1:2),aux(3,1:2),'LineWidth',4,"Color",'red');
    plot3(plotCima,aux(1,2:3),aux(2,2:3),aux(3,2:3),'LineWidth',4,"Color","green");
    hold(plotCima);

% Seleção da vista a ser plotada, vista lateral ou superior
    if select==1
        plot(plotBaixo,[0,0],[-l1*0.2,0],'LineWidth',6,'Color','blue');
        hold(plotBaixo);
        plot (plotBaixo,aux(1,1:2),aux(3,1:2),'LineWidth',4,'Color','red');
        plot (plotBaixo,aux(1,2:3),aux(3,2:3),'LineWidth',4,'Color','green');
        hold(plotBaixo);
    else
        plot (plotBaixo,aux(1,1:2),aux(2,1:2),'LineWidth',4,"Color",'red');
        hold(plotBaixo);
        plot (plotBaixo,aux(1,2:3),aux(2,2:3),'LineWidth',4,"Color",'green');
        hold(plotBaixo);
    end

end
```

APÊNDICE B – Função escrita em Matlab para a cinemática inversa, usada pelo aplicativo

```
function tethas = inversa(P,l1, l2)
% P – Matriz de transformação, posição e orientação desejada
% Atribuição do vetor de posição px,py e pz, da matriz de rotação r e das variáveis de
dimensionamento
   px = P(1,4);     py = P(2,4);     pz = P(3,4);
   r = [P(1,1) P(1,2) P(1,3); P(2,1) P(2,2) P(2,3); P(3,1) P(3,2) P(3,3)];
   a2 = l1;     a3 = 0;     d3 = 0;     d4 = l2;

% Calculo do θ1
   tetha1 = (atan2(py,px));

% Calculo do θ3
   K = (px^2 + py^2 + pz^2 - a2^2 - a3^2 - d3^2 - d4^2)/(2*a2);
   tetha3 = atan(2*a3/d4) - atan(K/sqrt(a3^2 + d4^2 - K^2));

% Calculo do θ2
   tetha23 = real(asin(((-a3 - a2*cos(tetha3))*pz + (cos(tetha1)*px
             + sin(tetha1)*py)*(a2*sin(tetha3) - d4)) / (pz^2 + (cos(tetha1)*px
             + sin(tetha1)*py) ^2)));
   tetha2 = (tetha23 - tetha3);
   x = cos(tetha1)*(cos(tetha2)*(l1 - l2*sin(tetha3)) - l2*cos(tetha3)*sin(tetha2));
   y = sin(tetha1)*(cos(tetha2)*(l1 - l2*sin(tetha3)) - l2*cos(tetha3)*sin(tetha2));
   z = - sin(tetha2)*(l1 - l2*sin(tetha3)) - l2*cos(tetha2)*cos(tetha3);

   if round(x,3,'decimals') ~= round(px,3,"decimals") || round(y,3,"decimals")
     ~= round(py,3,"decimals") || round(z,3,"decimals") ~= round(pz,3,"decimals")
       t2a = ((pi-tetha23) - (tetha3));
       x = cos(tetha1)*(cos(t2a)*(l1 - l2*sin(tetha3)) - l2*cos(tetha3)*sin(t2a));
       y = sin(tetha1)*(cos(t2a)*(l1 - l2*sin(tetha3)) - l2*cos(tetha3)*sin(t2a));
       z = -sin(t2a)*(l1 - l2*sin(tetha3)) - l2*cos(t2a)*cos(tetha3);
       if round(x,3,'decimals') ~= round(px,3,"decimals") || round(y,3,"decimals")
         ~= round(py,3,"decimals") || round(z,3,"decimals") ~= round(pz,3,"decimals")
       else
          tetha2 = t2a;
          tetha23 = (pi-tetha23);
       end
   end

% Calculo do θ4
```

```
    tetha4 = atan2((-r(1,3)*sin(tetha1))+r(2,3)*cos(tetha1) , (-r(1,3)*cos(tetha1)*cos(tetha23)
            - r(2,3)*sin(tetha1)*cos(tetha23) + r(3,3)*sin(tetha23)));

% Calculo do θ5
    s5 = -(r(1,3)*(cos(tetha1)*cos(tetha23)*cos(tetha4) + sin(tetha1)*sin(tetha4))
            +r(2,3)*(sin(tetha1)*cos(tetha23)*cos(tetha4)-cos(tetha1)*sin(tetha4))
            -r(3,3)*(sin(tetha23)*cos(tetha4)));
    c5 = (r(1,3)*(-cos(tetha1)*sin(tetha23)) + r(2,3)*(-sin(tetha1)*sin(tetha23))
            + r(3,3)*(-cos(tetha23)));
    tetha5 = atan2(s5,c5);

% Calculo do θ6
    s6 = (-r(1,1)*(cos(tetha1)*cos(tetha23)*sin(tetha4) - sin(tetha1)*cos(tetha4)) -
            r(2,1)*(sin(tetha1)*cos(tetha23)*sin(tetha4) + cos(tetha1)*cos(tetha4)) +
            r(3,1)*(sin(tetha23)*sin(tetha4)));
    c6 = (r(1,1)*((cos(tetha1)*cos(tetha23)*cos(tetha4) + sin(tetha1)*sin(tetha4))*cos(tetha5) -
            cos(tetha1)*sin(tetha23)*sin(tetha5)) + r(2,1)*((sin(tetha1)*cos(tetha23)*cos(tetha4) -
            cos(tetha1)*sin(tetha4))*cos(tetha5) - sin(tetha1)*sin(tetha23)*sin(tetha5)) -
            r(3,1)*(sin(tetha23)*cos(tetha4)*cos(tetha5) + cos(tetha23)*sin(tetha5)));
    tetha6 = atan2(s6,c6);

% Conversão dos θ de rad para graus
    t1 = tetha1*(180/pi);
    if (tetha2*(180/pi))<180
       t2 = (tetha2*(180/pi));
    else
       t2 = (tetha2*(180/pi))-360;
    end
    t3 = tetha3*(180/pi);
    t4 = tetha4*(180/pi);
    t5 = tetha5*(180/pi);
    t6 = tetha6*(180/pi);
% tehtas – θ calculados pela cinemática inversa, retorno da função
    tethas = [t1 t2 t3 t4 t5 t6];
    tethas = round(tethas,3,'decimals');
end
```

APÊNDICE C – Funções escritas em Matlab para a geração de trajetória, usada pelo aplicativo

```
% Função principal da geração de trajétoria
% Recebe como parâmetros os gráficos a serem plotados, matriz de transformação da posição
% inicial, Pi, e final, Pf, o tempo e passo da simulação
function a=trajetoria(plotsT,plotCima, plotBaixo, Pi,Pf,tempo,passo,l1,l2,select)

  tf=tempo;   T=passo;   t=0:T:tf;   ji=inversa(l1,l2,Pi);   jf=inversa(l1,l2,Pf);
     % Geração de trajetória com a função do polinômio cúbico
  for i=1:(tf/T+1);
     % dji -> posição junta i
     % ddji -> Velocidade junta i
     % dddji -> Aceleração junta i
     [dj1(i),ddj1(i),dddj1(i)]=polin(ji(1),jf(1),tf,t(i));
     [dj2(i),ddj2(i),dddj2(i)]=polin(ji(2),jf(2),tf,t(i));
     [dj3(i),ddj3(i),dddj3(i)]=polin(ji(3),jf(3),tf,t(i));
     [dj4(i),ddj4(i),dddj4(i)]=polin(ji(4),jf(4),tf,t(i));
     [dj5(i),ddj5(i),dddj5(i)]=polin(ji(5),jf(5),tf,t(i));
     [dj6(i),ddj6(i),dddj6(i)]=polin(ji(6),jf(6),tf,t(i));
  end


  % aplicativo
  plotMovimento(l1,l2,(tf/T+1),round(dj1,1,"decimals"),round(dj2,1,"decimals"),
     round(dj3,1,"decimals"),plotCima,plotBaixo,select,T);


  % juntas do robô
  plot(plotsT(1,1),t,round(dj1,1,"decimals"));
  plot(plotsT(1,2),t,ddj1);
  plot(plotsT(1,3),t,dddj1);

  plot(plotsT(2,1),t,round(dj2,1,"decimals"));
  plot(plotsT(2,2),t,round(ddj2,1,"decimals"));
  plot(plotsT(2,3),t,round(dddj2,1,"decimals"));

  plot(plotsT(3,1),t,round(dj3,1,"decimals"));
  plot(plotsT(3,2),t,round(ddj3,1,"decimals"));
  plot(plotsT(3,3),t,round(dddj3,1,"decimals"));

  plot(plotsT(4,1),t,round(dj4,1,"decimals"));
```

```
    plot(plotsT(4,2),t,round(ddj4,1,"decimals"));
    plot(plotsT(4,3),t,round(dddj4,1,"decimals"));

    plot(plotsT(5,1),t,round(dj5,1,"decimals"));
    plot(plotsT(5,2),t,round(ddj5,1,"decimals"));
    plot(plotsT(5,3),t,round(dddj5,1,"decimals"));

    plot(plotsT(6,1),t,round(dj6,1,"decimals"));
    plot(plotsT(6,2),t,round(ddj6,1,"decimals"));
    plot(plotsT(6,3),t,round(dddj6,1,"decimals"));
end


% Função polin, calculo das equações do polinômio cúbico
function [d,dd,ddd] = polin(di,df,tf,t)     a0=di;
    a1=0;
    a2=(3/(tf^2))*(df-di);
    a3=-(2/(tf^3))*(df-di);
    d=a0+a1*t+a2*t^2+a3*t^3;
    dd=a1+2*a2*t+3*a3*t^2;
    ddd=2*a2+6*a3*t;
end


% Função plotMovimento, gera uma animação para a geração de trajetória
function a = plotMovimento(l1,l2,T,t1,t2,t3,plotCima,plotBaixo,select,p)

    a=pi/180;     p=p/10
    for i=1:T
        P13(1,i)=l1*cos(t1(i)*a)*cos(t2(i)*a);
        P13(2,i)=l1*cos(t2(i)*a)*sin(t1(i)*a);
        P13(3,i)=-l1*sin(t2(i)*a);

        P14(1,i)=cos(t1(i)*a)*(cos(t2(i)*a)*(l1 - l2*sin(t3(i)*a)) - l2*cos(t3(i)*a)*sin(t2(i)*a));
        P14(2,i)=sin(t1(i)*a)*(cos(t2(i)*a)*(l1 - l2*sin(t3(i)*a)) - l2*cos(t3(i)*a)*sin(t2(i)*a));
        P14(3,i)= -sin(t2(i)*a)*(l1 - l2*sin(t3(i)*a)) - l2*cos(t2(i)*a)*cos(t3(i)*a);
    end

    b=-l1*0.2;
```

```
    for i=1:T
        plot3(plotCima,[0,0],[0,0],[b,0],'LineWidth',6,'Color','blue'); hold(plotCima);
        plot3(plotCima,[0,P13(1,i)],[0,P13(2,i)],[0,P13(3,i)],'LineWidth',4,'Color','red');
        plot3(plotCima,[P13(1,i),P14(1,i)],[P13(2,i),P14(2,i)],[P13(3,i),P14(3,i)],'LineWidth',4,
              'Color','green');
        hold(plotCima);
        if select == 1
            plot(plotBaixo,[0,0],[b,0],'LineWidth',6,'Color','blue');
            hold(plotBaixo);
            plot(plotBaixo,[0,P13(1,i)],[0,P13(3,i)],'LineWidth',4,'Color','red');
            plot(plotBaixo,[P13(1,i),P14(1,i)],[P13(3,i),P14(3,i)],'LineWidth',4,'Color','green');
            hold(plotBaixo);
        else
            plot(plotBaixo,[0,P13(1,i)],[0,P13(2,i)],'LineWidth',4,'Color','red');
            hold(plotBaixo);
            plot(plotBaixo,[P13(1,i),P14(1,i)],[P13(2,i),P14(2,i)],'LineWidth',4,'Color','green');
            hold(plotBaixo);
        end
        pause(p);
    end

    hold(plotCima);
    plot3(plotCima,P14(1,:),P14(2,:),P14(3,:),'-');
    hold(plotCima);
end
```