MANIPULADOR ROBÓTICO COM DETECÇÃO VISUAL INTEGRADA

Relatório Parcial do Projeto RAJA

Autores:
Anderson Queiroz do Vale
Aziel Martins de Freitas Júnior
Jean Paulo Silva
Rodrigo Formiga Farias

Facilitadores:
Rebeca Tourinho Lima
Marco Antônio dos Reis

Salvador

Bahia, Brasil

Abril de 2020

Título: MANIPULADOR ROBÓTICO COM DETECÇÃO VISUAL INTEGRADA	
PROD. TEC. BIR - 001 / 2020	Versão
Classificação: ( ) Confidencial (X) Restrito ( ) Uso Interno ( ) Público	01

Informações Confidenciais - Informações estratégicas para o BIR e Senai Cimatec. Seu manuseio é restrito a usuários previamente autorizados pelo Gestor da área.

Informações Restritas - Informação cujo conhecimento, manuseio e controle de acesso devem estar limitados a um grupo restrito de pesquisadores que necessitam utilizá-la para exercer suas atividades profissionais.

Informações de Uso Interno - São informações destinadas à utilização interna por pesquisadores e parceiros.

Informações Públicas - Informações que podem ser distribuídas ao público externo, o que, usualmente, é feito através dos canais apropriados.

Dados Internacionais de Catalogação na Publicação (CIP)

Anderson Queiroz do Vale
Aziel Martins de Freitas Júnior
Jean Paulo Silva
000 Rodrigo Formiga Farias

Rebeca Tourinho Lima
Marco Antônio dos Reis

MANIPULADOR ROBÓTICO COM DETECÇÃO VISUAL INTEGRADA
Salvador
Bahia, Brasil
Abril de 2020

Keywords:
1. Manipulator. 2. Simulation. 3. Computer vision.

000

SUMÁRIO EXECUTIVO

O projeto de Manipuladores - Desafio.2, também conhecido como **RAJA Robotics** se configura sob o Programa de Formação de Novos Talentos do Serviço Nacional de Aprendizagem Industrial, Departamento Regional da Bahia - Senai/DR/BA, sendo este o principal fomentador do programa.

O projeto teve início no dia 27 de fevereiro de 2020 e o prazo de execução foi de 50 dias.

RESUMO

Este trabalho tem por finalidade desenvolver um manipulador robótico automatizado capaz de localizar um marcador e realizar a ação de pressionamento de um botão. Para tanto, utiliza-se da detecção e identificação de marcadores fiduciais, do estudo da cinemática direta e inversa, a fim de planejar e executar o movimento do manipulador, e do estudo da ferramenta *MoveIt* a fim de conseguir alcançar um ponto de interesse.

ABSTRACT

This work aims to develop an automated robotic manipulator capable of locating a marker and performing the action of pressing a button. For that, it uses the detection and identification of fiducial markers, the study of direct and inverse kinematics, in order to plan and execute the manipulator's movement, and the study of the MoveIt tool in order to reach a point of interest.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTRA DE SÍMBOLOS E ABREVIATURAS

AR *Augmented Reality*

ARUCO *Augmented Reality University of Cordoba*

CAD *Computer Aided Design*

DC *Direct Current*

GUI *Graphic User Interface*

PWM *Pulse Width Modulation*

QR *Quick Response*

ROS *Robot Operating System*

RViz *ROS Visualizer*

SCARA *Selective Compliance Articulated Robot Arm*

URDF *Universal Robotic Description Format*

XML *Extensible Markup Language*

D-H *Denavit-Hartenberg*

OpenCV *Open Source Computer Vision Library*

XACRO XML *macros*

FMECA *Failure Mode and Effect Criticality Analysis*

SUMÁRIO

1 INTRODUÇÃO

A robótica é um campo relativamente jovem da tecnologia moderna que atravessa os limites da engenharia tradicional (SPONG; HUTCHINSON; VIDYASAGAR, 2005). O estudo da robótica é um ramo da tecnologia que engloba área de Mecânica, Eletrônica e Computação, com graus de teoria de controle, microeletrônica, inteligência artificial, fatores humanos e de produção (PIMENTA, 2009). Segundo (ERTHAL, 1992a), o crescimento da robótica na indústria é justificado em face das exigências de maior qualidade, produtividade e flexibilidade nos processos fabris. Na área industrial, a robótica evoluiu devido ao aumento de uso de robôs e manipuladores industriais.

O estudo do desenvolvimento de manipuladores robóticos foi iniciado por volta de 1954 com George Devol, quando foi desenvolvido o primeiro robô programável e desde então grandes desenvolvimentos nessa área foram atingidos. Como resultado desse avanço, o investimento de empresas mundiais em torno de 16,5 bilhões de dólares em 2018, chegando a marca de 420 mil unidades enviadas globalmente, com perspectiva de crescimento médio de 12% ao ano entre 2020 e 2022. (INDUSTRIAL. . . , 2019). O investimento de empresas mundiais em torno de 16,5 bilhões de dólares em 2018, chegando a marca de 420 mil unidades enviadas globalmente, com perspectiva de crescimento médio de 12% ao ano entre 2020 e 2022. (INDUSTRIAL. . . , 2019).

Um manipulador robótico consiste em um dispositivo mecânico composto de elementos rígidos (elos) que proporcionam a sustentação e alcance do braço. A inevitabilidade de apresentar algum grau de flexibilidade faz com que esses elos necessitem ser projetados para apresentar elevada rigidez aos esforços que o manipulador será submetido. Esses elos são conectados entre si através de articulações (juntas), que oferecem graus de liberdade ao manipulador e controle de movimento relativo entre os elos. Essas juntas podem ser basicamente divididas em dois grupos: juntas prismáticas e juntas de rotação. Neste projeto foi utilizado as juntas de rotação. A disposição dessas juntas, determina a classificação dos manipuladores, sendo classificados como: robô de coordenadas cartesianas, de coordenadas esféricas, *Selective Compliance Articulated Robot Arm* (SCARA), entre outros (ROMANO, 2002).

O presente trabalho consiste no desenvolvimento de um manipulador robótico antropomórfico de cinco graus de liberdade, com capacidade de detecção e aquisição de dados. Para isso, será abordado conceitos sobre cinemática direta e inversa, arquisição e processamentos de dados através de uma câmera e conceitos relacionado ao *Robot Operating System* (ROS). Este projeto foi realizado no Laboratório de Robótica e Sistemas Autônomos do SENAI CIMATEC.

1.1 Objetivos

O objetivo deste projeto é a construção, simulação e a aplicação de um manipulador robótico antropomórfico automatizado, com a finalidade de acionar uma botoeira de emergência através da leitura de um marcador quadrado, que possui um identificador, chamado *Augmented Reality University of Cordoba* (ARUCO). A modelagem estrutural e matemática, bem como o dimensionamento e criação de um robô compatível com o *framework* do *ROS*, foram etapas precedentes à simulação.

1.2 Justificativa

Dados da (INTERNATION..., 2018) monstram que o ritmo da automação industrial está acelerando em grande parte do mundo desenvolvido. Ele ainda aponta que nas américas a taxa de crescimento atingiu 20% a mais do que no ano anterior. O Brasil, segundo (EXECUTIVE..., 2019), terá um crescimento de 10% no número unidades de robôs para os anos de 2019 até 2021. O Brasil em 2017 tinha 12.373 robôs e que esse número é muito inferior se compara com a China que possui 473.429, que lidera o *rank* (BRASIL..., 2019). Segundo (BRASIL..., 2019), em 2019 o Brasil possui em torno de 16 mil robôs, sendo a maioria no setor automobilístico. Dessa forma, o Brasil precisa de mais capacitação para o evoluir desenvolvimento da robótica e poder assim se aproximar das grandes potênciais mundiais.

Na indústria, a utilização de um braço robótico vem da necessidade de aumentar a produtividade e melhorar a qualidade dos produtos. No campo acadêmico é uma ferramenta que auxilia nos processos de ensino e pesquisa, além de aplificar os conceitos de robótica sobre problemática da movimentação no espaço tridimensional, modelos matemáticos da cinemática e da dinâmica de movimento e aspectos de controle de movimento.

1.3 Organização do relatório

O trabaho está divido da segunite forma: O Capitulo 2 apresenta os conceitos do sistema de um manipulador mecânico, os requisitos do cliente e técnicos, toda teoria relacionado aos manipuladores e o ambiente de operação; O Capitulo 3 denota o desenvolvimento do sistema, detalhando a descrição do sistema, as especificações funcionais, a arquitetura de software, simulação do sistema e os teste realizados; O Capitulo 4 aponta a confiabildade do sistema; O Capitulo 5 apresenta as licções aprendidas; POr fim, o Capitulo 6 apresenta as conclusões finais.

2 CONCEITO DO SISTEMA

Neste capitulo serão apresentados os principais conceitos relacionado aos manipuladores mecânico.

A norma técnica (ISO-8373, 2012) criada para padronizar o vocabulário referente aos robôs e dispositivos robóticos operando em ambientes industriais e não industriais, define o manipulador como uma máquina na qual o mecanismo, geralmente, consiste em uma série de segmentos, articulados ou deslizantes entre si, com o objetivo de empunhar e/ou mover objetos (peças ou ferramentas) em vários graus de liberdade. Em outras palavras, um manipulador é um equipamento baseado em atuadores programável, com um certo grau de liberdade, projetado para realizar uma variedade de atividades, assim como realização de diversos processos industriais (ISO-8373, 2012).

2.1 Parâmetros básicos

Um manipulador robótico é constituído pela combinação de alguns elementos fundamentais para seu funcionamento, sustentação, movimentação e interação com o meio em que está inserido (ROMANO, 2002). Para isso, ainda (ROMANO, 2002), divide-se sua estrutura em cinco componentes principais: manipulador mecânico, atuadores, sensores, unidade de controle, unidade de potência e efetuador. O detalhamento desses conceitos serão descritos a seguir.

2.1.1 Manipulador mecânico

Esse componente refere-se ao aspecto mecânico e estrutural do manipulador, sendo composto por elementos rígidos (corpos ou elos) que são projetados de forma a suportar esforços de flexão e torção. Esses elos, comumente, são compostos de perfis de alumínio ou aço, mas também podem ser compostos de materiais como plásticos reforçados, fibras de carbono ou fibras de vidro em aplicações mais nobres (ROMANO, 2002). Os elos são conectados através de articulações, que permitem girar e controlar o movimento relativo entre os elos. As articulações também determinam os graus de liberdade que um robô apresenta. Por último, a parte estrutural possui o sistema de transmissão de potência mecânica originada dos atuadores. O sistema de transmissão depende do tipo do projeto e como a potência será transmitida aos elos.

2.1.2 Atuadores

São componentes que enviam potência mecânica aos elos para assim permitir a movimentação. Os atuadores podem ser do tipo hidráulicos, pneumáticos ou eletromagnéticos. Esses atuadores apresentam grande eficiência de torque em diferentes faixas de velocidade

e alguns deles possuem sensores de posição angular e de velocidade para controle de posicionamento, permitindo assim um melhor controle da posição e orientação do efetuador (WEBER et al., 2001).

2.1.3 Sensores

Sensores são dispositivos que podem ser adicionados a um manipulador para conceber sentidos e percepção. Um exemplo disso é a capacidade de perceber um espectro além do visível ao olho humano. Sensores são transformadores de energia e convertem alguma grandeza física em um sinal elétrico (RUOCCO, 2013). São usados para influenciar o comportamento do manipulador, conferindo a capacidade de avaliar posição e velocidade dos elos ou obter características e informações do ambiente para poder realizar o planejamento de trajetórias e de forma autônoma realizar as atividades determinadas.

2.1.4 Unidade de controle

O gerenciamento e monitoração dos parâmetros de operação para realizar as atividades são realizados na unidade de controle. Os comandos de movimentação enviados aos atuadores são originados através de uma integração da unidade de controle e os sensores.

2.1.5 Unidade de potência

A energia necessária para a movimentação dos atuadores e consequentemente das juntas é proveniente da unidade de potência.

2.1.6 Efetuador

O efetuador é a ferramenta do robô, sendo chamado pela comunidade da robótica de *"end effector"*. Esse elemento, importante para realização da tarefa do manipulador, é um dispositivo que se localiza no final do braço robótico e é responsável por interagir com o ambiente. Aspersores de tinta e geradores de arco de solda são exemplos comuns na indústria.

2.2 Requisitos do cliente

Requisitos predeterminados pelo cliente consistem em exigências de funcionamento que devem ser observadas ao final do projeto, para que se considere um sucesso a concepção deste. Para tal, foram determinadas algumas característica desejáveis no projeto:

1. O manipulador deve estar acoplado em uma base fixa;

2. Ele deve acionar uma botoeira de emergência posicionada a uma distância de

aproximadamente um metro da base do manipulador;

3. A posição e orientação da botoeira serão estimadas a partir da leitura de uma *tag*;

4. Deverão ser utilizados câmera Teledyne Genie Nano C2590 e motores Dynamixel.

2.3 Requisitos técnicos

Os requisitos técnicos de um projeto são especificações necessárias para o funcionamento esperado do projeto. Podem ser sobrepostos aos requisitos do cliente em caso de conflito entre o esperado pelo cliente e o necessário para que o projeto seja bem sucedido, objetivando manter o projeto o mais eficiente dentro do escopo planejado. Os requisitos foram:

1. Apresentar pelo menos quatro graus de liberdade;

2. Não ultrapassar carga máxima de 19 N no efetuador;

3. Utilizar *MoveIt* para planejar a trajetória do manipulador;

4. Utilizar ROS;

5. Capacidade de identificar a *tag* e calcular toda trajetória até o acionamento do botão.

2.4 Fundamentação Teórica

O primeiro manipulador industrial, conhecido como *UNIMATE*, foi instalado em 1961 numa planta da *General Motors* em Nova Jersey (IEEE. . . , s.d). Desde então vários equipamentos puderam ser agregados aos manipuladores, concedendo-lhes utilizações diversas em robôs industriais no processo produtivo, mas também expandindo-os a vários âmbitos, o que possibilitou avanços constantes nas áreas de mecânica, eletrônica digital, ciência da computação entre outras.

2.4.1 Robô Manipulador

Manipuladores podem ser teleoperados ou autônomos. São comumente utilizados em tarefas que apresentam riscos aos operadores, por repetitividade demasiada, ou onde se exige uma combinação de rapidez e precisão além capacidade humana. Soldagem, pintura, montagem, manipulação de materiais perigosos e inspeção de ambientes inóspitos são mais eficientemente realizadas por robôs. Segundo (CARRARA, 2015), os manipuladores podem ser subdivididos quanto a estrutura mecânica em cinco grupos: Robô de coordenadas cartesianas; Robô de coordenadas cilíndricas; Robô de coordenadas esféricas; Robô SCARA e Robô antropomórfico.

2.4.1.1 Robô de coordenadas cartesianas

Este robô possui três juntas prismáticas, resultando em movimentos de translação. Tal nomenclatura deve-se ao movimento coincidente com um sistema cartesiano de coordenadas espaciais. Ele possui um *workspace*, ou seja, um volume de trabalho limitado ao formato de um retângulo, como mostrado na figura 1.

Figura 1: Volume de trabalho de robô cartesiano.

Fonte: (COLLINS, 2018).

2.4.1.2 Robô de coordenadas cilíndricas

A configuração do movimento deste robô é descrita num sistema de coordenadas cilíndricas. É constituído de uma junta de rotação e duas juntas prismáticas, gerando um volume de trabalho limitado por superfícies de simetria cilíndrica como na figura 2.

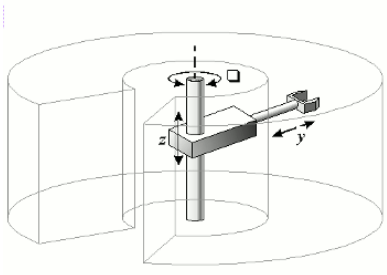


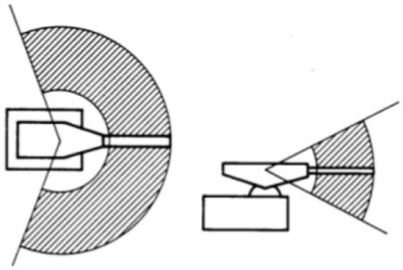
Figura 2: Volume de trabalho de robô de coordenadas cilíndricas.

Fonte: (ALL. . . , s.d.)

2.4.1.3 Robô de coordenadas esféricas

O movimento dos eixos desse robô formam um sistema de coordenadas de referência polar, com uso de duas juntas de rotação e apenas uma junta prismática responsável pelo movimento de translação. Neste robô o volume de trabalho está limitado por surperfícies de simetria esférica, como mostra a figura 3.

Figura 3: Áreas de trabalho de robô de coordenadas esféricas.

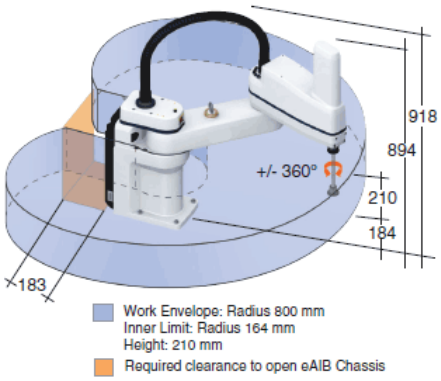


Fonte: (C3, 2015)

2.4.1.4 Robô SCARA

O robô SCARA, assim como o robô de coordenadas esféricas, apresenta duas juntas rotacionais e uma junta prismática. Esse robô é muito utilizado em pequenas atividades e montagem de pequenos componentes. Seu volume de trabalho, como mostra a figura 4, é gerado de forma aproximadamente cilíndrica.

Figura 4: Volume de trabalho de robô SCARA.

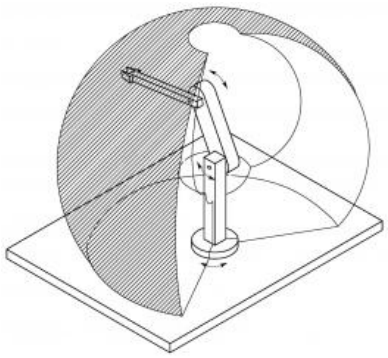


Fonte: (AUTOMATION, 2016)

2.4.1.5 Robô antropomórfico

O objeto de estudo deste projeto se baseia em um robô do tipo antropomórfico, que tem como configuração ao menos três juntas rotacionais. Esse manipulador possibilita diversas posições e orientações do seu efetuador devido a sua maior mobilidade, formando um volume de trabalho mais complexo em comparação as demais configurações. O volume de trabalho desse robô pode ser visto na figura 5.

Figura 5: Volume de trabalho de robô antropomórfico.



Fonte: (FEDERICA. . . , s.d.)

2.4.2 Motor

Um motor é um dispositivo que converte outras formas de energia em energia mecânica, de forma a impelir movimento a uma máquina ou veículo (WIKIPéDIA, 2020). Os tipos de motores empregados em manipuladores seguem abaixo, em ordem de utilização, da menor para a maior.

2.4.2.1 Motor de passo

Motores de passo são motores *Direct Current* (DC) cujo rotor gira em incrementos angulares discretos quando os enrolamentos do estator são energizados de maneira programada (ATHANI, 2005). São de fácil controle de velocidade e posição, e num primeiro momento são bastante previsíveis, pois se assume que a quantidade de passos dada é a quantidade especificada, a menos que ocorra um deslizamento das juntas por influência de forças externas. Isso demanda a presença de sensoriamento para o correto posicionamento dos elos. Além disso, não é possível controlar o torque entre passos, e comparado aos motores que seguirão, sua potência é inferior. Utilizado em poucas aplicações (WEBER et al., 2001).

2.4.2.2 Motor de corrente contínua

Conhecidos por motores DC são controlados por tensão, tipicamente por meio de modulação por largura de pulso, ou *Pulse Width Modulation* (PWM). O seu controle de velocidade é dependente de sensoriamento, tipicamente realizado por um encoder colocado no eixo do motor. São motores baratos mas não oferecem altas potências. Utilizam escovas para a conexão elétrica entre o rotor e a fonte de energia (WEBER et al., 2001).

2.4.2.3 Motor de corrente contínua *brushless*

Não possuem as escovas dos motores DC. Tipicamente utilizam sensores de efeito Hall na medição da posição do rotor. Motores brushless tem desempenho melhor que motores DC com escovas, mas são mais caros (WEBER et al., 2001).

2.4.2.4 Motor de corrente alternada

Motores de corrente alternada são os mais robustos levando em conta o custo benefício porém são mais difíceis de controlar por conta das suas características eletromagnéticas não-lineares, necessitando o uso de inversores de frequência. Seu comportamento não-linear oportuniza o uso de técnicas de controle não-linear. São usados amplamente em aplicações industriais (WEBER et al., 2001).

2.4.3 Cinemática

A cinemática de um manipulador é o estudo do conjunto de relações entre as posições, velocidades e acelerações de seus elos (SANTOS, 2004). A cinemática é usualmente dividida em duas partes: cinemática direta e cinemática inversa. Segundo (NAKAMURA, 1986), a cinemática possui dois problemas a estudar: a cinemática de posição e a diferencial. A cinemática de posição trata da relação entre as coordenadas de juntas e posição do efetuador, enquanto que a cinemática diferencial trata da relação entre os deslocamentos infinitesimais das juntas e os do efetuador.

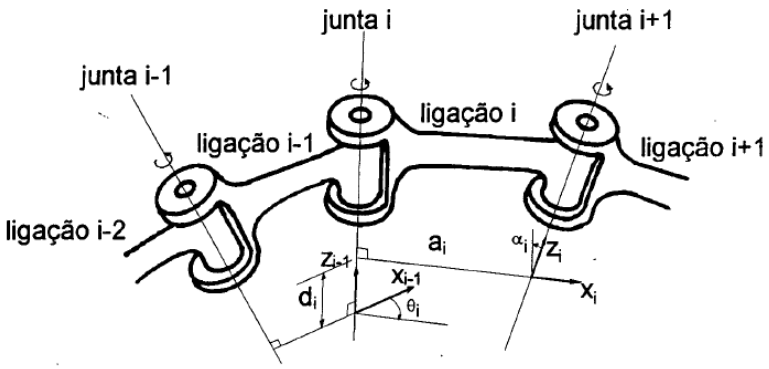
2.4.3.1 Cinemática direta

Na cinemática direta visa-se obter a posição e orientação no espaço cartesiano (x, y, z) em relação ao sistema fixo de coordenadas, a partir dos ângulos das juntas do manipulador (SANTOS, 2016). Ela determina qual a posição e orientação do efetuador, sendo conhecidas as posições das juntas do robô.

As técnicas mais comumente utilizadas para a obtenção da equação cinemática baseiam-se na geometria do robô ou em equações algébricas (ERTHAL, 1992b). Para este trabalho foi utilizado a notação *Denavit-Hartenberg* (D-H) com as matrizes homogêneas. Por esse motivo, será descrito somente essa técnica para obter a solução da cinemática direta.

Segundo (ERTHAL, 1992b), a formulação utilizando matrizes homogêneas juntamente com a notação de D-H é a mais utilizada por permitir estudos tanto em cinemática quanto em dinâmica. Os sistemas de coordenadas das ligações são fixados ao longo das juntas de modo a facilitar a definição das rotações e translações (ERTHAL, 1992b). As origens dos sistemas das ligações são localizadas na normal comum entre as juntas que a limita, como mostra a figura 6

Figura 6: Modelagem utilizando matrizes homogêneas e a notação de D-H.



Fonte: (ERTHAL, 1992b)

Dessa forma, segundo (SLOTINE HARUHIKO ASADA, 1986), a transformação do sistema $i-1$ para o sistema i é composta por quatro transformações elementares agrupadas numa mesma matriz, como mostra a equação 2.1.

$$A_i^{i-1} = \begin{bmatrix} cos\theta_i & -sen\theta_i cos\alpha_i & sen\theta_i sen\alpha_i & a_i cos\theta_i \\ sen\theta_i & cos\theta_i cos\alpha_i & -cos\theta_i sen\alpha_i & a_i sen\theta_i \\ 0 & sen\alpha_i & cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.1}$$

Como pode ser visto na equação 2.1, i representa o índice da junta, θ_i o ângulo da junta i, d_i o deslocamento da junta i, a_i o comprimento da ligação i e α_i a torção da ligação i. Os parâmetros θ_i, d_i, a_i e α_i são denominados parâmetros de D-H. Três desses parâmetros são constantes e apenas um é variável (θ_i ou d_i), conforme o tipo de junta seja prismatica ou rotativa (ERTHAL, 1992b).

O algoritmo de D-H permite obter os sistemas de coordenadas e as transformações associadas a cada elo de manipulador (SANTOS, 2004). É utilizado principalmente quando há uma complicação na análise do manipulador por ter muitos graus de liberdade. O cálculo da cinemática direta é feito pelas transformações sucessivas entre os sistemas de coordenadas desde a base do robô até o efetuador, quando considerado que os parâmetros já são conhecidos. Isso é obtido pelas multiplicações sucessivas das matrizes A_i^{i-1} da equação 2.1, fazendo i variar de 1 até N, em que N é número total de juntas. Deste modo, a equação da cinemática direta é representada pela equação 2.2.

$$T_N = A_1^0(\theta_1) \times A_2^1(\theta_2), ..., A_N^{i-1}(\theta_N) \tag{2.2}$$

2.4.3.2 Cinemática Inversa

A obtenção da cinemática inversa é mais complexa que a direta por se tratar da resolução de um sistema de equações não lineares. A cinemática inversa nem sempre é um problema de solução analítica, ou muitas vezes não possui uma solução (HERNANDES, 2014). Contudo todas as configurações conhecidas possuem soluções, somente para novas configurações é que não existe um método específico e talvez seja necessária a invenção de um método novo (SANTOS, 2004).

A cinemática inversa é o método pelo qual se procura determinar o conjunto de valores das juntas que se relaciona com uma dada configuração do espaço cartesiano ou operacional (SANTOS, 2004). Em outras palavras, segundo (ERTHAL, 1992b), a cinemática inversa trata da obtenção das coordenadas das juntas, dadas as coordenadas externas do efetuador.

Um manipulador é dito solucionável quando todos os conjuntos de variáveis de juntas, associados a um dado posicionamento do efetuador, puderem ser obtidos numa forma explícita (RAO, 1989). Existem diversos métodos descritos na literatura para solucionar a cinemática inversa, mas somente será detalhado o método inverso de Jacobiano.

Na cinemática direta tem-se uma relação do tipo $\vec{r} = \vec{F}(\vec{q})$, ou seja, a posição do manipulador é a função da posição das juntas, no qual $\vec{r}$ representa a velocidade linear e angular do efetuador e $\vec{q}$ representa a velocidade angular das juntas (SANTOS, 2004; BARINKA, s.d.). Já na cinemática inversa tem-se a relação inversa do tipo $\vec{q} = \vec{F}_{-1}(\vec{r})$, isto é, a posição das juntas está em função da configuração espacial do manipulador (SANTOS, 2004). Desse modo, para relacionar as velocidades de cada junta do manipulador, é necessário ter o conceito da matriz jacobiana, que se relaciona com as posições das juntas e do espaço operacional, dado pela equação 2.3.

$$d\vec{r} = J \times d\vec{q} \tag{2.3}$$

Pela equação 2.3, segundo (HERNANDES, 2014), as velocidades espaciais estão relacionadas com a velocidade das juntas através da matriz jacobiana, que é definida para n juntas e m variáveis cartesianas, como mostra a equação 2.4.

$$J = \begin{bmatrix} \frac{\partial r_1}{\partial q_1} & \cdots & \frac{\partial r_1}{\partial q_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial r_m}{\partial q_1} & \cdots & \frac{\partial r_m}{\partial q_m} \end{bmatrix} \tag{2.4}$$

2.5 ROS

O ROS é um esforço colaboativo para criar um mecanismo robusto de uso geral na criação de aplicativos para robótica (ANDERSON, s.d.). Segundo (WIKIPéDIA, 2019), o

ROS é uma coleção de *frameworks* de *software* para desenvolvimento de robôs, que fornece a funcionalidade de um sistema operacional em um *cluster* de computadores heterogêneo. Ele fornece serviços padrões de um sistema operacional, tais como abstração de *hardware*, controle de dispositivos de baixo nível, a implementação de funcionalidades comumumente usadas, passagem de mensagens entre processos e gerenciamento de pacotes.

2.6 Gazebo

O Gazebo começou o desenvolvimento em 2002 na Universidade do Sul da Califórnia, criando um conceito de simulador de alta fidelidade, no qual sugiu da necessidade de simular robôs em ambientes externos sob várias condições (GAZEBO, 2014). Segundo (GAZEBO, 2014), a simulação de robô é uma ferramenta essencial para todo desenvolvedor de robótica. Um simulador bem projetado torna possível testar rapidamente algoritmos, projetar robôs, executar testes de regressão e treinar o sistema de inteligencia artificial usando cenários realistas. Ainda (GAZEBO, 2014) que diz oferecee uma capacidade de simular com precisão e eficiência populações de robôs em ambientes internos e externos complexos, onde existe um mecanismo de física robusto, gráficos de alta qualidade e interfaces gráficas e programáticas covenientes.

2.7 OpenCV

O *Open Source Computer Vision Library* (OpenCV) é uma biblioteca de *software* livre de visão computacional e aprendizado de máquina (OPENCV, 2020). Ela foi desenvolvida para fornecer uma infraestrutura comum para aplicativos de visão computacional e acelerar o uso da percepção da máquina nos produtos comerciais, que possui mais 2500 algoritmos otimizados (OPENCV, 2020). Segundo (OPENCV, 2020), esses algoritmos podem ser usados para detectar e reconhecer rostos, identificar objetos, classificar ações humanas em vídeos, rastrear movimentos de câmeras, rastrear objetos em movimento, extrair modelos 3D de objetos, entre outras aplicações. A biblioteca possui interfaces *C++*, *Python*, *Java* e *MATLAB* com suporte para os sistemas operacionais *Windows*, *Linux* e *MAC OS*.

2.8 MoveIt

Moveit é um pacote que é executado sobre o ROS, fornecendo funcionalidades para cinemática, planejamento de movimento e caminho, verificação de colisão, percepção 3D, interação do robô, entre outras aplicações (MOVEIT, s.d.b). O MoveIt é a principal fonte de muitas funcionalidades para manipuladores no ROS. Ele se baseia nos sistemas de mensagens e construção do ROS e utiliza algumas das ferramentas comuns do ROS, como o *ROS Visualizer* (RViz) e o formato do robô *Universal Robotic Description Format* (URDF). Segundo (MOVEIT, s.d.b), o MoveI está se tornando o ponto de entrada no ROS, especialmente através do uso do MoveIt Setup Assistant para configurar novos robôs. Como

nosso foco é sobre os manipuladores, para melhor compreensão sobre as funcionalidades e os conceitos do MoveIt pode ser consultado na referência (MOVEIT, s.d.a).

2.9 Ambiente de operação

O manipulador desenvolvido será acoplado a uma bancada que possui uma caixa com uma botoeira de emergência e uma *tag* do tipo ARUCO a ela acoplada. O centro da base quadrada do manipulador será fixado em uma extremidade da bancada e a caixa será colocada em um local qualquer dentro do volume de trabalho do manipulador. Não haverá outro tipo de obstáculo, como ilustrado na figura 7.

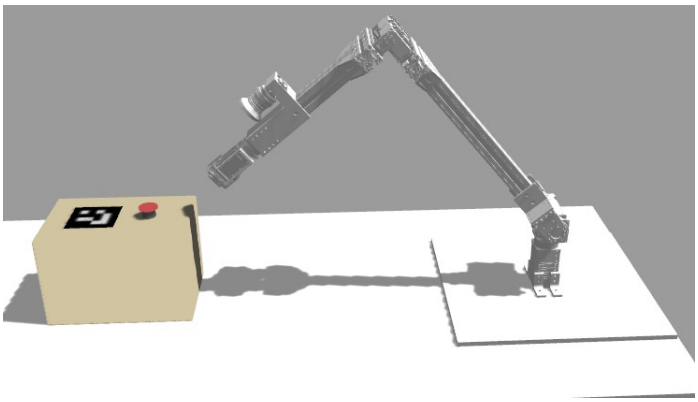


Figura 7: Ambiente de operação simulado.

Fonte: Os autores.

3 DESENVOLVIMENTO DO SISTEMA

Uma boa conduta no desenvolvimento de um sistema robótico é levar os seus componentes para representações em simulação. Para isso necessitou-se, em primeiro lugar, utilizar uma ferramenta de *Computer Aided Design* (CAD). A ferramenta online gratuita *Onshape* foi escolhida e através realizou-se a montagem dos objetos que virão a compor a cena esperada do sistema. A partir disto pôde-se inserir os objetos, por meio de uma descrição em arquivos do tipo URDF. Estes arquivos são códigos em linguagem XML onde se fazem declarações sobre como os elementos que compõem um robô, do ponto de vista geométrico principalmente, se conectam. O simulador *Gazebo* utiliza estes arquivos como parte dos dados necessários para calcular a dinâmica do sistema para posteriormente exibir estes resultados através de uma interface gráfica.

3.1 Descrição do sistema

O conjunto formado pelo manipulador, seu espaço de trabalho e os objetos com os quais ele deverá interagir compõem o sistema abordado neste trabalho. O espaço de trabalho foi representado como uma bancada de 1,70 m × 0,80 m × 0,028 m. Há apenas um objeto com o qual o manipulador deverá interagir, uma caixa de dimensões 0,30 m ×0,20 m × 0,20 m.

Nesta caixa há dois objetos: um marcador fiducial do tipo ARUCO que é a codificação gráfica de um número, tal como os *Quick Response* (QR) code já bastante populares. Essa etiqueta é responsável por sinalizar ao manipulador que uma determinada localização foi encontrada, e, a partir disto, uma tomada de decisão deve ser feita por ele. A decisão, para este trabalho, é a de pressionar a botoeira de emergência contida na caixa e referênciada pela tag.

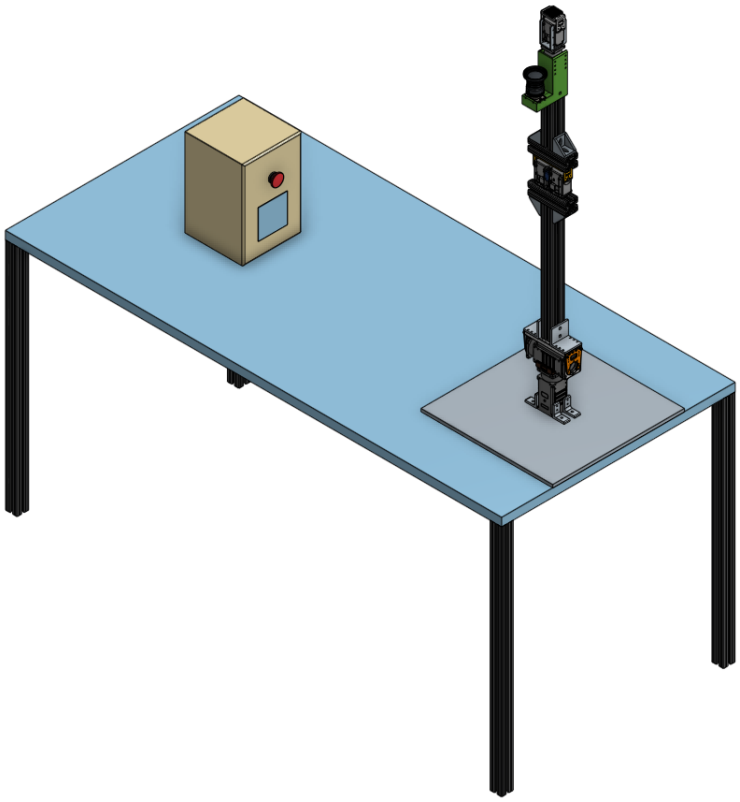
Por fim, há o próprio manipulador, que é articulado em cinco juntas onde se encontram servomotores que realizam a ação de girar, de modo controlado, os elos a eles presos, conferindo cinco graus de liberdade à ferramenta do robô: uma chapa de metal destinada a fazer contato físico com a botoeira e acioná-la.

3.1.1 Arquitetura geral

A arquitetura geral do sistema consiste na integração entre os processos do ROS sendo operado pelo sistema principal, como mostra a figura 9. Ocorrem então o movimento do manipulador, sendo controlado pelo *MoveIt*, e a simulação, que é a integração dos componentes virtuais dos sistema. Nesses componentes estão presentes o manipulador, que possui atuadores e sensores de estados, e uma câmera que simula a parte visual.

A interface humano-máquina pode ser controlada utilizando os softwares *RViz* e *Gazebo*,

Figura 8: Representação visual do sistema.



Fonte: Os autores

permitindo assim ter-se uma visualização geral do cenário que está sendo simulado. Para a detecção visual, é utilizado um pacote externo para interligar os componentes interessados no cenário chamado "*bir_marker_localization*". O *MoveIt Setup Assistent* é responsável por realizar a configuração do manipulador, para que o movimento seja realizado. *MoveIt commander* é interligado com o ROS e ele é utilizado com a finalidade de controle do movimento do manipulador.

Representa-se na figura 9, de maneira resumida, os componentes do sistema:

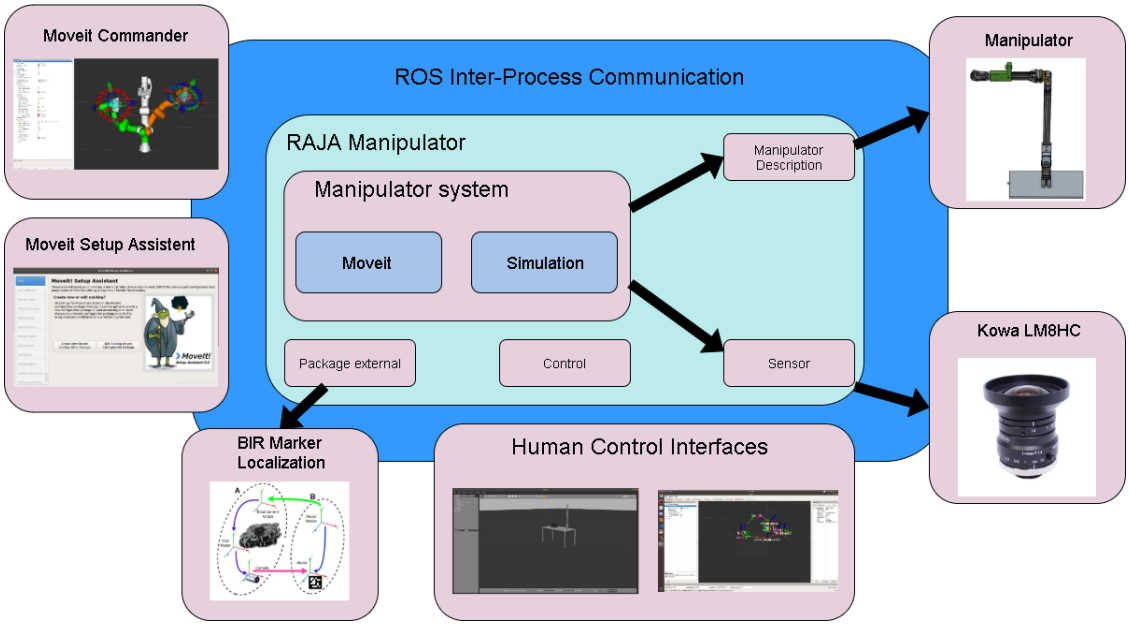
3.1.2 Especificação técnica

A presente especificação técnica constitui elemento fundamental para o cumprimento dos requisitos do cliente e o cumprimento do objetivo final do manipulador. Esse tópico será subdividido em alguns pontos para melhor apresentar tais especificações.

3.1.2.1 Estrutura do manipulador

Para estruturação dos elos do robô, são utilizados perfis de alumínio estrutural 40×40-10 mm, com rosca interna M12, na figura 10. Segundo fabricante deste tipo de perfil (OBR. . . , s.d.), os mesmos apresentam tipicamente uma massa de 0,8 kg/m. Foram utilizados dois

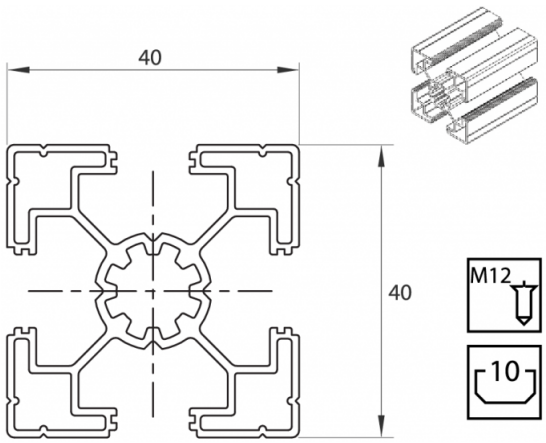
Figura 9: Visão geral dos elementos do sistema.



Fonte: Os autores

elos deste material, o primeiro possuindo 40 cm de comprimento e o segundo, 20 cm. Estas dimensões foram projetadas levando em consideração a capacidade de alcance do braço para realizar a tarefa, bem como a capacidade estrutural do manipulador, para assim suportar esforços de natureza estática e dinâmica. Para a base, foi projetado uma chapa de liga de alumínio 5000 com dimensões 50×50 cm^2 e espessura de 10 mm. Essa base será utilizada para fixar o manipulador, gerando estabilidade durante a execução da tarefa.

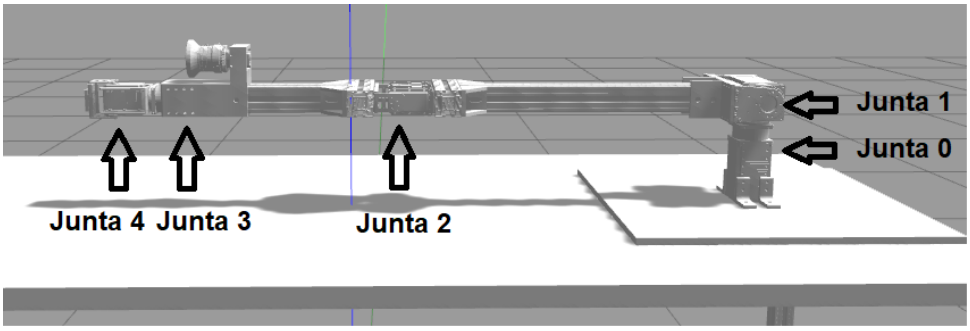
Figura 10: Estrutura dos elos.



Fonte: (OBR. . . , s.d.)

O manipulador foi dimensionado com o comprimento de 1 m. Levou-se em consideração o mínimo alcance necessário para que a missão possa ser realizada conforme as especificações do cliente. Então foram projetadas cinco juntas rotacionais como mostra a figura 11.

Figura 11: Juntas do manipulador.



Fonte: Os autores.

Tabela 1: Análise de esforços das juntas.

Junta	Momento Torsor	Torque de atuação dos motores	Força máxima suportada pela junta(ROBOTIS, s.d.)
0	17 N/m	44.7 N/m	70 N
1	17 N/m	44.7 N/m	70 N
2	4.4 N/m	2 x 3 N/m	19 N
3	0.4 N/m	5.1 N/m	50 N

Fonte: Os autores.

A análise de esforços mecânicos aos quais o manipulador está exposto foi realizada para embasamento e confirmação da capacidade da estrutura e dos atuadores suportarem os esforços de momento torsor. Para isso foram analisadas individualmente as juntas em seus estados críticos. Na tabela 1 é apresentada a análise de esforços nas juntas e comparados com os máximos esforços suportados pelos atuadores, levando em consideração especificações do fabricante, para evitar falhas mecânicas.

Levando em consideração os valores obtidos nas análises de esforços, tem-se a junta 2 como junta crítica do sistema, ou seja, esta junta chegará a falha mecânica antes das demais juntas devido aos esforços que é submetida e a capacidade dos seus dois motores. Na configuração apresentada a junta 2 suporta a carga máxima de 19 N sem chegar a falha mecânica, logo o manipulador possui um limite (*payload*) de aproximadamente 1,95 kg na sua extremidade.

3.1.2.2 Atuadores

Os atuadores do manipulador são motores de corrente contínua, integrados com redutor de velocidade, controlador e driver. Foram utilizados atuadores *Dynamixel* da fabricante ROBOTIS. Entre os modelos figuram o MX-106R (figura 12a) para as articulações 2 e 4, porém na junta 2 eles foram acoplados em paralelo para trabalhar em sincronismo, aumentando assim a capacidade de torque na articulação. Para as juntas 0 e 1, foi

utilizado o motor PH54-200-S500-R (figura 12b). O motor PH42-020-S300-R (figura 12c) foi utilizado para a articulação 3. As especificações do fabricante mais relevantes no projeto estão apresentadas na tabela 2 para os motores utilizados. As folhas de dados estão disponíveis no *website* da ROBOTIS para os atuadores MX-106-R (ROBOTIS. . . , s.d.a), PH42-020-S300-R (ROBOTIS. . . , s.d.b) e PH54-200-S300-R (ROBOTIS. . . , s.d.c).

Tabela 2: Especificações do fabricante para os atuadores.

Especificações dos motores em operação contínua					
Tipo	Torque (N/m)	Voltagem (V)	Corrente (A)	Velocidade de rotação (rpm)	Dimensões (mm)
PH54-200-S500-R	44.7	24.0	9.3	29.0	54.0 X 126.0 X 54.0
PH42-020-S300-R	5.1	24.0	1.5	29.2	42.0 X 84.0 X 42.0
MX-106R	2.9	12.0	5.2	(Não especificado)	40.2 X 65.1 X 46.0

Fonte: (ROBOTIS, s.d.)

Figura 12: Atuadores Dynamixel.

(a) MX-106R.

(b) PH54-200-S500

(c) PH42-020-S300-R

Fonte: (ROBOTIS, s.d.)

3.1.2.3 Câmera

Para prover o sistema com capacidade de detecção da *tag* e assim obter dados necessários para aquisição de pose e orientação relativa, utilizou-se uma câmera de video modelo Teledyne Genie Nano C2590 (figura 13). De pequenas dimensões e massa reduzida, essa câmera pode ser acoplada próxima a extremidade do manipulador robótico, facilitando a detecção da *tag* e não comprometendo a estrutura do manipulador.

3.1.2.4 Volume de trabalho

A área de trabalho ou *workspace* de um manipulador é o volume total alcançado pelo efetuador do manipulador, executando todos os possíveis movimentos, ou seja, é a região dentro da qual o manipulador pode posicionar o efetuador. Essa área é limitado pela geometria do manipulador, bem como as restrições mecânicas sobre as articulações. Para este projeto foi desenvolvido um script em *Octave* que calcula o *workspace* do manipulador, a partir de um modelo de um braço robótico com 3 juntas possuindo o mesmo comprimento

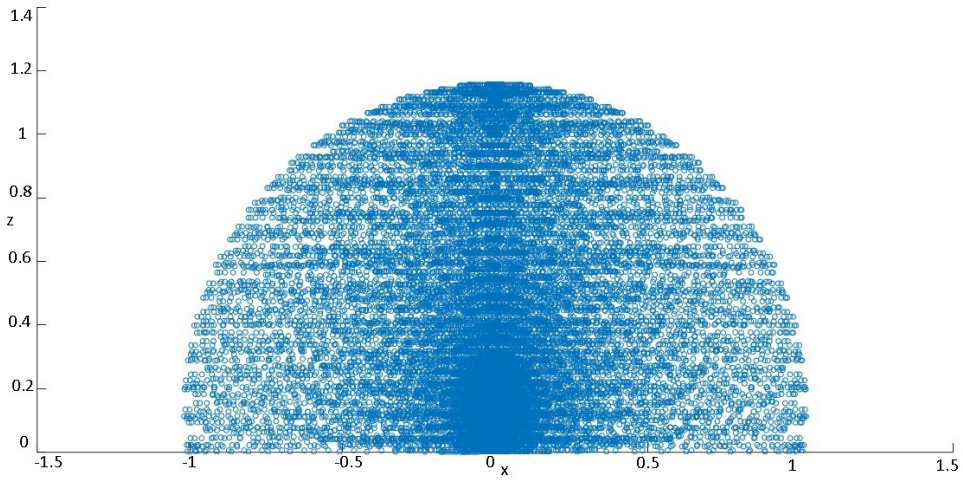
Figura 13: Teledyne Genie Nano C2590.

Fonte: (TELEDYNE. . . , s.d.)

e obedecendo as distâcias entre cada junta até o efetuador. O código desse script pode ser visto no apêndice B.

A partir do script do apêndice B pode-se gerar os gráficos que representa o volume de trabalho alcançado pelo manipulador, como pode ser visto nas figuras 14, 15, 16. A figura 14 mostra o alcance máximo vertical de 1,2 m e horizontal de 1 m. A figura 15 mostra o alcance horizontalpela perspectiva lateral, porém não fica muito evidente o alcance mínimo pois há interposição dos valores. Na figura 16 apresenta uma visão de cima da amplitude do manipulador.

Figura 14: Área de trabalho projetada no plano XZ.

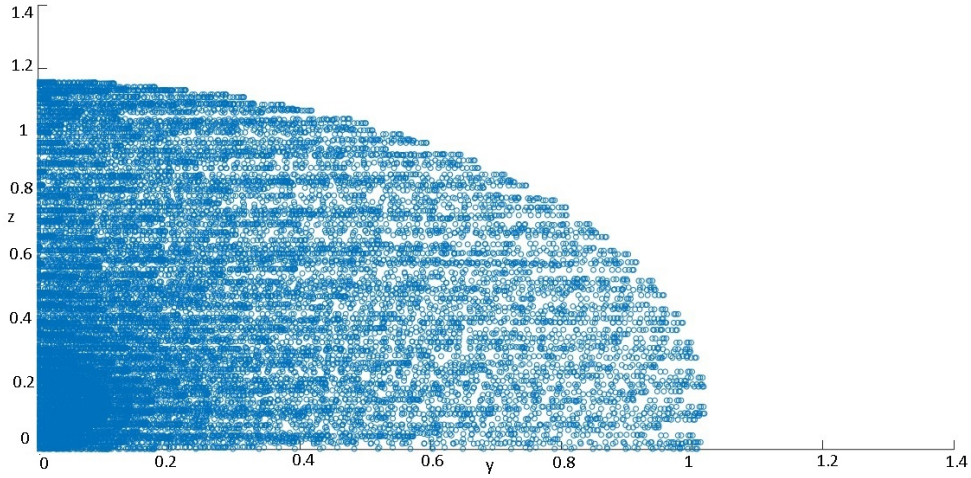


Fonte: Os autores.

3.1.2.5 Unidade de potência

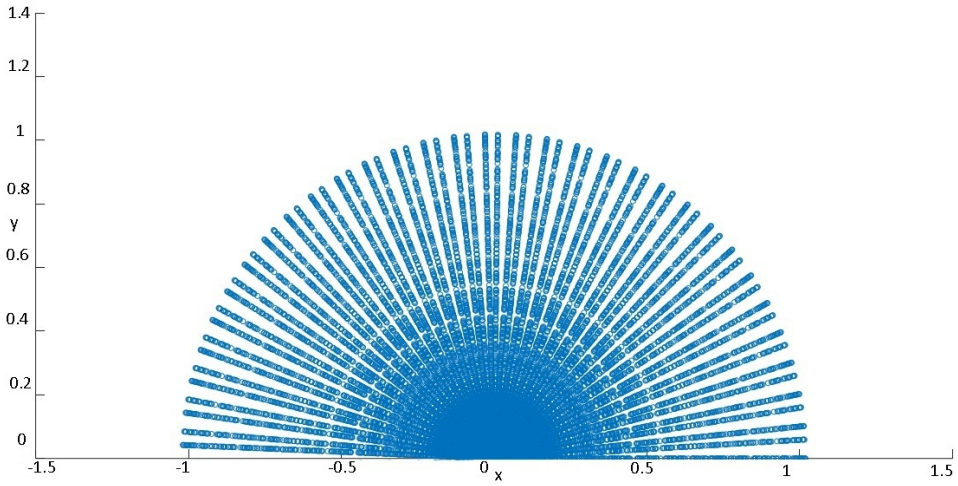
O sistema necessitará ser energizado em dois níveis de tensão, 12 V e 24 V, os mesmos níveis dos atuadores utilizados. Para isso, uma fonte de tensão será necessária juntamente com um conversor DC-DC para ou elevar ou reduzir a tensão de alimentação, caso ela seja 12 V ou 24 V, respectivamente. Pela forma como os motores serão empregados, a potência de entrega da fonte pode ficar muito abaixo da potência máxima do conjunto

Figura 15: Área de trabalho projetada no plano YZ.



Fonte: Os autores.

Figura 16: Área de trabalho projetada no plano XY.



Fonte: Os autores.

de atuadores. Assim uma fonte que forneça 400 W é necessária. Um esquema elétrico é fornecido no apêndice A, indicando as conexões dos cabos entre os motores.

3.1.3 Estrutura analítica do protótipo

A estrutura analítica do protótipo consiste em uma subdivisão hierárquica dos seus subsistemas e componentes. Essa estrutura tem como objetivo primário a organização dos níveis que compõem o projeto, proporcionando uma visão ampla do projeto e um maior controle. Foi concebida uma análise estrutural do manipulador robótico, levando em consideração quatro níveis de hierarquia como mostrado na figura 17.

Figura 17: Estrutura analítica do protótipo.

Fonte: Os autores.

3.2 Especificação funcional

O manipulador simulado deve ser capaz de realizar uma inspeção visual e busca da *Augmented Reality* (AR) *tag*. No momento em que esta for encontrada, a busca deve ser interrompida e o manipulador inicia uma rotina de aproximação do botão. Esta rotina é realizada em duas etapas, que se separam pela precisão com a qual o manipulador deve se deslocar no espaço. Até que a aproximação atinja um dado limiar, a sua movimentação pode ser realizada de maneira mais abrupta; atingido o limiar, uma aproximação mais suave conduz o efetuador até o botão e o seu pressionamento é realizado, concluindo o objetivo do sistema.

Na figura 18 é ilustrado o fluxograma da missão.

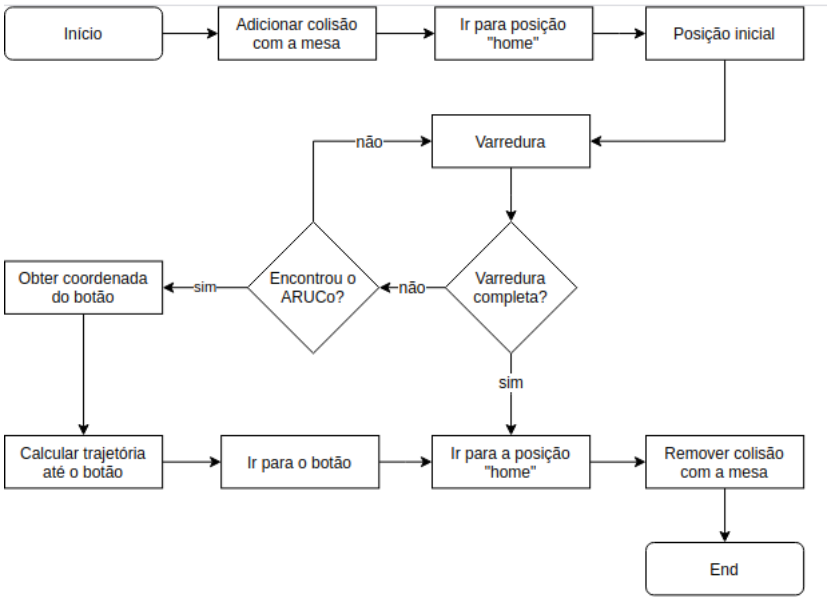
3.2.1 Funcionalidade de busca

Para poder localizar-se no espaço de trabalho, o manipulador percorre uma trajetória pré-definida a partir da pose inicial padrão, denominada *home pose*, a fim de detectar, por meio da câmera instalada, a *tag* pré-especificada em suas configurações.

3.2.1.1 Descrição

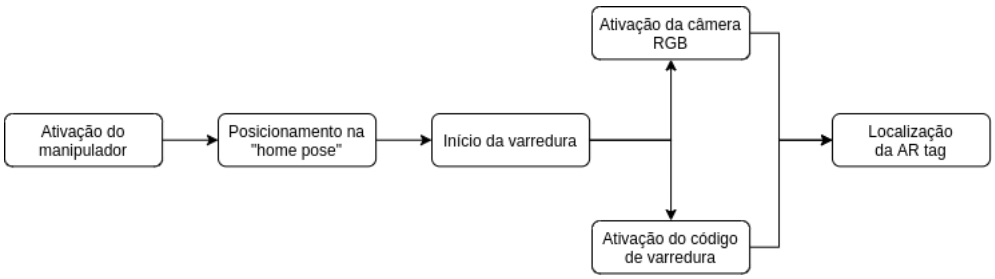
O manipulador inicia a busca a partir da sua posição *home* e realiza dois ciclos. A junta 0 realiza meia volta em cada ciclo de busca. Também em cada ciclo a junta 1, que inicia a 45° com o plano da horizontal, passa por incrementos de 22,5°. Caso o marcador fiducial não seja encontrado, o manipulador retorna à posição *home*. A figura 19 ilustra a sequência do processo de busca.

Figura 18: Fluxograma do sistema.



Fonte: Os autores.

Figura 19: Detalhamento da funcionalidade de busca.



Fonte: Os autores.

3.2.1.2 Premissas necessárias

Assume-se que há uma *tag* de tamanho grande o suficiente para ser detectada pela câmera empregada; também assume-se que o movimento executado na busca está adequado ao campo de visão da câmera.

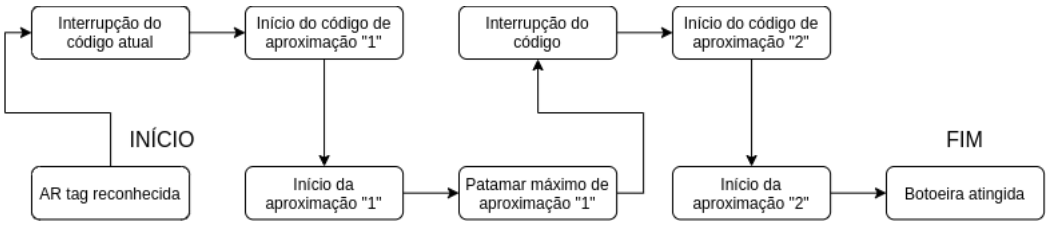
3.2.2 Funcionalidade de aproximação

Após a detecção da *tag* e estimativa da pose relativa do efetuador, inicia-se um procedimento de aproximação baseado na posição que o manipulador reconhece que ele ocupa no mundo. Conforme mencionado, a aproximação acontece em dois perfis diferenciados pela velocidade até que se pressione a botoeira de emergência.

3.2.2.1 Descrição

A figura 20 ilustra as etapas da aproximação do manipulador ao botão.

Figura 20: Detalhamento da funcionalidade de aproximação.



Fonte: Os autores.

3.2.2.2 Premissas necessárias

A funcionalidade aproximação depende do sucesso da aquisição de coordenadas na funcionalidade de busca. É iniciada autonomamente após o sucesso da funcionalidade anterior.

3.3 Arquitetura de software

O sistema é dividido em *software* e *hardware*. Os principais componentes que integram o *hardware* são o manipulador e a câmera Teledyne. Já os componentes de *software* são relacionados ao framework ROS, juntamente com o pacote de movimento do sistema, incluindo a biblioteca do *MoveIt Commander*.

3.4 Simulação do sistema

Para realizar a simulação no *Gazebo* utilizaram-se ferramentas do framework ROS, além dos modelos em CAD e os códigos concebidos. O framework (ROS. . . , s.d.) consiste em uma série de programas pré-concebidos e ajustáveis usados para representar em simulação dispositivos usuais encontrados no mundo da robótica, e são denominados *plugins*. Os *plugins*, após devidamente personalizados são organizados em pacotes para serem utilizados, possivelmente em conjunto com programas à parte. Neste trabalho, utilizaram-se *plugins* para simular o deslocamento angular dos elos em cada junta e a câmera. Não sendo estes *plugins* suficientes para contemplar todas as tarefas que a ele cabiam, conceberam-se os programas responsáveis por realizar a busca em busca do marcador fiducial e a lógica de movimentação do efetuador para o acionamento da botoeira.

3.4.1 *CAD*

Os modelos em CAD foram adquiridos nos sites dos fabricantes dos produtos (os atuadores *Dynamixel* e os perfis de alumínio) ou foram desenhados (as cantoneiras que conferiram maior rigidez às juntas) e montados na plataforma *Onshape*. Lá, além das dimensões e formas, a massa e a densidade dos materiais utilizados foram fornecidos e com isso foram obtidos os momentos de inércia para serem utilizados nos arquivos URDF possibilitando a simulação adequada da dinâmica do sistema.

3.4.2 *ROS*

O ROS contém duas ferramentas para a visualização da simulação. A primeira, o *Gazebo* é um software feito em duas partes, onde uma é dita *server* pois é o responsável por processar todas as informações relacionadas a dinâmica dos objetos, a luminosidade no ambiente, e todos os outros agentes físicos presentes na cena representada. A outra parte é o *graphical client*, que dispõe de uma interface gráfica Graphic User Interface *(GUI)* responsável por gerar para o usuário o visual da cena (GAZEBO. . . , s.d.). A outra ferramenta é *RViz*, destinada a fornecer informações numéricas de maneira gráfica para uma análise eficiente dos elementos dinâmicos que são geridos pelo *Gazebo server*. Dessa forma, os plugins do *Gazebo server* fornecem dados tanto para o *Gazebo client* quanto para o *RViz*. Um exemplo de exibição que foi bastante útil no projeto foram os eixos coordenados dos frames de cada uma das juntas. Ou a imagem capturada pelo plugin de câmera, permitindo ao usuário ver o que era gerado pela câmera artificial, que por sua vez permitiu avaliar o quão bem estava funcionando a detecção da AR *tag*.

3.4.3 *MoveIt*

A descrição dos elementos físicos de todo o sistema foi feita a partir de arquivos *Extensible Markup Language* (XML), ou mais especificamente XML *macros* (XACRO). Estes arquivos são uma versão generalizada dos URDF, permitindo o uso de expressões matemáticas simples e as já mencionadas macros. Foram também empregados na atuação do software de planejamento dinâmico da movimentação do manipulador, *MoveIt*. Este software calculou as interações entre os elos, avaliando possíveis colisões. Ele também adicionou aos arquivos XACRO parâmetros que permitiram que as juntas fossem controladas por um plugin do *Gazebo*. Foram criados então dois pacotes - um para a descrição do manipulador e outro contendo as informações agregadas ao primeiro pelo *MoveIt*.

3.4.4 Pacote *bir_marker_localization*

Após a concepção de um modelo controlável em ambiente simulado, pôde-se programar a sua busca pela AR *tag* e implementar o pacote responsável pela sua detecção. Neste

pacote, um arquivo de configuração detém as informações sobre a *tag* utilizada, pois a mesma precisa ser previamente esperada para que a detecção ocorra. Este arquivo também contém parâmetros de câmera e as transformadas (TF. . . , s.d.) pertinentes aos objetos que se deseja conhecer a posição. A premissa do pacote é que avistando uma *tag* é possível estimar a posição de uma parte do manipulador em relação a ela. A versatilidade contida em "parte qualquer" é devida ao fato de que todos os elos do manipulador sendo rídigos, de comprimento fixo, é sempre possível saber a posição relativa entre todos eles. Dessa forma a pose do efetuador é determinada e o programa de posicionamento entra em ação para premir a botoeira de emergência.

3.5 Testes realizados e resultados

Estatística é a ciência que nos ajuda a fazer decisões e tirar conclusões na presença de variablidade (MONTGOMERY, 2012). Segundo (LENTH, 2001), determinar o tamanho das amostras é um passo importante para planejar o estudo estatístico. A quantidade de amostras deve ser o suficiente para que cada uma tenha significância estatística.

Considerando a premissa que o manipulador possa interagir com uma *AR tag* independente de sua orientação no espaço e que a mesma possa ser vista pela câmera no ciclo de escaneamento, estabelecemos duas posições para os testes de localização: a caixa na horizontal e na vertical, ambas em relação à mesa. Vale salientar que a posição que o manipulador deseja alcançar está a frente do botão, para que não ocorram colisões desnecessárias.

Utilizando o tempo de simulação do *Gazebo* como recurso, foram selecionados 30 amostras para cada posição. Cada amostra pode ser dividida em duas partes: tempo em segundos da posição *Home* até o sistema identificar a *AR tag* e o tempo em segundos da identificação até o posicionamento final. Foi gerado um histograma de cada parte com uma linha que representa sua densidade. Depois, foi calculado cada média e aplicado o teste t de Student para identificar se os valores apresentam um nível de confiabilidade de 95% e uma distribuição normal a partir da mediana da população ao resultar em um *p-value* maior 0.05, segundo (MONTGOMERY, 2012).

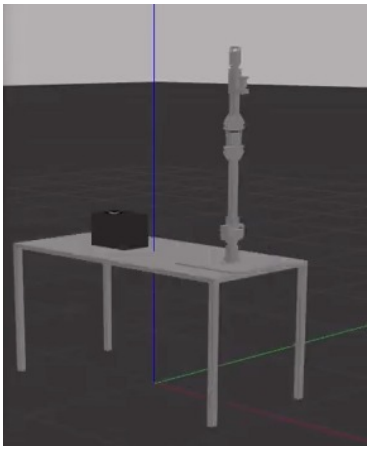
A média de tempo de deslocamento da posição *Home* até a identificação da *AR tag* é de 12.48 segundos e a sua mediana é de 12.42 segundos, a partir dos dados da tabela 3. Seu desvio padrão é de 0.2526 e seu teste t resultou em um *p-value* de 0.1888, apresentando 95% de confiabilidade entre os valores de 12.38774 segundos e 12.57639 segundos. Estes valores são aceitáveis pois o ciclo de escaneamento é programado para uma mesma movimentação, e a caixa estava na mesma posição durante as 30 amostras. Houveram dois valores atípicos nas amostras de número 5 e 21 que contribuem para o histograma, visto na figura 23, ser deslocado para a esquerda.

Tabela 3: Testes do manipulador em curso ao objetivo - caixa horizontal.

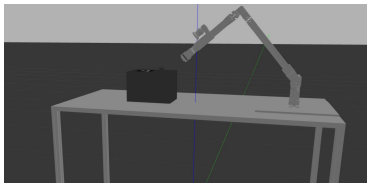
Amostra	*Home-Scan* (s)	*Scan-Goal* (s)
1	12.539	2.478
2	12.419	2.972
3	12.34	2.769
4	12.516	5.345
5	13.224	4.543
6	12.425	4.364
7	12.416	2.6
8	12.413	4.734
9	12.645	5.539
10	12.345	5.331
11	12.319	3.094
12	12.317	2.762
13	12.426	4.936
14	12.226	5.79
15	12.318	5.979
16	12.334	2.605
17	12.421	2.547
18	12.549	2.516
19	12.354	5.134
20	12.252	4.926
21	13.406	5.368
22	12.311	1.972
23	12.61	4.557
24	12.356	5.567
25	12.616	2.259
26	12.414	2.336
27	12.427	5.608
28	12.439	4.526
29	12.629	4.931
30	12.456	2.531

Fonte: Os autores.

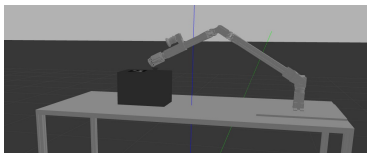
Figura 21: Manipulador em curso ao objetivo - caixa horizontal.



(a) Manipulador em *home pose*.



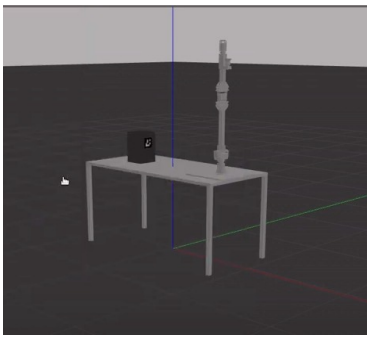
(b) Manipulador em busca da AR *tag*



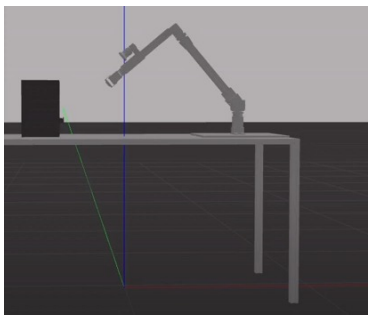
(c) Objetivo alcançado.

Fonte: Os autores.

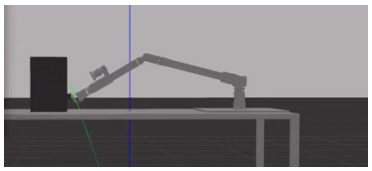
Figura 22: Manipulador em curso ao objetivo - caixa vertical.



(a) Manipulador em *home pose*.



(b) Manipulador em busca da AR *tag*



(c) Objetivo alcançado.

Fonte: Os autores.

A média de tempo de deslocamento da posição de identificação da *AR tag* até o botão é de 4.021 segundos e a sua mediana é de 4.535 segundos. Seu teste t resultou em um *p-value* de 0.04597. A figura 24 mostra o histograma que não apresenta uma distribuição próxima a normal devido a cinemática do manipulador apresentar diversas soluções para resolver o problema da trajetória. As amostras apresentam um valor mínimo de 1.972 segundos e um valor máximo de 5.979 segundos. Seu desvio padrão é de 1.351131, um valor muito maior se comparado com a atividade anterior.

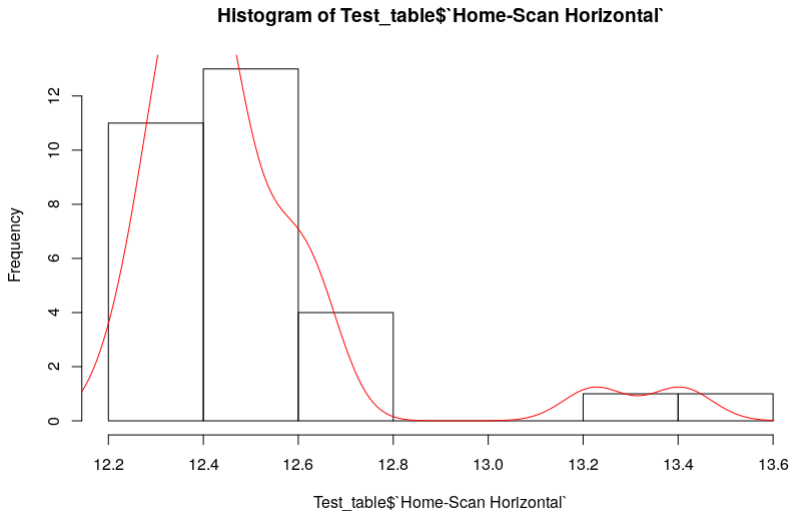
A média de tempo de deslocamento da posição *Home* até a identificação da *AR tag* é de 12.54 segundos e a sua mediana é de 12.42 segundos, a partir dos dados da tabela 4. Seu desvio padrão é de 0.3984279 e seu teste t resultou em um *p-value* de 0.101, apresentando 95% de confiabilidade entre os valores de 12.39442 segundos e 12.69198 segundos. A

Tabela 4: Testes do manipulador em curso ao objetivo - caixa vertical.

Amostra	*Home-Scan* (s)	*Scan-Goal* (s)
1	12.439	2.715
2	12.239	2.329
3	13.609	2.525
4	12.321	2.33
5	12.226	2.704
6	12.331	2.726
7	12.452	2.719
8	12.411	2.31
9	12.317	2.72
10	12.406	2.725
11	12.411	2.723
12	12.416	2.705
13	13.404	2.724
14	13.109	2.323
15	12.416	2.72
16	12.548	2.714
17	12.638	2.719
18	13.812	2.719
19	12.319	2.721
20	12.445	2.716
21	12.335	2.323
22	12.345	2.721
23	12.219	2.722
24	12.319	2.724
25	12.444	2.713
26	12.42	2.311
27	12.458	2.319
28	12.533	2.71
29	12.525	2.707
30	12.429	2.319

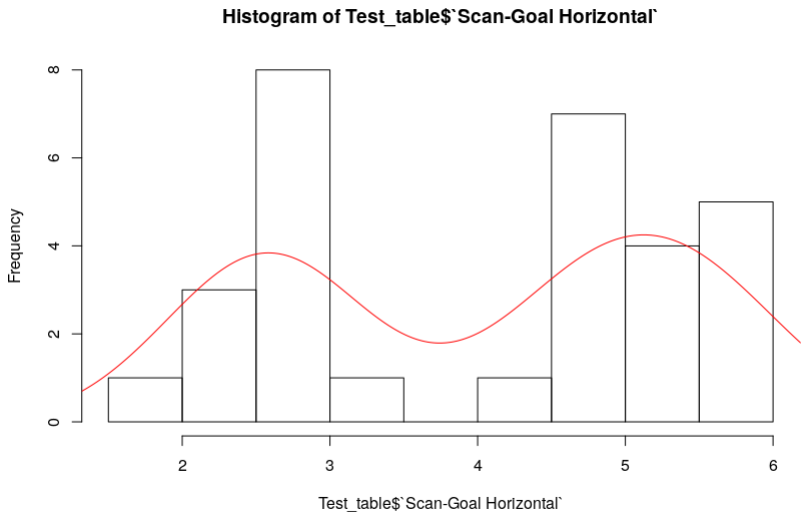
Fonte: Os autores.

Figura 23: Histograma e densidade das amostras da movimentação da posição *Home* até identificação da *tag* - caixa horizontal.



Fonte: Os autores.

Figura 24: Histograma e densidade das amostras da movimentação do fim do ciclo de escaneamento até o botão - caixa horizontal.

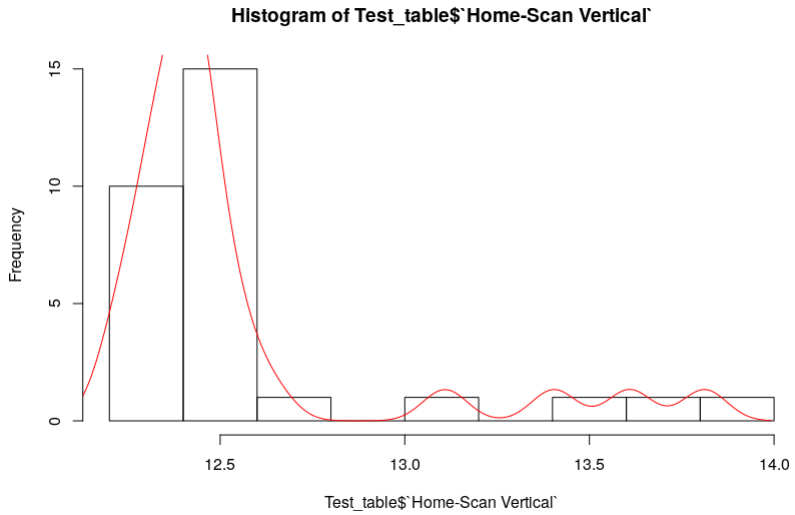


Fonte: Os autores.

justificativa dos valores se mantém semelhante com a mesma atividade representada na caixa horizontal. Porém, pode-se perceber que a quantidade de valores atípicos a mais influenciaram o valor do desvio padrão da amostra, apesar desta atividade ser mais precisa. Houveram quatro valores atípicos nas amostras de número 3, 12, 13 e 18 que contribuem para o histograma, como pode ser visto na figura 25, ser deslocado para a esquerda.

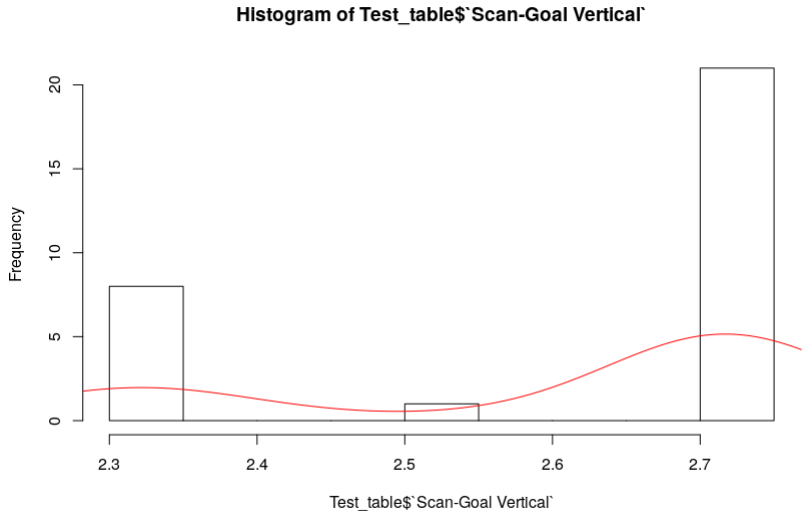
A média de tempo de deslocamento da posição de identificação da *AR tag* até o botão

Figura 25: Histograma e densidade das amostras da movimentação da posição *Home* até identificação da *tag* - caixa vertical.



Fonte: Os autores.

Figura 26: Histograma e densidade das amostras da movimentação do fim do ciclo de escaneamento até o botão - caixa vertical.



Fonte: Os autores.

é de 2.605 segundos e a sua mediana é de 2.715 segundos. Seu teste t resultou em um *p-value* de 0.002114. O histograma 26 não apresenta uma distribuição próxima a normal devido a cinemática do manipulador apresentar diversas soluções para resolver o problema da trajetória, mesma justificativa para a caixa na horizontal. As amostras apresentam um valor mínimo de 2.31 segundos e um valor máximo de 2.726 segundos. Vale ressaltar que a frequência de valores é bem maior perto do seu valor máximo, resultando em um desvio padrão considerado baixo para a atividade (0.1781923).

4 CONFIABILIDADE DO SISTEMA

Após conhecimento dos subsistemas presentes no desenvolvimento do manipulador robótico bem como dos seus componentes apresentados na estrutura analítica do protótipo, foi elaborado um plano de aplicação da *Failure Mode and Effect Criticality Analysis* (FMECA) para identificar os modos de falhas que podem ocorrer no produto ou protótipo. Seguindo o formulário FMECA apresentado nas tabela 8 estão indicados: a) Funções do componente analisado; b) Modos de falhas que possam vir a ocorrer; c) Os possíveis efeitos causados no sistema pelas falhas; d) Uma análise de causas possíveis que gerariam os erros analisados; e) São apresentadas algumas ações recomendadas para que a falha seja corrigida ou minimizada; f) A classificação da competência em cada caso.

4.1 Análise dos modos e efeitos de falhas

Para uma realizar uma análise de riscos foram parametrizadas inicialmente duas avaliações dos modos e efeitos de falha. A primeira delas foi a severidade da falha, que indica o quanto ela pode comprometer o funcionamento global do sistema. Na segunda avaliação, é feita uma análise quanto a detectabilidade da falha, que é o quanto essa falha pode ser facilmente detectada, para assim aplicar o plano de ação recomendado para minimizá-la. Esses critérios são apresentados respectivamente nas tabelas 5 e 6 a seguir.

Tabela 5: Classificação de severidade de falhas.

Classificação	Severidade	Critério de classificação
1	Mínima	O cliente mal percebe que a falha ocorre
2	Pequena	Ligeira deterioração no desempenho com leve descontentamento do cliente
3	Moderada	Deterioração significativa do desempenho de um sistema com descontentamento do cliente
4	Alta	Sistema deixa de funcionar e grande é o descontentamento do cliente
5	Muito Alta	Agravamento do anterior com risco a segurança

Fonte: Os autores.

A tabela 8 condensa os estudos de FMECA realizados para os subsistemas que compõem o projeto, apresentando uma análise de severidade (S), detecção (D) e ocorrência da falha (O). O índice de criticidade é dado mediante cálculo dos demais índices apresentados, para assim realizar uma priorização dos modos de falha.

Tabela 6: Classificação de facilidade de detecção de falhas.

Classificação	Capacidade de Detecção	Critério de classificação
1	Muito grande	Certamente será detectado
2	Grande	Grande possibilidade de ser detectado
3	Moderada	Provavelmente será detectado
4	Pequena	Provavelmente não será detectado
5	Muito pequena	Certamente não será detectado

Fonte: Os autores.

Tabela 7: Classificação da probalilidade de ocorrência de falhas.

Valor	Ocorrência	Critério
1	Remota	1:1.000.000
2	Pequena	1:10.000
3	Moderada	1:1.000
4	Alta	1:10
5	Muito alta	>1:8

Fonte: Os autores.

Como observado na tabela 8 é apresentado o índice de criticidade mediante cálculo dos índices de severidade, detecção e ocorrência. Esse índice permite enfatizar os subsistemas de atuação e de potência como críticos para o manipulador robótico desenvolvido, pois esses apresentam elevado índice de criticidade em um ou mais modos de falha.

Tabela 8: Matriz de modos e efeitos de falhas para os subsistemas do manipulador

Função(ões)	Modo(s) de falha em potencial	Efeito(s) potencial(is) da falha	S	Causa(s) potencial(is) / mecanismo(s) de falha(s)	D	O	Ação(ões) recomendada(s)	Competência	S x D x O
Atuação das juntas do manipulador (motores).	Não funciona mento dos atuadores	Perda do controle de uma ou mais juntas do manipulador	4	Problemas elétricos	1	3	Inspeções periódicas	Elétrica	12
		Perda de movimentação do manipulador	4	Falha na comunicação com o controlador	3	3	Inspeções periódicas	Elétrica	36
	Movimentação não compatível com a programação	Perda de desempenho do manipulador	3	Falhas no *software*	3	2	Inspeções periódicas	Elétrica	18
		Trajetória incompatível	3	Vibrações mecânicas	4	1	Isolamento da área de trabalho	Mecânica	12
		Colisão com demais objetos	5	Falhas na comunicação com o controlador	3	3	Sistema de detecção de obstáculos e isolamento da área de trabalho	Elétrica	45
Detecção da *tag* (câmera).	Não detecção da *tag*	Perda da aquisição de posicionamento e orientação	3	Problemas elétricos	2	3	Inspeções de conexões e cabos	Elétrica	18
		Não realização da tarefa	4	Falha na comunicação com o controlador	3	3	Inspeções de conexões e cabos	Elétrica	36
	Coleta de dados inconsistentes	Comprometimento dos dados obtidos	3	Problemas de visibilidade da *tag*	2	3	Certificar luminosidade e ausência de obstáculos	Elétrica	18
		Erro no posicionamento e orientação	3	Calibração incorreta	3	3	Executar varias verificações	Elétrica	27
			3	Danificação da lente da câmera	2	1	Inspeção e instalação correta	Mecânica	6
Processamento dos subsistemas do manipulador (NUC).	Falha no processamento	Não funcionamento do manipulador	4	Falha no sistema operacional	2	3	Executar testes de verificação	Elétrica	24
			4	Falha na comunicação com o controlador	3	3	Inspeções de conexões e cabos	Elétrica	36
		Perda de dados	3	Falha da memória interna	3	3	Rotina de teste	Elétrica	27
Converter níveis de tensão DC (Conversor DC/DC)	Não converter níveis de tensão	Não alimentação do sistema	4	Falha nos componentes do conversor	4	2	Executar testes de verificação	Elétrica	32
	Conversão de níveis de tensão incompatíveis com os atuadores	Não funcionamento dos sistemas dependentes do conversor	4	Falha nos componentes do conversor	4	2	Executar testes de verificação	Elétrica	32
		Danificação dos sistemas	4	Uso de tensão de entrada não especificada	2	4	Executar testes de verificação	Elétrica	32
	Superaqueci mento	Danificação do conversor	5	Mau contato nos terminais	3	3	Executar testes de verificação	Elétrica	45
Estabilizar níveis de tensão do sistema (Conversor DC/DC)	Não estabilizar níveis de tensão do sistema	Geração de ruído no sistema	4	Falha nos componentes do conversor	4	2	Executar testes de verificação	Elétrica	32

Fonte:Os autores.

5 GESTÃO DO CONHECIMENTO

5.1 Lições aprendidas

Durante o processo de desenvolvimento de um projeto diversos são os eventos imprevistos, podendo ser causados por aleatoriedades ou inexperiências. Assim sendo, torna-se válido ter um memorial coerente destes eventos, visando o crescimento da equipe envolvida num dado projeto e nos que sucederão.

Tabela 9: Condensação de eventos marcantes do projeto.

Lições aprendidas					
Tema	Fase	Impacto	O que ocorreu?	Como resolveu?	Resultados
Aquisição	Execução	Negativo	Erro na execução de furos nas chapas devido a problemas de medição	Foi realizados vários testes de medição antes da confecção das peças	Confecção correta das peças
Gestão	Execução	Negativo	Erro na configuração da comunicação dos motores, gerando perda de tempo de serviço.	Realização de uma nova configuração dos motores	Motores foram reconfigurados
Gestão	Planejamento	Negativo	Peça dimensionada de forma não compatível com a estrutura	Nova modelagem da peça levando em consideração os erros cometidos	Modelagem correta da peça
Gestão	Planejamento	Negativo	Perda de tempo por mudanças no tipo de AR Tag utilizada no projeto	Foi dedicado tempo extra para minimizar atrasos	Prosseguimento no projeto
Gestão	Planejamento	Positivo	Reuniões diárias para atualização do andamento do projeto e compartilhamento de experiências	Foram realizadas as reuniões diárias	Forte impacto positivo no andamento do projeto

5.2 Guia de uso

A configuração recomendada é possuir um computador com sistema operacional Ubuntu 18.04, *framework ROS* (ROBOTICS, s.d.) instalado e o pacote *OpenCV* 3.3.1 (CHANDEL, 2017).

Siga os seguintes passos para a instalação das dependências do projeto:

Abra seu terminal para instalar os pacotes:

Instalar o pacote *ROS* Controller:

```
$ sudo apt-get install ros-melodic-controller
```

Instalar o pacote RQT:

```
$ sudo apt-get install ros-melodic-rqt*
```

Instalar o pacote de juntas:

```
$ sudo apt-get install ros-melodic-joint*
```

Instalar os pacotes de controle do Gazebo:

```
$ sudo apt-get install ros-melodic-gazebo-ros-control
```

Instalar o pacote de cinemática Trac-IK:

```
$ sudo apt-get install ros-melodic-trac-ik
```

Instalar o pacote *MoveIt*:

```
$ sudo apt-get install ros-melodic-moveit*
```

Crie um ambiente de trabalho no seu computador digitando em seu terminal:

```
$ mkdir -p catkin_ws/src
$ cd catkin_ws
$ catkin_make
```

Entre na pasta "src"e clone o pacote externo BIR-Marker-Localization:

```
$ cd src
$ git clone -b final_settings https://github.com/Brazilian-
   Institute-of-Robotics/bir_marker_localization/tree/
   final_settings.git
```

Clone o pacote do manipulador RAJA em seguida:

```
$ git clone https://github.com/Brazilian-Institute-of-Robotics/
   raja_manipulator.git
```

Retorne para a pasta raiz do ambiente de trabalho e digite o comando para compilar os pacotes:

```
$ cd ..
$ catkin_make
```

Utilização:

A aplicação é dividida em duas partes, então é necessário abrir dois terminais para que ela funcione.

Em ambos os terminais, navegue até a pasta do ambiente de trabalho e ative o ambiente utilizando o comando:

```
$ source devel/setup.bash
```

No primeiro terminal, digite:

```
$ roslaunch raja_gazebo gazebo.launch
```

Após a aplicação do Gazebo ser executada, despause o sistema apertando o botão *play* no canto inferior da interface gráfica do Gazebo. No segundo terminal digite:

```
$ rosrun raja_move_interface move_interface.py
```

A aplicação fará com o que o manipulador saia da posição *Home* e procure uma *tag* ligada à uma caixa e um botão de emergência. Ao achá-la, o manipulador posiciona sua ferramenta no em cima do botão de emergência. Caso não seja encontrada até o fim do ciclo de escaneamento, o robô volta à sua posição inicial.

6 CONCLUSÃO

O objetivo deste manipulador é levar seu *end effector* até um ponto no espaço que representa a posição do botão de emergência da caixa que está no mesmo ambiente de trabalho. Este objetivo foi concluído com sucesso. Esta etapa ambientada no sistema de simulação Gazebo, e apresentou resultados esperados nas atividades de inspeção visual, busca, identificação e planejamento e execução de trajetória. A atividade de orientação da ferramenta em relação ao objetivo final requer ajustes.

O uso de ARUCO *tags* provou-se uma eficiente ferramenta para identificação e localização de objetos externos ao robô utilizando câmera, trazendo a caixa, mesa e manipulador para o mesmo referencial de trabalho.

Ainda se está testando o comportamento do manipulador para diferentes ferramentas e atividades mais complexas, que possam requerer cálculos mais complexos de trajetória ou diferentes pontos no espaço.

A metodologia deste relatório será utilizada para a construção física do manipulador RAJA. Este será aplicado em um ambiente de trabalho similar para realizar as mesmas atividades e ter seus parâmetros atualizados. A próxima etapa será implementar o manipulador em uma base móvel para que ele possa interagir com um objeto em uma área externa.

REFERÊNCIAS

ALL On Robots - Cylindrical robot type. s.d. <http://www.allonrobots.com/cylindrical-robot.html>. Acessado: 14-04-2020. Citado na página 16.

ANDERSON, M. *Introduction to the Robot Operating System (ROS) Middleware*. s.d. <https://elinux.org/images/1/11/IntroductionToROS_Anderson.pdf>. Acesso em: 28-04-2020. Citado na página 21.

ATHANI, V. V. *Stepper Motors: Fundamentals, applications and design*. New Delhi: New Age International (P) Limited, 2005. Citado na página 18.

AUTOMATION, O. I. *eCobra 800 Lite / Standard / Pro*. 2016. <http://www.ia.omron.com/products/family/3517/dimension.html>. Acessado: 14-04-2020. Citado na página 17.

BARINKA, I. R. B. L. *Inverse Kinematics - Basic Methods*. Prage, Czech Republic: [s.n.], s.d. Citado na página 21.

BRASIL perde a corrida da automação industrial. 2019. <https://valor.globo.com/brasil/noticia/2019/07/29/brasil-perde-a-corrida-da-automacao-industrial.ghtml>. Accessed: 2020-04-23. Citado na página 12.

C3, C. d. C. C. *Introdução Robôs Manipuladores*. 2015. <https://pt.slideshare.net/giselemoraessimas/sr-aula1-robosindustriais>. Acessado: 14-04-2020. Citado na página 17.

CARRARA, V. Introdução à robótica industrial. *INPE-Instituto Nacional de Pesquisas Espaciais, São José dos Campos*, 2015. Citado na página 15.

CHANDEL, V. S. *Install OpenCV3 on Ubuntu*. 2017. <https://www.learnopencv.com/install-opencv3-on-ubuntu/>. Acesso em: 28-04-2020. Citado na página 47.

COLLINS, D. *What is a Cartesian robot?* 2018. <https://www.linearmotiontips.com/what-is-a-cartesian-robot/>. Acesso em: 27-04-2020. Citado na página 16.

ERTHAL, J. L. Estudo de métodos para a solução da cinemática inversa de robôs industriais para implementação computacional. *UFSC, Programa de Pós-Graduação*, 10 1992. Citado na página 11.

ERTHAL, J. L. *Estudo de métodos para a solução da cinemática inversa de robôs industriais para implementação computacional*. Florianópolis: Departamento de Engenharia Mecânica, 1992. Citado 3 vezes nas páginas 19, 20 e 21.

EXECUTIVE Summary World Robotics 2018 Industrial Robots. 2019. <https://ifr.org/downloads/press2018/Executive_Summary_WR_2018_Industrial_Robots.pdf>. Accessed: 2020-04-23. Citado na página 12.

FEDERICA Web Learning. s.d. <https://mooc.federica.eu/l/introduction_to_robotics#22>. Acessado: 14-04-2020. Citado na página 18.

GAZEBO. *History*. 2014. <http://gazebosim.org/>. Acesso em: 28-04-2020. Citado na página 22.

GAZEBO Components. s.d. Http://gazebosim.org/tutorials?tut=quick_start. Acessado: 24-04-2020. Citado na página 35.

HERNANDES, A. G. *Estudo da Modelagem Robótica*. Londrina, Paraná: [s.n.], 2014. Citado na página 21.

IEEE Guide to the world of robots. s.d. <https://robots.ieee.org/robots/unimate/>. Acesso em: 15-04-2020. Citado na página 15.

INDUSTRIAL Robots: Robot Investment Reaches Record 16.5 billion USD. 2019. <https://ifr.org/ifr-press-releases/news/robot-investment-reaches-record-16.5-billion-usd/>. Acesso em: 15-04-2020. Citado na página 11.

INTERNATION Federation of Robotics. 2018. <https://ifr.org/ifr-press-releases/news/robot-investment-reaches-record-16.5-billion-usd>. Accessed: 2020-04-23. Citado na página 12.

ISO-8373. *Robots and robotic devices — Vocabulary*. Geneva, CH, 2012. Citado na página 13.

LENTH, R. V. Some practical guidelines for effective sample size determination. 2001. Citado na página 36.

MONTGOMERY, D. C. *Estatística Aplicada e Probabilidade para Engenheiros*. 5th. ed. [S.l.]: LTC, 2012. Citado na página 36.

MOVEIT. *Concepts*. s.d. <https://moveit.ros.org/documentation/concepts/>. Acesso em: 28-04-2020. Citado na página 23.

MOVEIT. *Frequently Asked Questions*. s.d. <https://moveit.ros.org/documentation/faqs/>. Acesso em: 28-04-2020. Citado na página 22.

NAKAMURA, H. H. Y. *Inverse Kinematic Solutions With Singularity Robustness for Robot Manipulator Control*. Universidade de Aveiro: Journal of Dynamic Systems, Measurement and Control, 1986. Citado na página 19.

OBR Equipamentos industriais. s.d. <http://www.obr.com.br/produtos/>. Acessado: 14-04-2020. Citado 2 vezes nas páginas 26 e 27.

OPENCV. *About OpenCV*. 2020. <https://opencv.org/about/>. Acesso em: 28-04-2020. Citado na página 22.

PIMENTA, T. T. Controle de manipuladores robóticos. *PUC-Rio de Janeiro, Departamento de Engenharia de Controle e automação*, 12 2009. Citado na página 11.

RAO, P. K. B. S. S. *Optimization in the Design and Control of Robotic Manipulators: A Survey*. [S.l.: s.n.], 1989. Citado na página 21.

ROBOTICS, O. *Install ROS Melodic*. s.d. <http://wiki.ros.org/melodic/Installation/Ubuntu>. Acesso em: 28-04-2020. Citado na página 47.

ROBOTIS. s.d. <http://www.robotis.us/>. Acessado: 12-04-2020. Citado 2 vezes nas páginas 28 e 29.

ROBOTIS MX-106-R. s.d. <http://www.robotis.us/dynamixel-mx-106r/>. Acessado: 28-04-2020. Citado na página 29.

ROBOTIS PH42-020-S300-R. s.d. <http://emanual.robotis.com/docs/en/dxl/p/ph42-020-s300-r/>. Acessado: 28-04-2020. Citado na página 29.

ROBOTIS PH54-200-S500-R. s.d. <http://www.robotis.us/dynamixel-ph54-200-s500-r/>. Acessado: 28-04-2020. Citado na página 29.

ROMANO, V. F. *Robótica Industrial: Aplicação na indústria de manufatura e processos*. São Paulo: Edgar Blucher, 2002. v. 256. Citado 2 vezes nas páginas 11 e 13.

ROS Introduction. s.d. <http://wiki.ros.org/ROS/Introduction/>. Acessado: 24-04-2020. Citado na página 34.

RUOCCO, S. R. *Robot sensors and transducers*. [S.l.]: Springer Science & Business Media, 2013. Citado na página 14.

SANTOS, F. L. R. Matheus Araújo dos. *Modelagem Matemática e Simulação da Cinemática Direta do Robô Manipulador com Seis Juntas Rotativas – 6 GL*. Campo Mourão, PR: III Simpósio de Tecnologia e Engenharia Eletrônica - III SIMTEEL, 2016. Citado na página 19.

SANTOS, V. M. F. *Robótica Industrial: Apontamentos teóricos, Exercícios para aula pratica e Problemas de exames resolvidos*. Universidade de Aveiro: Departamento de Engenharia Mecânica, 2004. Citado 3 vezes nas páginas 19, 20 e 21.

SLOTINE HARUHIKO ASADA, H. A. J. J. E. *Robot Analysis and Control*. New York: [s.n.], 1986. Citado na página 20.

SPONG, M. W.; HUTCHINSON, S.; VIDYASAGAR, M. Robot modeling and control. *Inc., ISBN-100-471-649*, Jon Wiley & Sons, 2005. Citado na página 11.

TELEDYNE Dalsa. s.d. <https://www.teledynedalsa.com/en/products/imaging/cameras/genie-nano-1gige/>. Acessado: 12-04-2020. Citado na página 30.

TF - ROS. s.d. <http://wiki.ros.org/tf>. Acessado: 24-04-2020. Citado na página 36.

WEBER, F. et al. State of the art on manipulators. *Mobile Robotics in Health Care Services*, v. 7, p. 227, 01 2001. Citado 3 vezes nas páginas 14, 18 e 19.

WIKIPéDIA. *Robot Operating System*. 2019. <https://pt.wikipedia.org/wiki/Robot_Operating_System>. Acesso em: 28-04-2020. Citado na página 21.

WIKIPéDIA. *Motor*. 2020. <https://pt.wikipedia.org/wiki/Motor>. Acessado: 27-04-2020. Citado na página 18.

APÊNDICE A

Tabelas

Lições aprendidas						
Tema	Fase	Impacto	O que ocorreu?	Como resolveu?	Resultados	Recomendações para os próximos projetos
Tecnológico	Execução	Negativo	Erro na execução de furos nas chapas devido a problemas de medição	Foi realizados vários testes de medição antes da confecção das peças	A peça foi corrigida, porém houve atraso na tarefa	Recomenda-se a verificação das dimensões mais de uma vez
Tecnológico	Planejamento	Negativo	Ausência de uma metodologia de trabalho	Reunião com objetivo de definir uma metodologia	A metodologia foi definida e atrasos nas atividades foram minimizados	Recomenda-se definir antes da execução uma metodologia de trabalho
Tecnológico	Planejamento	Negativo	Peça dimensionada de forma não compatível com a estrutura	Remodelagem da peça	Modelagem correta; atraso no cronograma	Recomenda-se a verificação das dimensões necessárias para a peça
Tecnológico	Planejamento	Negativo	Necessidade de mudança no planejamento de detecção, modificando a AR Tag utilizada	Foi realizada a mudança no planejamento	Foi dado continuidade no projeto, porém com atraso significativo na tarefa	Analisar os métodos disponíveis antes do planejamento
Gestão	Planejamento	Positiva	Reuniões diárias para atualização do andamento do projeto e compartilhamento de experiências	Foram realizadas as reuniões diariamente	Forte impacto no andamento do projeto	Realização de reuniões diárias com toda equipe do projeto

APÊNDICE B

Script em Octave para o cálculo do *workspace*

```
clc
clear
pkg load syms

syms theta1 d1 a1 alpha1 ...
      theta2 d2 a2 alpha2 ...
      theta3 d3 a3 alpha3

d2 = 0;
d3 = 0;
a1 = 0;
alpha1 = pi/2;
alpha2 = 0;
alpha3 = 0;

A_01 = [
  cos(theta1)  -sin(theta1)*cos(alpha1)  sin(theta1)*sin(alpha1)
      a1*cos(theta1);
  sin(theta1)  cos(theta1)*cos(alpha1)   -cos(theta1)*sin(alpha1)
      a1*sin(theta1);
      0        sin(alpha1)               cos(alpha1)      d1;
      0        0                         0                1];

A_12 = [
  cos(theta2)  -sin(theta2)*cos(alpha2)  sin(theta2)*sin(alpha2)
      a2*cos(theta2);
  sin(theta2)  cos(theta2)*cos(alpha2)   -cos(theta2)*sin(alpha2)
      a2*sin(theta2);
      0        sin(alpha2)               cos(alpha2)      d2;
      0        0                         0                1];

A_23 = [
  cos(theta3)  -sin(theta3)*cos(alpha3)  sin(theta3)*sin(alpha3)
      a3*cos(theta3);
  sin(theta3)  cos(theta3)*cos(alpha3)   -cos(theta3)*sin(alpha3)
      a3*sin(theta3);
      0        sin(alpha3)               cos(alpha3)      d3;
      0        0                         0                1];

A_03 = simplify(A_01 * A_12 * A_23)

pose_x = A_03[13]
```

```
pose_y = A_03[14]
pose_z = A_03[15]

d1 = 0.14;
a2 = 0.56;
a3 = 0.46;

i = 1;
k = 1;
theta1 = 0;
for theta1 = -pi:0.1:pi
    for theta2 = -pi:0.3:pi;
        for theta3 = -pi:0.3:pi
            aux_x = pose_x;
            aux_y = pose_y;
            aux_z = pose_z;
            if aux_z > 0 && aux_y > 0
                x(i) = aux_x;
                y(i) = aux_y;
                z(i) = aux_z;
                i = i + 1;
            end
            k = k + 1;
            disp(k)
        end
    end
end

scatter3(x, y, z)
xlabel("y")
ylabel("z")
zlabel("z")
```