Federação das Indústrias do Estado da Bahia
CENTRO UNIVERSITÁRIO SENAI CIMATEC

Engenharia Mecânica

Trabalho de Conclusão do Curso

Desenvolvimento do robô de inspeção.

Apresentada por: Bruno Rodrigues
Bruno de Sousa
Frederico Garcia
Leandro S O Nozela
Victor V. Rezende

Orientador: Prof. Marco Reis, M.Eng.
Co-orientador: João Lucas da Hora

Dezembro de 2019

Bruno Rodrigues
Bruno de Sousa
Frederico Garcia
Leandro S O Nozela
Victor V. Rezende

Desenvolvimento do robô de inspeção.

Trabalho de Conclusão do Curso apresentada ao , Curso de Engenharia Mecânica do Centro Universitário SENAI CIMATEC, como requisito parcial para a obtenção do título de **Bacharel em Engenharia**.

Área de conhecimento: Interdisciplinar

Orientador: Prof. Marco Reis, M.Eng.

Salvador
Centro Universitário SENAI CIMATEC
2016

Dedico este trabalho a ...

Agradecimentos

Resumo

Palavras-chave: Robô de Inspeção, Linhas de Transmissão, Navegação, Cinemática Inversa, Manipuladores

Abstract

Keywords: Inspection Robot,Transmission Lines,Navigation,Inverse Kinematics,Manipulators

Sumário

Lista de Tabelas

Lista de Figuras

Lista de Siglas

ELIR	*Electrical Inspection Robot*
URDF	*Universal Robot Description Format*
ROS	*Robotic Operation System*
QFD	*Quality Function Deployment*
SOTA	*State of the Art*
USB	*Universal Serial Bus*

Lista de Simbolos

∂ .... Bla bla bla
$\prod$ .... ble ble ble
∂ .... Bla bla bla
$\prod$ .... ble ble ble
∂ .... Bla bla bla
$\prod$ .... ble ble ble
∂ .... Bla bla bla
$\prod$ .... ble ble ble
∂ .... Bla bla bla
$\prod$ .... ble ble ble
∂ .... Bla bla bla
$\prod$ .... ble ble ble
∂ .... Bla bla bla
$\prod$ .... ble ble ble
∂ .... Bla bla bla
$\prod$ .... ble ble ble
∂ .... Bla bla bla
$\prod$ .... ble ble ble
∂ .... Bla bla bla
$\prod$ .... ble ble ble
∂ .... Bla bla bla
$\prod$ .... ble ble ble
∂ .... Bla bla bla
$\prod$ .... ble ble ble
∂ .... Bla bla bla
$\prod$ .... ble ble ble
∂ .... Bla bla bla
$\prod$ .... ble ble ble
∂ .... Bla bla bla
$\prod$ .... ble ble ble
∂ .... Bla bla bla
$\prod$ .... ble ble ble
∂ .... Bla bla bla
$\prod$ .... ble ble ble
∂ .... Bla bla bla
$\prod$ .... ble ble ble
∂ .... Bla bla bla
$\prod$ .... ble ble ble

Capítulo Um

Introdução

"Faça ou não faça, tentativa não há."

(Mestre Yoda)

Quando se ouve falar em robôs, logo associa-se a algo de extrema complexidade. Isso ocorre, sumariamente, devido à falta de informações simplificadas sobre o tema, ou devido a dificuldade de acesso a tais conteúdos. A palavra robótica é derivada da palavra robô, que, segundo (GONçALVES, 2007), é um dispositivo eletromecânico capaz de realizar tarefas de maneira autônoma ou pré-programada, e faz menção a ciência que estuda, cria e aplica robôs. No meio educacional, a palavra didática está presente de forma quase que impreterível, afinal, materiais didáticos, livros, projetos e a própria didática como um instrumento qualificador do professor, são componentes fundamentais do cotidiano educacional. Porém é notório que barreiras na educação da atualidade estão sendo quebradas, onde o estudante deixa de frequentar as salas de aula, tornando assim o professor apenas um facilitador do aprendizado do aluno, um tutor. A tutoria é um método muito utilizado para efetivar uma interação pedagógica. Segundo (Sá, 1998), na educação à distância, o tutor recebe o significado de "orientador de aprendizagem do aluno solitário e isolado". O sistema de tutoria torna mais fácil o acesso do aluno ao conhecimento, pois o professor passa a ser apenas um orientador, desta maneira aluno torna-se independente na busca das informações, assim, simplificando o aprendizado do aluno. Percebendo essa nova dinâmica da educação, e a falta de informações simplificadas sobre robótica, notou-se a possibilidade de criar um kit didático, para incentivar as pessoas através de desafios e simplificar as informações em torno da robótica.

1.1 Organização do Trabalho de Conclusão do Curso

O documento está organizado em cinco capítulos, seguindo a seguinte estrutura:

Capitulo 1 - Introdução: Faz a contextualização do âmbito no qual a pesquisa proposta está inserida. Apresenta, portanto, a problemática, objetivos e como este projeto Theoprax de conclusão de curso está estruturado

Capítulo 2 - Referencial Teórico: Apresenta a base teórica necessária para o

desenvolvimento do projeto.

Capítulo 3 - Metodologia: Define o método adotado para o desenvolvimento do projeto, explicitando seu fluxo de atividades e premissas necessárias para aplicar a metodologia.

Capítulo 4 - Desenvolvimento: Exibe os procedimentos realizados e resultados obtidos através de testes, unitários e integrados, durante o desenvolvimento do projeto.

Capítulo 5 - Conclusão: Apresenta as conclusões, contribuições e algumas sugestões de atividades de pesquisa a serem desenvolvidas futuramente.

Capítulo Dois

Fundamentação Teória

"Elementar , meu caro Watson."

(Scherlock Holmes)

O termo robô vem da palavra tcheca robota que tem como uma das possíveis traduções "trabalhador forçado" e ganhou o significado atual após o escritor tcheco Karel Capek (1809 - 1938), na sua obra de ficção científica "R.U.R. Rossumovi Univerzální Roboti", associar o termo às máquinas criadas pelo personagem principal para servi-lo. Mas a ideia de algo que desenvolva atividades de maneira autônoma é apresentada ao mundo muito tempo antes. (**??**) diz: "Se cada instrumento pudesse realizar sozinho a sua tarefa, obedecendo ou antecipando a nossa vontade, [...] os feitores não precisariam de servos, nem os senhores de escravos."

Diversas obras da ficção retratam diferentes tipos de robôs criados de forma a reproduzir comportamentos semelhantes aos de um ser humano. Com o passar do tempo, juntamente com o avanço tecnológico nas áreas da eletrônica, mecânica e informática, a construção dessas máquinas se tornou possível. A indústria observou nos robôs, o potencial para automatizar e otimizar as linhas de processo, onde atividades que pudessem demandar mais tempo se fossem executadas por seres humanos, seriam executadas de forma muito mais rápida e precisa com a utilização de máquinas programadas e autônomas, aumentando a produção.

A (**??**) define um robô como "mecanismo programável atuado em dois ou mais eixos com um grau de autonomia, movendo-se dentro do seu ambiente, para executar tarefas pretendidas". É resultado da integração de componentes como: Sensores; atuadores; unidade de controle; unidade de potência e manipulador mecânico. Sensores são os componentes que fornecem parâmetros sobre o ambiente em que o robô se encontra e sobre o comportamento do próprio sistema robótico. Já os atuadores são os dispositivos que movimentam as partes, quando convertem energia elétrica, hidráulica ou pneumática em mecânica. A energia necessária para o funcionamento dos atuadores é fornecida pela unidade de potência.

O gerenciamento dos parâmetros necessários para que o robô realize suas tarefas é de responsabilidade da unidade de controle. De onde também são emitidos os comandos para a movimentação. O manipulador mecânico é o conjunto de componentes estruturais

do robô, elos ou links, conectados entre si por articulações comumente denominadas de juntas. Graus de liberdade, segundo (**??**) "É o número mínimo de variáveis independentes de posição que precisam ser especificadas para se definir inequivocamente a localização de todas as partes de um mecanismo".

2.1 Cinemática

A cinemática é o ramo da física que descreve o movimento de um corpo, determinando características como posição, velocidade e aceleração. Na robótica, o estudo cinemático resulta em um conjunto de equações que caracterizam o movimento do robô, a complexidade da solução varia com a quantidade de graus de liberdade que esse robô tem. Em um manipulador mecânico composto por links que são conectados por juntas, cada conjunto link-junta caracteriza um grau de liberdade. Dessa maneira, um robô com n conjuntos link-junta tem n graus de liberdade, sendo o primeiro link a base de sustentação do robô no mundo e o último, onde está a seu end-effector.

2.1.1 Cinemática Direta

A cinemática direta é a solução para a movimentação de um robô com cálculo da posição e orientação do end-effector a partir de dadas posições das juntas. A notação de Denavit-Hartenberg é uma ferramenta utilizada para coordenar a descrição cinemática de sistemas mecânicos articulados com n graus de liberdade.

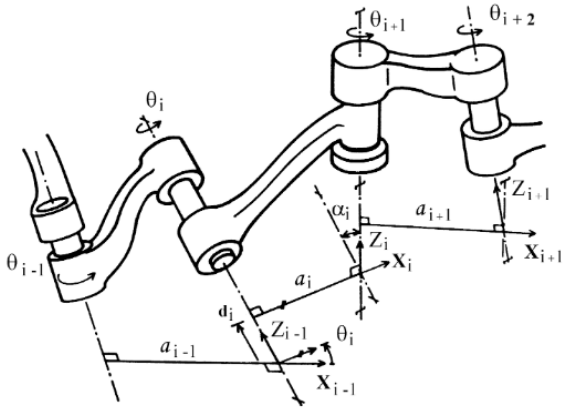


Figura 2.1: Parâmetros de Denavit-Hartenberg

Fonte: (**??**)

A figura mostra dois *links* ligados por uma junta de superfícies deslizantes uma sobre a outra. Um eixo de uma junta estabelece a conexão de dois *links*. Segundo (**??**), os eixos das juntas devem ter duas normais conectadas a eles, uma para cada um dos

links. Assim a posição relativa destes dois *links* conectados ($i-1$ e i) é dada por d_i, que é a distância medida ao longo do eixo da junta entre suas normais. O ângulo de junta θ_i entre as normais é medido em um plano normal ao eixo da junta. Dessa forma, d_i e θ_i são a distância e o ângulo entre os *links* adjacentes. Determinam a posição relativa de *links* vizinhos.

Um *link* pode apenas ser conectado a dois outros *links* ($i-1$ e $i+1$). Assim, dois eixos de juntas são estabelecidos em ambos terminais de conexão. Os *links* mantém uma configuração fixa entre as juntas e podem ser caracterizados pelos parâmetros a_i e α_i. O parâmetro a_i é a menor distância medida ao longo da normal comum entre os eixos da junta, chamado de comprimento de *twist*, já o α_i é o ângulo de *twist*. Esses quatro parâmetros determinam a estrutura do *link*, parâmetros da junta e a posição relativa aos *links* vizinhos.

A representação de Denavit-Hartenberg (**??**) tem como resultado uma matriz 4 x 4 representando cada sistema de coordenadas do *link* na junta em relação ao *link* anterior. Essa matriz é obtida através do produto das transformações: Translação de uma distância d_i ao longo do eixo Z_{i-1} para trazer os eixos X_{i-1} e X_i na coincidência; Rotação no eixo Z_{i-1} de um ângulo θ_i para alinhar os eixos X_{i-1} e X_i; Translação ao longo do eixo X_i de uma distância a_i para trazer as duas origens na coincidência; Rotação do eixo X_i um ângulo α_i para trazer os dois sistemas de coordenadas na coincidência. Isso resulta na matriz de transformação homogênea ${}^{i-1}A_i$.

$$ {}^{i-1}A_i = T_{z,d}T_{z,\theta}T_{x,a}T_{z,\alpha} \tag{2.1} $$

$$ {}^{i-1}A_i = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos\theta_i & -sin\theta_i & 0 & 0 \\ sin\theta_i & cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\alpha_i & -sin\alpha_i & 0 \\ 0 & sin\alpha_i & cos\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.2} $$

$$ {}^{i-1}A_i = \begin{bmatrix} cos\theta_i & -cos\alpha_i sin\alpha_i & sen\alpha_i sin\theta_i & a_i cos\theta_i \\ sin\theta_i & cos\alpha_i cos\theta_i & -sin\alpha_i cos\theta_i & a_i sin\theta_i \\ 0 & sin\alpha_i & cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.3} $$

2.1.2 Cinemática Inversa

Segundo (**??**) os robôs estão em um espaço onde o objeto a ser manipulado tem sua posição expressa no sistema de coordenadas do ambiente. Com o objetivo de controlar a posição e orientação do *end-effector* do robô, a solução da cinemática inversa é mais

adequada. A cinemática inversa consiste em, partindo de uma posição e orientação desejada, calcula-se as posições das juntas para que o robô alcance esse objetivo, é o processo inverso da cinemática direta.

Há de se observar que a cinemática inversa pode ou não ter solução, caso a posição de interesse esteja fora do espaço de trabalho do robô, não há posições de juntas que execute a tarefa. Nos momentos em que a posição desejada pode ser alcançada, podem existir mais de uma solução. Um ponto importante na solução da cinemática inversa é, quando há mais de uma solução deve-se atentar para qual delas é a melhor opção, levando em consideração o ambiente em que o robô se encontra, principalmente os obstáculos à sua volta. A demanda energética para a execução dos possíveis movimentos e o esforço qual as juntas serão submetidas nesta ação, é crucial para o planejamento da movimentação do robô.

2.2 Modelagem Cinemática de um Braço Planar

O robô ELIR tem na sua estrutura, braços que se movimentam apenas em dois eixos, x e z, através da atuação de duas juntas, podendo assim ser modelado cinematicamente como um braço planar do tipo RR. A figura a seguir mostra um exemplo desse braço, RR por ter duas juntas rotativas, que se movimenta no plano $x - y$:

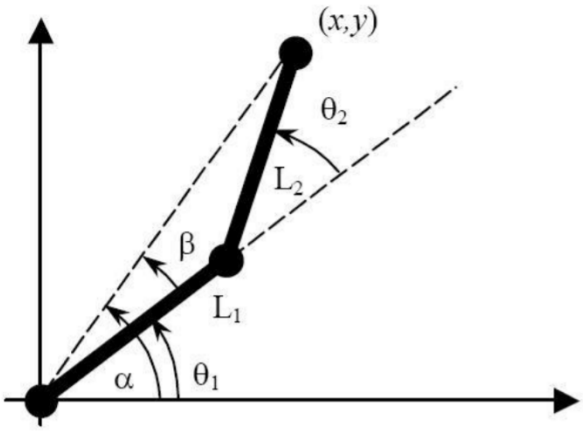


Figura 2.2: Braço planar do tipo RR

Fonte: (**??**)

Usando a análise da cinemática direta, consegue-se determinar a posição do *end-effector* com base nos ângulos θ_1 e θ_2 e nas dimensões L_1 e L_2. Logo tem-se que:

$$x = L_1 cos(\theta_1) + L_2 cos(\theta_1 + \theta_2) \tag{2.4}$$

$$y = L_1 sen(\theta_1) + L_2 sen(\theta_1 + \theta_2) \tag{2.5}$$

Aplicando a lei dos cossenos ao triângulo formado pelo braço e pela linha entre a origem do braço e o seu *end-effector* obtém-se:

$$\theta_2 = \pm arccos\frac{(x^2 + y^2 - (L_1)^2 - (L_2)^2)}{2L_1L_2} \tag{2.6}$$

Para determinar o θ_1 considera-se a relação trigonométrica:

$$tan(A - B) = \frac{tan(A) - tan(B)}{1 + tan(A)tan(B)} \tag{2.7}$$

e tomando:

$$tan(\beta) = \frac{L_2 sin\theta_2}{L_1 + L_2 cos\theta_2} \tag{2.8}$$

tem-se que:

$$\theta_1 = arctan[\frac{y(L_1 + L_2 cos\theta_2) - xL_2 sin\theta_2}{x(L_1 + L_2 cos\theta_2) - yL_2 sin\theta_2}] \tag{2.9}$$

Assim é possível fazer a solução da cinemática inversa para um braço planar RR.

2.3 Desenvolvimento de Robôs

Para o desenvolvimento de sistemas robóticos, é necessária a integração de vários dispositivos, assim sendo necessário utilizar ferramentas e tecnologias que poupem tempo no desenvolvimento, de forma a facilitar o processo de comunicação entre as diversas camadas de abstração. As camadas de abstração se referenciam ao alto e baixo nível da máquina, onde baixo nível é uma referência para aplicações mais simples, que estão mais próximas da linguagem da máquina, como por exemplo aplicação de comunicação somente via *bytes*. Um exemplo de um elemento que está numa camada de abstração de alto nível é uma Interface Homem-Máquina , onde o usuário consegue interagir com a máquina diretamente, sem ter que necessariamente entender o seu funcionamento interno.

2.3.1 Framework

Em ambientes computacionais, a utilização de ferramentas para realização de atividades e desenvolvimento de soluções é de extrema importância. Estas ferramentas podem ser softwares específicos para execução de uma determinada atividade ou *frameworks*. Segundo (**??**) "*Frameworks* são estruturas de classes que constituem implementações incompletas que, estendidas, permitem produzir diferentes artefatos de software". Os *frameworks* em geral permitem o desenvolvimento de soluções computacionais baseadas em determinadas funcionalidades, seguindo uma estrutura definida pelo *framework*. De acordo com (**??**) os *frameworks* definem uma arquitetura para um conjunto de subsiste-

mas, dando os construtores necessários para a sua criação.

A principal característica de um *framework* é a sua capacidade de reutilização, afinal a sua utilização permite que diversos conjuntos de produtos possam ser gerados partindo de uma única estrutura que possua os conceitos mais gerais. Segundo (**??**) *frameworks* podem ser classificados em dois tipos principais: *Frameworks* de Aplicações Orientado a Objetos e *Frameworks* de Componente. Os *frameworks* orientados a objetos geram famílias de aplicações orientadas a objetos e seus pontos de extensão são definidos como classes abstratas ou interfaces, onde se estendem por cada instância da família de aplicações. Para *frameworks* de componentes, o suporte é previsto para componentes que sigam um determinado modelo, possibilitando que as instâncias destes componentes sejam acopladas ao *framework*. Também são estabelecidas as condições necessárias para que um componente seja executado, regulando a sua interação entre as instâncias de outros componentes.

Os *frameworks* utilizados para robótica, são extremamente importantes, pois o uso de suas ferramentas possibilita o desenvolvimento e criação das soluções computacionais e códigos necessários para cada funcionalidade de um robô, de forma que o funcionamento delas em conjunto seja otimizado pela natureza do *framework* de realizar a compatibilização entre as estruturas.

2.3.2 Simulação

Em sistemas complexos, onde diversas variáveis definem o seu funcionamento, e por consequência as suas respostas a determinados estímulos, torna-se extremamente difícil e irresponsável executar a sua fabricação antes de realizar uma validação prévia de seu funcionamento.

Este tipo de procedimento de análise prévia do comportamento de um sistema é chamado de simulação. De acordo com (**??**) "Simulação refere-se a uma ampla coleção de métodos e aplicações para imitar o comportamento do sistema real, por meio de um computador com um *software* apropriado". Diversos tipos de sistemas se utilizam da ferramenta de simulação para validar previamente o funcionamento de projetos.

O processo de utilização de uma simulação consiste em basicamente recriar o sistema em questão em um ambiente computacional e então são fornecidas as entradas para o sistema, as rotinas de tratamento destas entradas e por fim as saídas da simulação.

Em sistemas robóticos, uma ferramenta extremamente útil e bastante utilizada, é a simulação. Segundo (**??**)

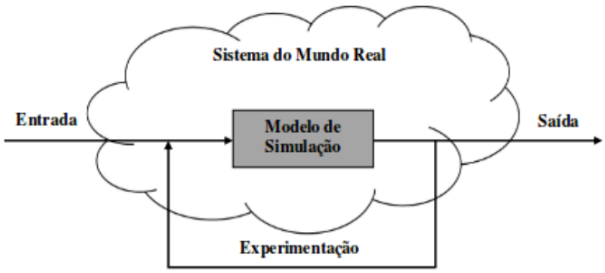


Figura 2.3: Diagrama de funcionamento de um processo de simulação

Fonte: (**??**)

> "Quando se trabalha com robótica, o uso de uma simulação é de importância significante. Por um lado, ela permite a validação de diferentes alternativas durante o design do sistema robótico, levando assim, a melhores decisões e preservação de custos. Por outro lado, auxilia o processo de desenvolvimento de *software*, disponibilizando uma reposição para robôs que não estejam em mãos".

Através do uso de *softwares* de simulação é possível criar uma representação computacional não só o modelo físico de um robô, mas também os parâmetros referentes à objetos do ambiente no qual o mesmo será posto em funcionamento. A avaliação prévia da execução das tarefas e do funcionamento do robô, permite a observação do comportamento do sistema em determinadas situações, facilitando assim, a tomada de decisões mais efetivas no processo de desenvolvimento do protótipo real.

2.3.3 Odometria

A odometria consiste no cálculo para estimar a mudança de posição do robô no tempo, onde isso pode se dar por meio de diversos dispositivos que possibilitem o cálculo de deslocamento. Onde segundo (**??**), "Odometria - a medição da distância - é um método fundamental usado por robôs para navegação". A medição de tempo é fácil utilizando o *clock* interno do computador embutido. Medir velocidade é mais difícil: em alguns robôs educacionais utilizam codificadores são usados para contar as rotações da rodas, enquanto em outros a velocidade é estimada das propriedades dos motores.

No caso da análise de deslocamento do robô na linha por meio de roldanas, o movimento é caracterizado como linear, já que o deslocamento ocorre em somente uma direção, analogamente a odometria utilizada é a linear, onde o deslocamento pode ser calculado simplesmente pela equação 2.10 onde s representa o espaço caminhando, v a velocidade e t o tempo.

$$s = v * t \tag{2.10}$$

Utilizando o medidor de tempo interno do computador embutido nos sistemas robóticos, pode se calcular a variação de espaço para um tempo muito pequeno, onde esses pequenos incrementos são somados ou subtraídos para encontrar o deslocamento do robô.

A velocidade de deslocamento das roldanas pode ser encontrada utilizando a equação 2.11 com as informações do raio da roldana r e a sua velocidade de giro w em radianos por segundo. A informação da velocidade de giro da roldana geralmente é extraída dos servomotores utilizados para tração.

$$v = 2\pi * r * w \tag{2.11}$$

O cálculo da odometria por meio da velocidade das rodas é denominado no âmbito da robótica como odometria de roda, *wheel odometry* em inglês, porém, outras técnicas são utilizadas, já que existem diversos tipos de deslocamento diferentes. Outro tipo aplicação muito encontrada é a odometria visual, que segundo (**??**) "Odometria Visual (OV) é o processo de estimação do deslocamento de um agente (ex: veículo, humano e robô) utilizando a entrada de uma ou múltiplas câmeras conectadas a ele". Os domínios da aplicação incluem robótica, realidade aumentada, automotiva e 'computadores vestíveis'.

2.3.4 Gestão de Energia

O conceito de gestão de energia se dá pela forma como a energia elétrica é utilizada em um sistema composto de diversos dispositivos elétricos e eletrônicos. Para sistemas robóticos, este conceito representa um fator importante para garantir uma operação autônoma de qualidade. Os robôs quando nesse tipo de operação, geralmente não dispõem de uma fonte de energia constante, e portanto, são geralmente alimentados por baterias e tendo interação por meio de conexões sem fio.

O uso de diversos dispositivos eletrônicos de baixo consumo energético, como sensores e interfaces microcontroladas, podem não se mostrar um problema para um curto período de operação, porém, para maximizar o tempo da atividade exercida pelo robô, é necessário encontrar uma forma eficiente de gerir a operação dos dispositivos conectados na rede de alimentação. Segundo (**??**)

"Gestão de energia é um conceito importante em redes de sensores, porque

> uma estrutura de energia cabeada geralmente não está disponível e um conceito óbvio é utilizar a energia disponível da bateria de forma eficiente".

Quanto mais atividades diferentes o robô desempenha maior será a demanda de energia entre os dispositivos interconectados, isso faz com que seja necessário que os desenvolvedores busquem uma forma de otimizar o custo de energia individual das atividades e do fluxo de operação como um todo. Os diversos dispositivos utilizados em sistemas robóticos fazem com que o mesmo se utilize de diferentes níveis de tensão e corrente, já que comumente, os dispositivos utilizados são comerciais, e devido às diferenças das suas características e parâmetros, definidos por empresas diferentes, responsáveis pela produção e fabricação das ferramentas, é necessário que a gestão de energia leve em consideração a compatibilidade entre diferentes dispositivos.

2.3.5 Conceito de segurança e Integridade

Em diversas áreas, é comum a verificação das condições antes da execução de atividades, a aviação é um grande exemplo de uso desse conceito. Neste seguimento, o *checklist* é utilizado toda vez antes de um avião decolar, assim é possível verificar se os sistemas vitais para o vôo estão em ordem. O principal objetivo dessa ação é identificar os riscos que existem para o cumprimento da atividade.

Para que um dispositivo robótico execute as tarefas para as quais ele foi desenvolvido, deve-se verificar se os seus sistemas, como um todo, e os componentes individualmente, estão em condições de funcionamento, garantindo assim a integridade do sistema como um todo. Essa análise deve ser feita levando em conta a importância de cada sistema e de cada componente desses sistemas, a fim de aumentar a capacidade de operação em condições adversas do robô. Podem existir sistemas que, mesmo quando não estão operando adequadamente, não comprometem a execução da missão do robô.

2.3.6 Comunicação em sistemas robóticos

Dispositivos eletrônicos são capazes de realizar transmissão de dados, afinal, a interconexão entre eles é de extrema importância em sistemas em que existam diversos dispositivos responsáveis por funções distintas. Dispositivos que se comunicam entre si, são capazes de criar uma rede em todo o sistema, permitindo um aumento na confiabilidade das funções do sistema, através da troca de informações de parâmetros que venham a ser importantes para o funcionamento do sistema como um todo.

Para que os dispositivos possam se comunicar entre si, os mesmos adotam o que se chama de protocolos de comunicação. Protocolos de comunicação são arquiteturas que estabelecem a troca de dados entre dispositivos eletrônicos. Os dispositivos comerciais possuem diferentes tipos de protocolos de comunicação e por isso, torna-se extremamente importante se atentar a qual protocolo utilizar durante a conceituação de um projeto que se tenha a necessidade da interconexão de dispositivos.

Uma das formas mais comuns de se realizar a transmissão de dados entre dispositivos embarcados é a comunicação serial. (**??**) define a comunicação serial como um envio de *bits* de forma serial, similar a uma fila. Possuindo dois canais principais: o canal TX para envio e o canal RX para recebimento. Dentro desse processo de comunicação alguns parâmetros devem ser levados em conta, como a taxa de transmissão de dados (*BaudRate*); bits de paridade, para assegurar que o número de bits no campo de dados é par ou ímpar; bits de parada para indicar o início ou fim de uma comunicação.

Outro meio de comunicação muito utilizado é o USB (*Universal Serial Bus*), criado com a intenção de tornar a comunicação serial mais simplificada e com uma taxa de transmissão muito mais elevada. Os cabos conectores USB possuem geralmente quatro fios condutores, sendo dois deles para alimentação e dois outros cabos de dados. Os cabos de dados são nomeados como D+ e D-, onde a comunicação entre os dispositivos se dá pela variação de tensão entre estes dois sinais. Dentro de ampla complexidade como um robô, onde diversos dispositivos necessitam estar trocando informações, a utilização de protocolos de comunicação serial se tornam extremamente importantes para a garantia da confiabilidade na execução de tarefas e operações.

Capítulo Três

Metodologia

"Tudo o que temos de decidir é o que fazer com o tempo que nos é dado."

(Gandalf)

De acordo com (**??**) metodologia é "o conjunto de métodos e técnicas aplicadas para um determinado fim. É o caminho percorrido, a maneira utilizada para atingir um objetivo". Por certo, descreve os métodos que padronizam uma produção, visando a chegada em um resultado. Em trabalhos acadêmicos a sua importância vai além de descrever o processo de confecção do projeto mas também permite que o mesmo possa ser replicado por outros pesquisadores.

A metodologia aplicada para para o projeto toma como base o desenvolvimento de sistemas robóticos, nesse caso sendo voltada para o desenvolvimento de um sistema de movimentação robótico, presente no robô *ELIR*. A divisão do projeto em fases maiores e menores facilita o fluxo para o desenvolvimento, assim sendo definidas quatro partes maiores, sendo elas: conceituação, *design*, desenvolvimento e operacionalização. O fluxo do projeto está explicitado no Figura 3.1 a seguir:

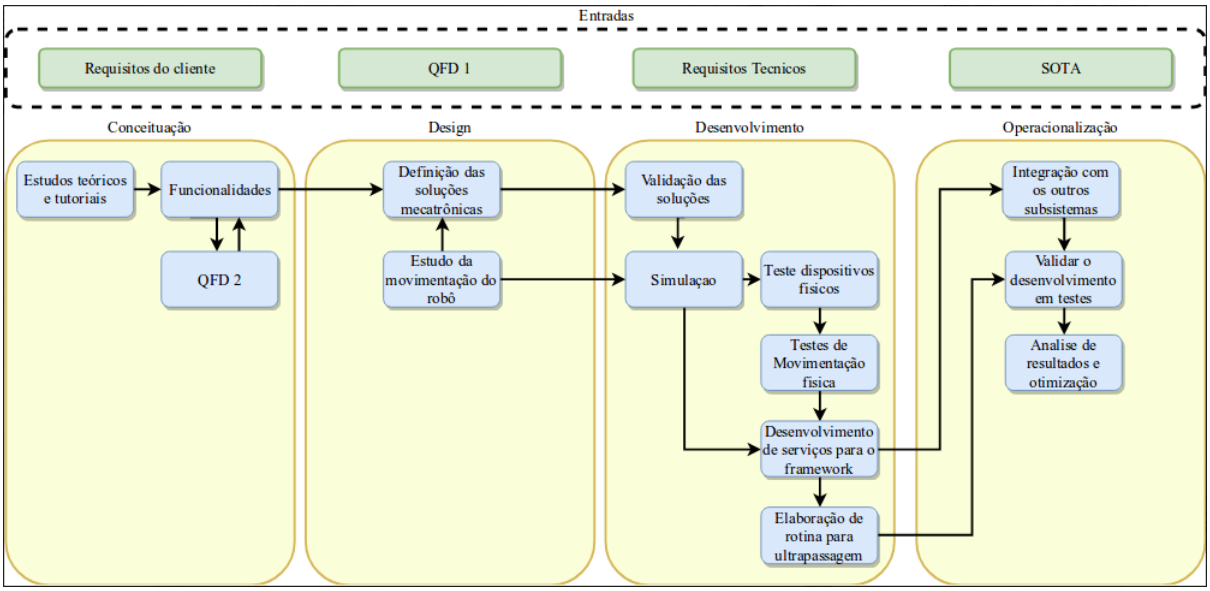


Figura 3.1: Fluxograma de desenvolvimento

Devido a complexidade envolvida nesse tipo de sistema, são utilizadas ferramentas específicas de desenvolvimento, assim como o cliente fornece certos parâmetros iniciais para impulsionar o projeto e guiar desenvolvimento para um resultado satisfatório. São

consideradas entradas para a metodologia os requisitos do cliente, requisitos técnicos, um *QFD* inicial, denominado *QFD* 1 e o Estudo do Estado da Arte (*SOTA*).

O requisito consiste na definição documentada de uma propriedade ou comportamento que um produto ou serviço particular deve atender. Existem os requisitos do cliente, no qual são as necessidades e as expectativas do cliente, e os requisitos técnicos possui uma visão técnica para atender as necessidades do cliente e os objetivos do projeto.

Durante a fase inicial do projeto, foi conversado com o cliente e deixado explícito seus requisitos para o *ELIR*, sendo eles:

- Realizar as funções de forma autônoma;
- Transpor obstáculos e cadeia de isoladores;
- Deslocar-se através do consumo de baterias;
- Deslocamento/movimento realizada por servomotores.

Foi determinado pelo cliente para que o projeto desempenhe corretamente os seguintes requisitos técnicos:

- Desempenho de deslocamento de 15km por dia
- Velocidade de deslocamento médio sem obstáculos de 0.5 m/s
- Ultrapassagem de obstáculos de volume máximo de 410x330x150mm
- Autonomia de potência de 2 horas
- Sistema operacional Linux,
- *Backend* em C++ e Python
- *Framework* ROS Kinetic Kame.

A ferramenta de Desdobramento da Função Qualidade (*QFD*) torna-se importante para guiar o projeto e a incorporar as reais necessidades do cliente. Por meio de um conjunto de matrizes parte-se dos requisitos expostos pelos clientes e realiza-se um processo de "desdobramento"transformando-os em especificações técnicas do produto. Esse desdobramento entre os requisitos do cliente, influencia no desenvolvimento, permitindo encontrar o que impacta mais no resultado final do projeto, assim fazendo com que a prioridade de certas atividades mude.

O estudo do estado da arte é o mapeamento que possibilitará o conhecimento e/ou reconhecimento de estudos que estão sendo, ou já foram realizados com temáticas, ou linhas de pesquisa , iguais ou parecidas a que está sendo estudando. No caso de projetos desenvolvidos em conjunto que incorporam diversas teses acadêmicas, esse estudo facilita a continuação do trabalho e dá uma base o projeto.

3.1 Conceituação

A fase da conceituação consiste na criação de um conceito para o sistema, sendo assim recolhido todo o embasamento teórico necessário para a confecção do projeto. Assim fazendo com que seja elaborada uma ideia para o sistema, o que é a base para todo o projeto, guiando as próximas fases.

As entradas do cliente são de suma importância para essa etapa, onde as mesmas são a base para a ideia do sistema. Com o estudo teórico do que será necessário e o uso das ferramentas como o *QFD*, é possível elaborar as funcionalidades que serão desempenhadas pelo sistema. A elaboração de um segundo *QFD* por parte da equipe acontece em paralelo com a elaboração das funcionalidades e se utiliza do *QFD* 1 junto com os requisitos técnicos e do cliente, buscando conceituar o sistema de forma concreta.

As funcionalidades recebem entradas e saídas, sendo assim, interligadas, esse tipo de metodologia se mostra muito eficiente pois consegue dividir o robô em subsistemas e funções a serem desempenhadas, podendo assim dar uma idéia de como será o seu funcionamento e troca de informações internas. Com a definição das funcionalidades do sistema, é possível partir para a forma da idéia, como ela será aplicada, o que acontece na etapa de *Design*;

3.2 Design

Com um conceito firme para o sistema, a etapa de *design* consiste na forma que a idéia irá ter, para que a mesma seja possível. Com as funcionalidades definidas, é possível decidir quais ferramentas devem ser utilizadas no projeto, de forma a garantir que as mesmas sejam executadas.

Por se tratar do desenvolvimento de um sistema de movimentação, é importante que seja realizado um estudo da forma como será necessário se realizar o movimento do robô, já que isso impacta profundamente na seleção das ferramentas, esse estudo consiste na busca do entendimento de como será sua aplicação real, e está relacionado também

com a simulação, que colabora para o sucesso dessa etapa. São tomados como critérios para a escolha da ferramentas: A quantidade de informação sobre cada ferramenta que há disponível, suporte da comunidade que a utiliza e os tutoriais que cada ferramenta possui. O estudo da movimentação é de grande valia no *design* das soluções mecatrônicas pois, conhecendo as maneiras quais o robô tem que se movimentar, definir os recursos necessários para esse objetivo se torna mais fácil.

3.3 Desenvolvimento

O desenvolvimento consiste na aplicação prática da idéia, sendo a parte que demanda mais tempo e a partir dela, já é possível ter a noção de como será o dispositivo físico final. Contém toda a produção de software, estruturas necessárias para o projeto, e também a construção do protótipo. Com a definição das ferramentas realizada na etapa de *design*, é possível começar a aplicação no sistema de interesse, validando o que foi decidido anteriormente.

3.3.1 Validação das ferramentas

A etapa de validação das ferramentas é onde se realiza os estudos e testes para compreender o funcionamento das mesmas e verificar se suas mecânicas e funcionalidades são adequadas para a solução e desenvolvimento do projeto. Uma vez que a ferramenta esteja escolhida, é realizada a análise do seu funcionamento, seus aspectos gerais, configurações, e como integrá-las ao desenvolvimento do projeto.

Após realizado o estudo da ferramenta, e de como suas mecânicas funcionam e podem auxiliar no desenvolvimento das funcionalidades do projeto, são realizados os primeiros usos das ferramentas, por meio de pequenos testes específicos, a fim de buscar o entendimento amplo de como se operacionalizar e implementar as soluções a partir das funcionalidades das ferramentas.

Uma vez que os testes se mostram efetivos e o conhecimento sobre as suas funcionalidades esteja adquirido, a ferramenta torna-se válida, e pode ser utilizada no desenvolvimento do projeto, podendo ser utilizada em todas as etapas onde se mostre necessária

3.3.2 Movimentação simulada/em simulação

Para a validação das ferramentas, o uso de simulações computacionais é fundamental, devido que a simulação consegue prever os comportamentos do protótipo antes do mesmo estar em operação e com uso das ferramentas disponíveis na robótica e assim validando a maioria das ferramentas e estratégias. A simulação torna-se presente em todos os testes, desde a validação de ferramentas definidas durante a fase de *Design* até do robô em operação, sendo uma poderosa ferramenta para validação dos dispositivos e a movimentação física.

3.3.3 Teste com dispositivos físicos e Movimentação física

Antes de realizar testes do robô se movimentando é necessário garantir que todos os dispositivos físicos estejam funcionando corretamente. Durante essa fase deverá ser realizado a montagem do robô e garantir que esteja conforme a simulação. Logo após, é de extrema importância realizar simulações, testes de esforço, velocidades e posição dos servomotores utilizados na estrutura para que não haja nenhum imprevisto durante os próximos testes de operação.

Com os testes de dispositivos físicos realizados unitariamente, é necessário integrá-los para realizar os testes de movimentação física. Esse teste é realizado tanto na simulação quanto em operação em linha, sendo importante verificar se o mesmo está andando corretamente, observando influência externa do vento.

3.3.4 Desenvolvimento de serviços para framework e rotina para ultrapassagem

A estrutura de serviços disponibilizada pelo *framework*, são subrotinas em código para desempenhar uma função específica, a comunicação é feita por um par de mensagens, uma de solicitação e outra de resposta, que quando necessário fazer uso do serviço, uma mensagem de solicitação é enviada, logo após o serviço executa o código contigo nele e retorna uma mensagem de resposta.

Para realizar o rotina de ultrapassagem como um todo é conveniente separar em serviços as partes dos movimento de ultrapassagem, facilitando a identificação de erros e inconsistências. Com os serviços feitos e testados é preciso unificá-los para realizar a ultrapassagem. É importante organizá-los de forma sequencial e analisar como cada serviço se comporta para garantir que foram bem codificados, e se necessário realizar

ajustes na programação.

3.4 Operacionalização

A operacionalização põe em funcionamento todo o conjunto dos dispositivos, sistemas e rotinas necessárias para o funcionamento do robô. A partir do momento em que há itens, rotinas e ferramentas que são parte de uma funcionalidade, desenvolvidas, passa a ser possível a junção destes para que essa funcionalidade tome forma. Com estas finalizadas, é realizada sua integração com outras funcionalidades, para que possa receber suas entradas e entregar suas saídas.

Deve-se também verificar o desempenho das funcionalidades, a validação do desenvolvimento é realizada com testes de operação dos sistemas. Nestes testes, são executados procedimentos que repliquem as situações de operação do robô. É importante também que as condições dessa operação, ambiente, carga e etc. sejam similares as quais o robô irá encontrar.

Após a realização de testes para a validação do desenvolvimento, é possível analisar os resultados dos mesmos. As informações levantadas nos testes mostram se as funcionalidades desenvolvidas realizam o que se espera e a sua eficiência. A partir daí, pontos do projeto a serem melhorados ficam evidentes, possibilitando assim a otimização de funções.

Capítulo Quatro

Desenvolvimento e testes

"Minha mãe sempre dizia: 'a vida é como uma caixa de chocolate. Você nunca sabe o que vai encontrar'."

(Forrest Gump)

Durante o desenvolvimento do projeto, foram feitos diversos estudos relacionados com o âmbito da robótica. A diversidade de ferramentas existentes faz com que o pesquisador tenha diversos formas de alcançar um resultado final e por se tratarem de conhecimentos específicos, cada ferramenta exige complexidade e dedicação para que seu domínio seja efetivo e resultados satisfatórios sejam alcançados.

A constante pesquisa e trabalho com as mesmas ferramentas definidas no início do processo metodológico possibilitou a realização de diversos testes, buscando validar as decisões tomadas, assim como aumentar o domínio das ferramentas e descobrir a melhor forma de utilizar todo seu potencial.

Os dispositivos eletrônicos oferecem muita informação sobre o seu funcionamento, porém, é necessário conhecer o seu comportamento na prática, já que esses dispositivos serão expostos a situações específicas do projeto, além de que, por estarem funcionando em conjunto com diversos subsistemas, é importante conhecer o seu comportamento quando como organismo de um sistema maior.

Com o decorrer do fluxo metodológico diversos tipos de resultados foram alcançados, seja conhecimento específico de uma ferramenta, validação da sua funcionalidade para um sistema específico ou seu comportamento como um organismo. O conceito do sistema é algo fundamental e o mesmo foi um fruto dessa metodologia, onde as suas funcionalidades desempenham um importante papel para se conhecer a idéia.

4.1 Análise das Funcionalidades

O desenvolvimento do conceito do sistema resultou nas funcionalidades, que por sua vez representam o fluxo das informações. A forma de estabelecer esse fluxo foi idealizando a operação do robô, buscando se determinar as atividades à serem desenvolvidas.

A inspeção de linha, exemplificada como o ato de caminhar na linha e ultrapassar um obstáculo, foi denominada missão. Assim, cada vez que o robô inicia esse processo, é iniciada uma missão.

De forma a garantir a execução da missão de forma efetiva, utilizou-se os requisitos fornecidos para o cliente e o *QFD* 1 aliado ao estudo das ferramentas para determinar os subsistemas presentes no robô, produzindo assim uma versão do *QFD* 2 aliada com a definição das funcionalidades para o sistema robótico, já que ambas aconteceram em paralelo e são complementares, o *QFD* 2 está mostrado na figura A.1 , estando disponível no anexo A.

Figura 4.1: *QFD* 2

Com uma idealização do fluxo do sistema e suas funcionalidades, foi elaborada uma arquitetura geral do sistema de movimentação, mostrada na figura B.1 e disponível no anexo B. Antes do início da missão, é necessário que o sistema esteja apto a tal, assim foi definida uma funcionalidade para verificação da integridade do sistema, que requisita

os status dos dispositivos antes de realizar a missão por meio de uma checagem, e com o sucesso na checagem, fornece o comando para o início.

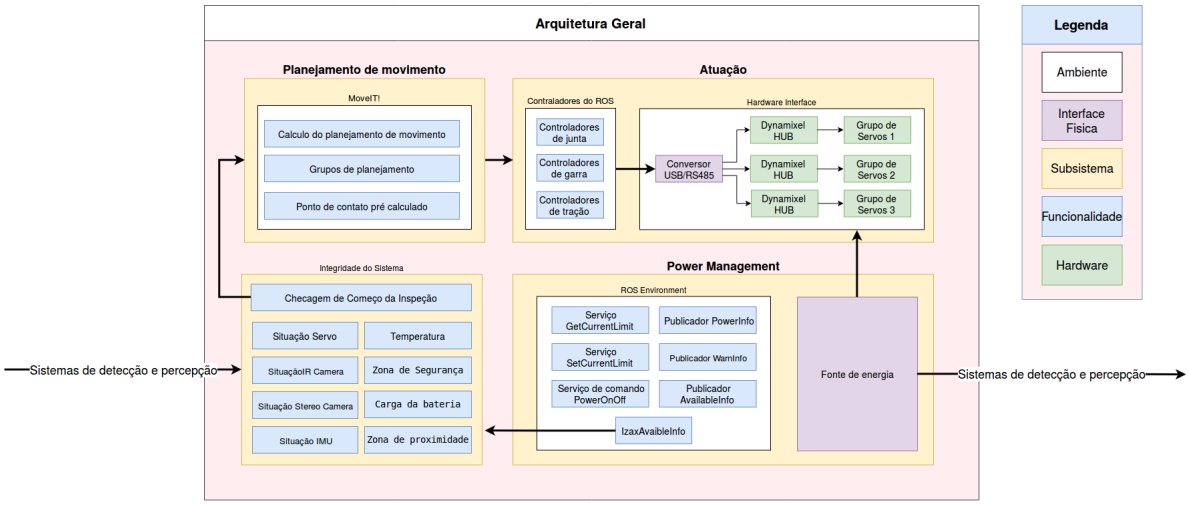


Figura 4.2: Arquitetura geral do sistema de movimentação

Determinou-se que a funcionalidade responsável pelo Planejamento de Movimento, realizasse o cálculo do plano de movimento junto com a administração dos grupos de controle e o cálculo de pontos de contato no robô. Onde essas informações sobre o plano de movimento do robô seriam transferidas para a funcionalidade de atuação do robô, que é responsável por receber esses comandos nas estruturas padrão do *ROS* e realizar o movimento físico do robô.

A movimentação exige muita energia do robô e aliado à necessidade do controle da potência disponível, foi idealizada a funcionalidade de Gerenciamento de energia, responsável por disponibilizar as informações de energia no ambiente *ROS* e fornecer a energia para o sistema de alimentação.

4.1.1 Atuação

Com a arquitetura geral do sistema de movimentação e o fluxo de comunicação, foi possível estabelecer as propriedades específicas da funcionalidade de atuação, onde buscou-se definir suas dependências, saídas e seu funcionamento.

Foi feito um fluxograma para representar a funcionalidade, onde estão ilustrados todos seus parâmetros, como mostra a figura 4.3.

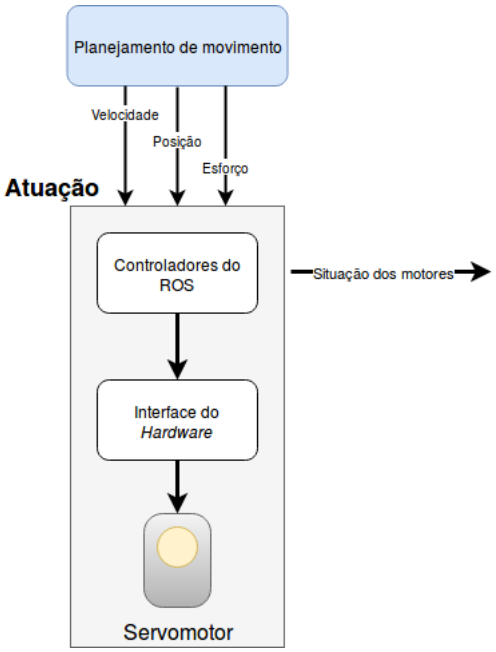


Figura 4.3: Fluxograma da funcionalidade de Atuação

Os comandos relativos a posição, esforço e velocidade para as juntas vêm da funcionalidade de planejamento de movimento, onde a atuação recebe esses comandos, transferindo os mesmos para os controladores do ambiente *ROS* que se comunicam com a interface de *hardware* que envia o comando para os motores realizarem o movimento.

4.1.2 Planejamento de Movimento

A forma que o robô vai realizar seus movimentos é determinado pela funcionalidade de Planejamento de Movimento. Que por sua vez tem a ferramenta *MoveIt!* como importante componente para que a movimentação ocorra de forma efetiva, sendo assim, as unidades de *software* da ferramenta que irão se comunicar com a funcionalidade explicitadas no fluxograma 4.4 estão incluídas no ambiente do *MoveIt!*, recebendo o comando da posição desejada para o *end-effector* e enviando os comandos necessários para as juntas.

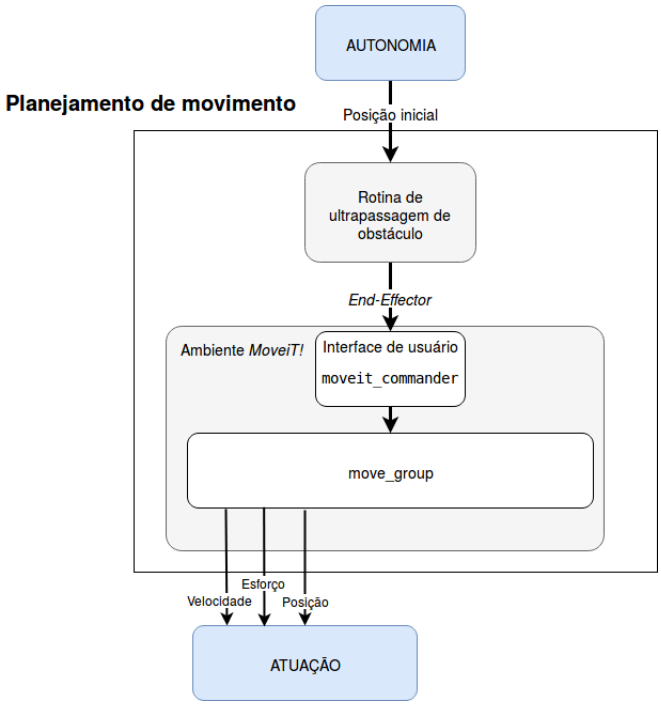


Figura 4.4: Fluxograma da funcionalidade de Planejamento de Movimento

Apresenta uma dependência do sistema de autonomia do robô, que envia uma posição de referência o robô, o que culmina na necessidade da ultrapassagem de um obstáculo, já que, em operação, a ultrapassagem de obstáculos acontece diversas vezes. A rotina de ultrapassagem de obstáculo então envia para o *software* o *end-effector* necessário para a ultrapassagem, e consequentemente são enviados os comandos contendo a posição, esforço e velocidades das juntas para a funcionalidade de atuação.

4.1.3 Gerenciamento de Energia

A necessidade de distribuir a energia elétrica proveniente de uma bateria entre os sistemas eletrônicos do robô, de maneira inteligente se mostrou uma parte fundamental do projeto. Para que isso fosse implementado, houve a necessidade de desenvolver uma funcionalidade responsável por monitorar os níveis de corrente e tensão fornecidos para cada um dos dispositivos eletrônicos do robô, permitindo a criação de um sistema de proteção contra surtos de corrente, com ajuste de limites para valores do fornecimento de corrente.

Após a análise e constatação da necessidade da implementação desta funcionalidade, foram feitas as definições de quais seriam suas entradas, saídas e suas dependências.

A funcionalidade de gerenciamento de energia depende essencialmente da placa mul-

tiplexadora, a qual é responsável por alimentar o *hardware* de *Power Management* com a energia proveniente das baterias, e de que todos os seus drivers e funções estejam instaladas no ambiente *ROS*, para que seja possível utilizar as suas ferramentas.

O sistema de *power management* cria no ambiente *ROS*, estruturas de *software* responsáveis por permitir o monitoramento das portas de alimentação (informando valores de tensão e corrente), bem como a configuração dos limites do fornecimento de corrente e a desativação ou acionamento dos relés digitais de cada uma das portas.

O gerenciamento de energia monitora os valores de corrente fornecidos nas portas, e verifica ocorrência de possíveis surtos, verificando o valor do pico e o tempo de duração. Caso seja detectado um surto de corrente, o fornecimento da porta é desativado, protegendo assim o sistema como um todo.

Uma vez que o estudo do *hardware* e das suas funções no ambientes *ROS* estavam completos, foram iniciados os primeiros testes com partes isoladas do robô para verificar o comportamento do *hardware* e se o seu funcionamento estava de acordo com o esperado, para atender os requisitos da funcionalidade de maneira satisfatória, para assim ser integrado ao sistema.

4.1.4 Checagem da Integridade do Sistema

Todas os sistemas do robô devem operar de maneira correta, para garantir o sucesso na execução da missão, e a partir desta observação, foi concebida a funcionalidade de checagem de integridade do sistema. Sua principal função é garantir que todas os sistemas do robô estejam sem apresentar falhas antes do início da missão. Uma vez que a checagem é realizada, o robô pode iniciar a sua missão.

Esta funcionalidade se mostra importante pelo fato de que a checagem prévia, reduz significantemente os riscos e prejuízos atrelados a uma má execução da missão, como componentes danificados e perda na eficiência durante toda a missão. Todos os sistemas que irão participar da missão, devem estar atrelados ao sistema por meio do ambiente *ROS*, onde cada um irá informar o seu estado para a funcionalidade. O fluxograma 4.5 foi desenvolvido para ilustrar o funcionamento da rotina de checagem integridade do sistema.

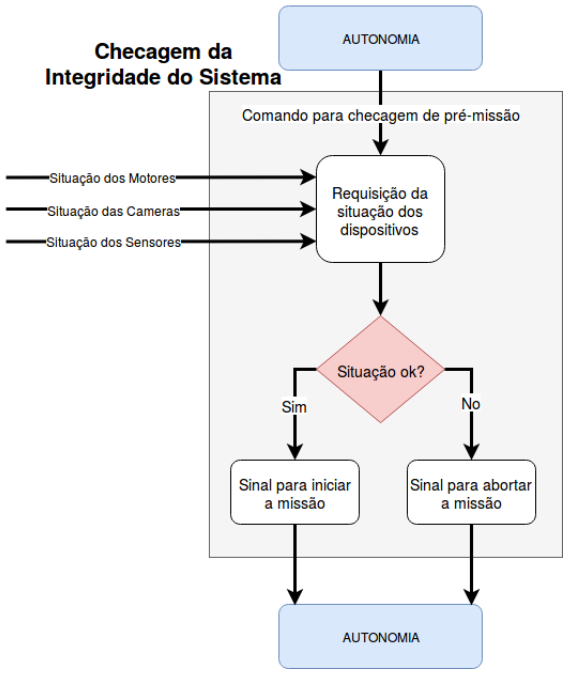


Figura 4.5: Fluxograma da funcionalidade de Checagem da Integridade do Sistema

Uma vez que esta funcionalidade foi concebida, toda as outras integrantes do sistema, deveriam se integrar ao sistema de integridade para garantir que a missão pudesse ser iniciada sem erros graves.

4.2 Estudo da Movimentação

Para o desenvolvimento das ferramentas para ultrapassagem de obstáculos, foi necessária a análise dos movimentos necessários para que fosse realizada a ultrapassagem de obstáculos. Os tipos de movimento a serem realizados influenciam nas ferramentas escolhidas para o controle e operação do robô, já que se busca a otimização do movimento e também redução do tempo gasto no desenvolvimento, de forma a garantir um resultado satisfatório.

Foi feita uma análise utilizando como referência o maior obstáculo, que foi o amortecedor, mostrado na figura 4.6 e levando em consideração a sua modelagem como um paralelepípedo de maior dimensão 535mm. Constatou-se que seria necessário que para a ultrapassagem desse obstáculo, o robô necessitaria abrir um dos braços e se deslocar somente com a unidade de tração central e outro braço. Para isso seriam necessários movimentos para abrir e fechar as garras do robô, assim como um comando específico para afastar a unidade de tração da linha a distância necessária para a abertura da garra. Assim como o movimento que abre o braço suficiente para que esse passe abaixo do obstáculo.

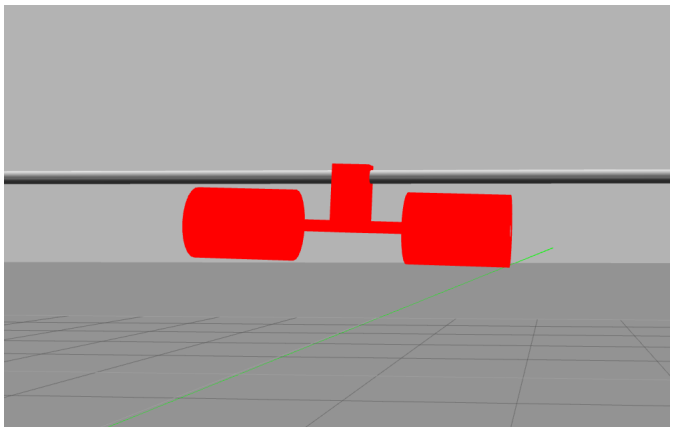


Figura 4.6: Visualização do obstáculo no simulador

A ferramenta de visualização presente no *ROS*, possibilitou testar os limites de giros das juntas e visualizar como seriam as poses do robô sem a necessidade da simulação. O conhecimento relacionado à como a movimentação ia ocorrer possibilitou nortear o desenvolvimento.

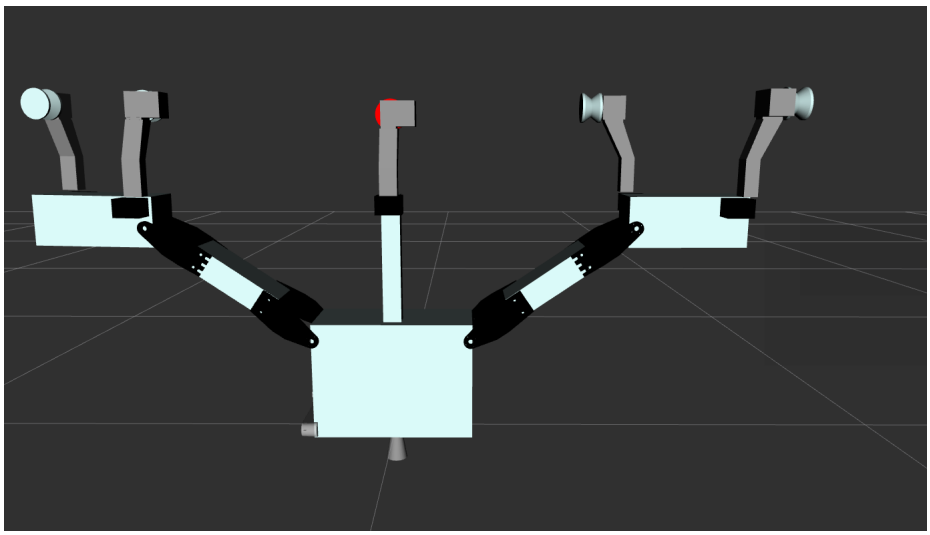


Figura 4.7: Robo *ELIR* no visualizador do *ROS* com garras abertas

4.3 Soluções Mecatrônicas para o sistema robótico

Uma vez que os principais aspectos das funcionalidades estavam definidos, foi necessário realizar a implementação das soluções mecatrônicas no robô. Foram realizados estudos de dispositivos eletrônicos e das suas ferramentas de configuração, bem como programas e *softwares* que seriam necessários para a implementação das funcionalidades.

4.3.1 Escolha a biblioteca de controladores no ROS

Para implementar as rotinas de controle e atuação dos motores Dynamixel, era necessário que houvesse algum meio de configurá-lo para operar de maneira desejada, podendo realizar a criação de juntas, e unidades de tração. Especificamente nos motores Dynamixel, isto é possível através da utilização de *drivers*.

Os *drivers* são estruturas de *software* que permitem a compatibilização do dispositivo físico com uma interface computacional, habilitando as suas funções e configurações dentro de um ambiente de *software*, permitindo que o dispositivo possa ser configurado e ter suas funções utilizada através de comandos provenientes de uma camada de *software* superior.

A movimentação das juntas e eixos do robô se dá pelo uso de servomotores, e para que os mesmos sejam controlados pelo *framework*, que no caso é o *ROS* - é necessário fazer uso de um *driver*.

Diante a variedade de bibliotecas para controlar os motores da dynamixel no *ROS* disponíveis, foi necessário escolher qual será utilizada. Para escolha a biblioteca foram levados alguns parâmetros que influenciam para o fluxo de informações e funcionalidades do robô, assim como complexidade de código, disponibilidade de tutoriais, suporte pela comunidade, as ferramentas disponíveis, compatibilidade do *firmware*, etc.

Foram selecionadas 2 bibliotecas que são disponibilizadas pelo próprio *framework ROS*, sendo elas: a *dynamixel_drivers* e a *dynamixel_workbench*. A escolha pela *dynamixel_drivers* se deu pelo fato ser mais velha e por consequência mais materiais disponíveis para consulta, apesar a *workbench* possuir suporte para novas versões do Dynamixel, essa possibilidade não teve muitos benefícios, já que o projeto foi utilizado para versões antigas de motores.

4.3.2 Solução para cinemática

O cálculo da cinemática pode ser realizado de diferentes formas, já que o mesmo parte da modelagem do robô por meio de um conjunto de equações. Existem diferentes solucionadores para problemas de cinemática, que podem utilizar diferentes parâmetros de entrada.

No âmbito da robótica podem ser utilizados tanto os modelos matemáticos já elaborados pelos desenvolvedores, como também o modelo 3D do robô. Esse modelo 3D

é definido geralmente por meio de arquivos do tipo *URDF* , que utilizam uma escrita simples, definindo o robô a partir de seus *links* e juntas. Esse arquivo é o mesmo utilizado para colocar o modelo do robô na simulação, e também é a forma padrão utilizada pelo *software MoveIt!*. Com o estudo do estado da arte fornecido, não era necessário encontrar o conjunto de equações que descrevem o robô, e com a simulação fornecida, também já era possível utilizar o modelo *URDF*. Foi feita uma análise entre as possíveis soluções cinemáticas, buscando encontrar a que garantisse um maior sucesso e facilitasse o desenvolvimento. Levando em consideração a disponibilidade de códigos e tutoriais para serem tomados como base, assim como o uso em projetos semelhantes, a complexidade da aplicação e integração com as estruturas do *ROS*.

Assim levantou-se 3 possível soluções para o cálculo da cinemática inversa, sendo elas: o uso do modelo matemático já existente proveniente do estudo do estado da arte, sendo realizada sua solução manualmente por meio de códigos de programação, o uso da ferramenta *MoveIt!* utilizando o modelo *URDF* proveniente da simulação, e o uso da ferramenta *MoveIt!* compatibilizada com modelo matemático.

A uso do modelo matemático completo para o robô, realizando a solução por meio de um código, se mostrou inviável, já que também seria necessário a integração com as estruturas de controle do *ROS*, e mesmo a implementação desse modelo no *MoveIt!* não se mostrou a melhor opção, devido a alta complexidade e a falta de projetos semelhantes para consulta, tal qual a inexistência de tutoriais voltadas para essa aplicação em específico. Assim sendo escolhido o uso da ferramenta *MoveIt!* com o modelo *URDF* proveniente da simulação, já que esse mesmo modelo já é estudado pela equipe, para seu uso na simulação e a aplicação padrão do *MoveIt!* utiliza esse modelo.

Essa solução utiliza os dados sobre o modelo do robô contidos no modelo *URDF* para realizar o cálculo da cinemática de forma analítica, gerando diversas trajetórias e encontrando a melhor possível, baseada nos parâmetros configurados pelo usuário.

4.3.3 Gerenciamento de Energia

A implementação da funcionalidade de gerenciamento de energia foi possível através da utilização de um *hardware* de *Power Management*, uma placa eletrônica, fornecida pelo cliente do projeto, iniciando assim a primeira etapa do desenvolvimento da funcionalidade de gerenciamento de energia, a qual se deu pelo estudo do *hardware*, através da leitura de *datasheets* e esquemas elétricos.

A placa contém sete portas de alimentação gerenciadas por relés digitais sendo cinco delas de 12 Volts e duas de 5 Volts. Dentre as portas de 12 Volts, 4 possuem

regulação de fornecimento de corrente, onde o controle do fornecimento é realizado por um microcontrolador Atmega32U4. Todas as portas possuem sensores de tensão e corrente individuais, facilitando e otimizando o processo de leitura dos . O *firmware* embarcado na placa de *Power Management* possui a principal de função de implementar os relés digitais, onde se realiza uma leitura dos valores de corrente que estão sendo demandados pelas portas.

Para que a integração do *hardware* de *Power Management* fosse realizada ao projeto foi necessário validar o seu funcionamento , e se os valores de tensão fornecidos, estariam de acordo com o configurado. As medições lidas no multímetro e no *software*, foram registradas e comparadas, verificando se os valores de erro eram muito altos. Como mostra a Tabela 4.1 a seguir:

Tabela 4.1: Medições de tensão para o *hardware* de *Power Management*

Porta Lida	Tensão - Multímetro	Tensão - *Software*
Porta 1	12.03 Volts	12.043 Volts
Porta 2	12.01 Volts	12.037 Volts
Porta 3	12.01 Volts	12.038 Volts
Porta 4	12.01 Volts	12.037 Volts

Foi possível perceber uma pequena variação na leitura em *software* para a leitura realizada pelo instrumento de medição, apesar disso, pelo fato do erro ser de uma ordem baixa para os níveis de operação do robô, considerou-se que esta variação não representaria risco para a operação, validando assim o funcionamento do *hardware* da *Power Management*.

4.4 Simulação

A estrutura do sistema *ROS* e suas ferramentas já compatibilizada faz com que a simulação do sistema impacte de forma efetiva e positiva durante o processo de desenvolvimento. O *software* Gazebo é uma das ferramentas para simulação de ambientes robóticos mais utilizadas atualmente, por apresentar integração nativa com o sistema *ROS* e foi utilizada para o projeto.

Os arquivos para simulação do robô foram fornecidos pelo cliente, e apresentam características semelhantes às esperadas no robô real, como o tipo dos controladores de juntas, que são o padrão do *ROS*, sendo os mesmos utilizados para o robô real, e consistindo assim um parâmetro fundamental para a simulação do sistema de movimentação.

Por apresentar os mesmos tipos de controladores do robô real, foi possível utilizar a simulação para validar a compatibilização do controle da ferramenta *MoveIt!* com os controladores presentes no robô. Essa alternativa possibilitou a compatibilização entre as funcionalidades sem a necessidade do protótipo do robô pronto funcional, economizando tempo do desenvolvimento, já que reduz dependência entre as fases do desenvolvimento.

4.4.1 Simulação para a rotina de ultrapassagem

A automatização dos movimentos do sistema é necessária para que ocorra a ultrapassagem, sendo assim esse conjunto de movimentos denominado de rotina. As estruturas do *ROS* possibilitam a criação de estruturas que quando acionadas realizam comandos específicos.

Por apresentar os mesmo controladores, e com o auxílio do estudo dos movimentos necessários para a ultrapassagem, foi possível desenvolver os códigos que enviam os comandos específicos para os controladores.

Com uso da simulação foi possível desenvolver e testar os serviços do *ROS*, na figura 4.8 é possível observar o serviço de suspender o conjunto de garras em funcionamento:

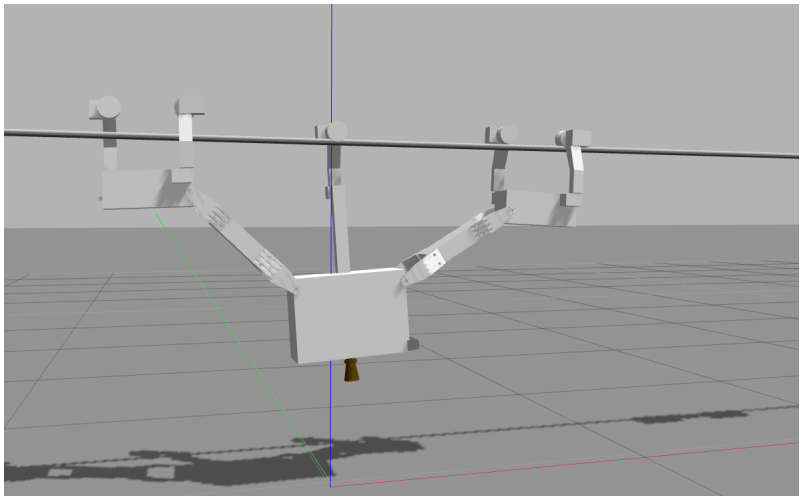


Figura 4.8: Robo *ELIR* no *Gazebo* com garras abertas

4.5 Testes de Movimentação Física

Os testes de movimentação física buscam compreender e validar os movimentos realizados pelo robô, indo desde da análise individual das partes do robô como braços e unidade de tração, até sua movimentação em conjunto.

4.5.1 Testes unitários

Os primeiros testes foram feitos com a estrutura parcialmente montada, já que o objetivo do teste é validar o funcionamento dos controladores de junta do *ROS*, que é a base para o desenvolvimento das funcionalidades de Atuação e Planejamento da Movimentação. Nestes testes, foram estabelecidos os limites de movimentação das juntas, com o intuito de preservar a integridade física dos componentes. As juntas dos braços do robô *ELIR* são compostas por dois servomotores Dynamixel MX-106R fabricados pela Robotis.

No decorrer desses primeiros testes, ocorreram falhas nos motores por excesso de carga em um deles, o que era configurado como mestre. Verificou-se que isso poderia ser contornado com o uso do cabo de sincronização ligando os dois motores de cada junta, já que os motores se comunicariam diretamente entre si para balancear a divisão da carga sobre a junta. Também foi o momento de analisar como se comportavam os servomotores utilizados nas juntas, sob influência do cabo de sincronização.

Tabela 4.2: Sentido de giro dos motores (Cabo de sincronização direto)

Valor pulicado na junta:	Positivo	Negativo
Mestre	Horário	Anti-horário
Escravo	Anti-horário	Horário

Tabela 4.3: Sentido de giro dos motores (Cabo de sincronização cruzado)

Valor pulicado na junta:	Positivo	Negativo
Mestre	Horário	Anti-horário
Escravo	Horário	Anti-horário

Levando em conta que, para realizar o movimento corretamente, os motores tem que se movimentar em sentidos opostos, já que estão fisicamente invertidos, caso seja utilizado o cabo de sincronização cruzado, os motores escravos devem ser configurados com o modo de giro reverso.

4.5.2 Testes na linha de transmissão

O teste no ambiente onde o robô irá operar é de suma importância para o seu desenvolvimento. Tendo como finalidade verificar o comportamento do robô e seus sistemas nessa situação, foram realizados testes de movimentação na linha de transmissão buscando executar etapas que fazem parte da rotina de ultrapassagem. Para verificar o

deslocamento horizontal por meio de roldanas, realizou-se um testes posicionando robô na linha e controlando somente as unidades de tração das roldanas, verificando assim que elas poderiam se soltar da linha. Dessa maneira, concluiu-se que as garras onde estão as roldanas deveriam ter o seu torque habilitado para travá-la na posição, para que o robô não se desprenda da linha.

Durante os testes de movimentação na linha, foram testadas outras etapas da rotina de ultrapassagem. Quando o braço era levantado para retirar as roldanas da linha, todo o robô se inclinava para a frente. Levantou-se a hipótese de que, o fato das juntas dos braços estarem "relaxadas"permitia essa inclinação. O teste foi repetido com o torque destas juntas habilitado, dessa forma o problema foi contornado.

4.6 Integração com os subsistemas

Após a validação do funcionamento dos subsistemas, verificando se estão operando corretamente através dos testes unitários, foi necessário conferir o funcionamento do sistema como um todo, analisando a integração das as funcionalidades, com o objetivo de observar se houveram erros no processo de integração. Os testes buscavam seguir com integração simples entre funcionalidades, até os testes finais, envolvendo todas as funcionalidades desenvolvidas durante o projeto.

4.6.1 Teste dos servomotores em rede

Como o robô *ELIR* possui ao total 18 servomotores Dynamixel, então torna-se importante verificar o seu comportamento quando todos estão pertencendo a mesma malha de controle. Para isso foi utilizado *hubs* para a separação em 3 grupos, contendo 6 motores por grupo. Para efetuar o teste foi utilizado a biblioteca dos controladores do *ROS* que indica quantos motores estão conectados a unidade de processamento.

Cada grupo de motor foi testado separadamente e todos os 3 foram encontrados, concluindo que as ligações, alimentação e comunicação foram feitas corretamente. Mas ao colocar os 3 grupos de motores, percebeu-se que nem todos os motores eram encontrados.

Buscou-se reduzir as fontes de erro, realizando uma verificação na alimentação e nas ligações responsáveis pela comunicação entre os motores, o que não foi eficiente, assim como a restauração do *firmware* de cada um. Como não haviam mais fontes de erro, buscou-se encontrar motores que poderiam estar induzindo esse problema no sistema. Assim, foi pensado em adicionar motor por motor para observar o comportamento e

a partir disso analisou-se que somente 2 motores específicos causavam falhas na rede, constando que os mesmo estavam com problemas em comunicação em rede e que vieram com defeitos de fábrica. Logo após a troca dos motores defeituosos a comunicação ocorreu de forma coerente e constatou-se que os 18 motores conseguem se comunicar em redes grandes.

4.6.2 Teste de movimentação alimentada por banco de baterias

Para que o robô *ELIR* desempenhasse suas funções de forma autônoma era necessário o uso de baterias que suportasse toda demanda energia do mesmo. Afim de saber quantas baterias iriam ser usadas dentro da estrutura do robô, foi feito uma estimativa a partir

Durante todo desenvolvimento do projeto foi utilizado fontes de bancada onde a tensão era de 14V onde os motores desempenham maior torque Antes de utilizar as baterias para que o robô pudesse realizar os movimentos na linha, tornou-se importante realizar testes para garantir que a mesma, juntamente com a placa de gerenciamento de energia, fornecem potência suficiente com tensão a 12V para que os motores realizem sua movimentação. Com isso, era interessante verificar por quanto tempo os motores aguentariam na posição que demanda maior torque no qual possui maior carga devido ao momento aplicado ao braço, como mostrada na figura 4.9 a seguir:

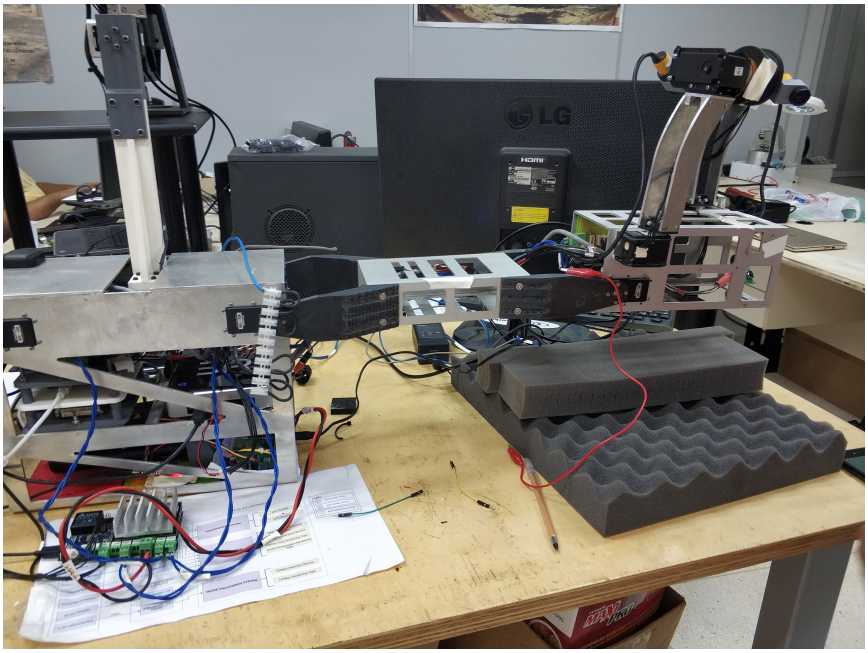


Figura 4.9: Foto do robô na posição que demanda maior torque

Verificou-se que o robô respondeu bem com as baterias ficando nesta posição durante 10 minutos por 3 vezes, validando o seu uso e comprovando que os motores a 12V

fornecidos pela placa realizam torque suficiente para a carga da estrutura.

4.7 Análise Preliminar

Algumas situações só são expostas durante o desenvolvimento do projeto, com a aplicação das ferramentas e a vivência de testes. O desenvolvimento do projeto passou por diversas etapas, onde fora necessário realizar o retrabalho de novos estudos sobre as problema de compatibilização do modelo do robô dentro da plataforma *MoveIt!*, uma vez que o *software* não funciona bem com robôs que possuem apenas dois graus de liberdade, o que é o caso do robô *ELIR*. A observação desse erro é fundamental para o processo de aprendizado no que tange ao desenvolvimento de robótica, tornando o *MoveIt!* uma solução não indicada para o problema da solução da cinemática inversa, porém ainda se mostra muito útil para o planejamento do movimento e integração com outras tecnologias, assim compensando a solução da cinemática manualmente, para projetos de menor complexidade.

Em termos de hardware, os erros que surgiram também foram importantes para ilustrar melhor como os mesmos deveriam ser configurados de maneira correta e também as suas limitações de trabalho, a exemplo dos motores *Dynamixel*, estes mostraram um desempenho mais efetivo ao trabalhar com tensões mais elevadas, 14V ao invés dos tradicionais 12V, diminuindo a corrente demandada durante a operação, otimizando o consumo.

Ainda se tratando do uso dos motores, a necessidade do uso dos cabos de sincronização para evitar erros de sobrecarga foi um fator crucial para o desenvolvimento do projeto. Neste ponto observou-se que os motores, ao trabalharem como *dual-joint* (junta com dois motores), necessitam do uso de cabos de sincronização para transmitir a informação de carga necessária para realização do movimento, balanceando o esforço realizado pelos motores trabalhando em conjunto.

Capítulo Cinco

Conclusão

"Se vi mais longe foi por estar de pé sobre ombros de gigantes."

(Sir Isaac Newton)

Neste trabalho foi desenvolvido o sistema de movimentação para um protótipo de robô de inspeção de linha. O fluxo metodológico foi uma grande conquista desse projeto, onde o se buscou seguir as diretrizes de desenvolvimento de projetos robóticos de alto porte, utilizando ferramentas de referência para estabelecer um conceito concreto, e também produzindo diversos tipos de documentação que possibilitaram a produção de um conteúdo palpável em relação a esse trabalho.

O desenvolvimento de aplicações que foi realizado para o projeto, consiste em um conteúdo de suma importância para desenvolvimento de projetos similares, buscou-se utilizar as boas práticas de organização e implementar os padrões utilizados para outros projetos de desenvolvimento.

Inicialmente o projeto consistia em uma fuga da zona de conforto, apresentando diversos conteúdos novos e solicitando uma nova visão sobre o trabalho com engenharia, e com o seu caminhar, saiu da concepção da idéia e desafio do aprendizado para o desenvolvimento de um sistema robótico real.

O conceito gerado para o sistema, utiliza parâmetros de operação semelhantes à sistemas de alta complexidade, apresentando um fluxo de informações de alto nível, possibilitando uma noção sobre o funcionamento do sistema e integração com os outros subsistemas já desenvolvidos.

A organização do sistema, junto com suas ferramentas e conhecimentos produzidos, representa uma grande conquista do projeto. O conhecimento deixado relacionado ao *framework ROS*, as conclusões tiradas da ferramenta *MoveIt!*,assim como o comportamento observado pelos dispositivos físicos faz com que haja uma referência para projetos semelhantes, possibilitando a criação de conceitos cada vez mais consistentes e execução de idéias de forma efetiva.

A estrutura do protótipo montada, com suas ligações para comunicação e alimentação feitas devidamente, assim como esquemáticos desenvolvidos especialmente para o projeto,

possibilita que o mesmo seja tomado como base de estudo, ao se realizar o design e desenvolvimento de outros protótipos.

O uso da estrutura do *ROS* possibilita que seja feita a integração com outros sistemas sem muitos problemas, já que foram adotados os padrões do *framework*, assim possibilitando que o que foi desenvolvido para esse robô, seja utilizado em futuras aplicações, sejam elas tomando como base esse protótipo ou outros semelhantes.

A documentação técnica produzida seguiu de acordo com as orientações de projeto visando o reaproveitamento e continuação do trabalho, atendendo as demandas e solicitações para o desenvolvimento do projeto.

O projeto alcançou as expectativas, problemas inesperados encontrados durante o seu desenvolvimento foram contornados e os grandes desafios do projeto foram concluídos, onde a experiência como um todo foi muito enriquecedora e se deu de forma prazerosa, onde todos os participantes conseguiram aprimorar suas habilidades, crescer como pessoa e como profissional. Pelas palavras do sábio Platão: "A coisa mais indispensável a um homem é reconhecer o uso que deve fazer do seu próprio conhecimento."

Apêndice A

QFD

Correlations
Positive +
Negative −
No Correlation

Relationships
Strong ●
Moderate ○
Weak ▽

Direction of Improvement
Maximize ▲
Target ◇
Minimize ▼

Row #	Weight Chart	Relative Weight	Technical Importance	Maximum Relationship	Technical Requirements \ Functionalities	ACQUISITON	LOCALIZATION	DETECTION	USER INTERFACE
					Column #	1	2	3	4
					Direction of Improvement	◇	▲	◇	◇
1		9%	**762**	9	Confiabilidade	●	●	●	▽
2		3%	219	9	Monitoramento de consumo energético	●	○		
3		2%	196	3	Sistema regulador de tensão para acoplar à bateria	○	○		
4		4%	373	9	Georreferenciamento		●	○	○
5		5%	445	9	Interface leve e simplista, com poucos elementos visuais [UI]	▽			●
6		6%	**491**	9	câmera espectro visível	●		○	●
7		4%	369	9	câmera IR	●		●	●
8		5%	385	9	resolução das câmeras	●		●	●
9		3%	274	9	luminosidade			●	
10		4%	312		massa do robot				
11		5%	450	9	Armazenamento de dados da missão	●	○	●	●
12		5%	390	9	desempenho de deslocamento			●	
13		4%	332		velocidade de deslocamento				
14		2%	208	9	Classificação de obstáculos	●		●	○
15		3%	235	9	detectar e ultrapassar obstáculos	▽		●	○
16		5%	418	3	autonomia de potência	○	○	○	
17		2%	132	9	medição da cota do cabo ao obstáculo ao chão	●		●	○
18		6%	**508**	9	registro de ocorrências	●	●	●	●
19		2%	204	9	Exibir obstáculos em visada direta	○		●	●
20		2%	169	9	Exibir distâncias dos obstáculos	○		●	●
21		4%	377	9	garantir e monitorar o posicionamento das garras na linha de transmissão	●			
22		6%	**515**	9	Sistema operacional + Framework	○	○	○	●
23		9%	**738**	9	Back-end	○	▽	●	●
24		0%	**0**						
					Target	realizar a comunicação e a aquisição dos dados	informar o posicionamento e orientação do sistema quando solicitado	identificar pontos quentes e objetos na linha e na faixa de servidão	disponibilizar de forma simplificada os dados mais relevantes
					Max Relationship	9	9	9	9
					Functional Importance Rating	499,95	246,03	574,03	494,9
					Relative Weight	28%	14%	32%	27%
					Weight Chart				
					Column #	1	2	3	4

Figura A.1: *QFD* 2

Apêndice B

Arquitetura

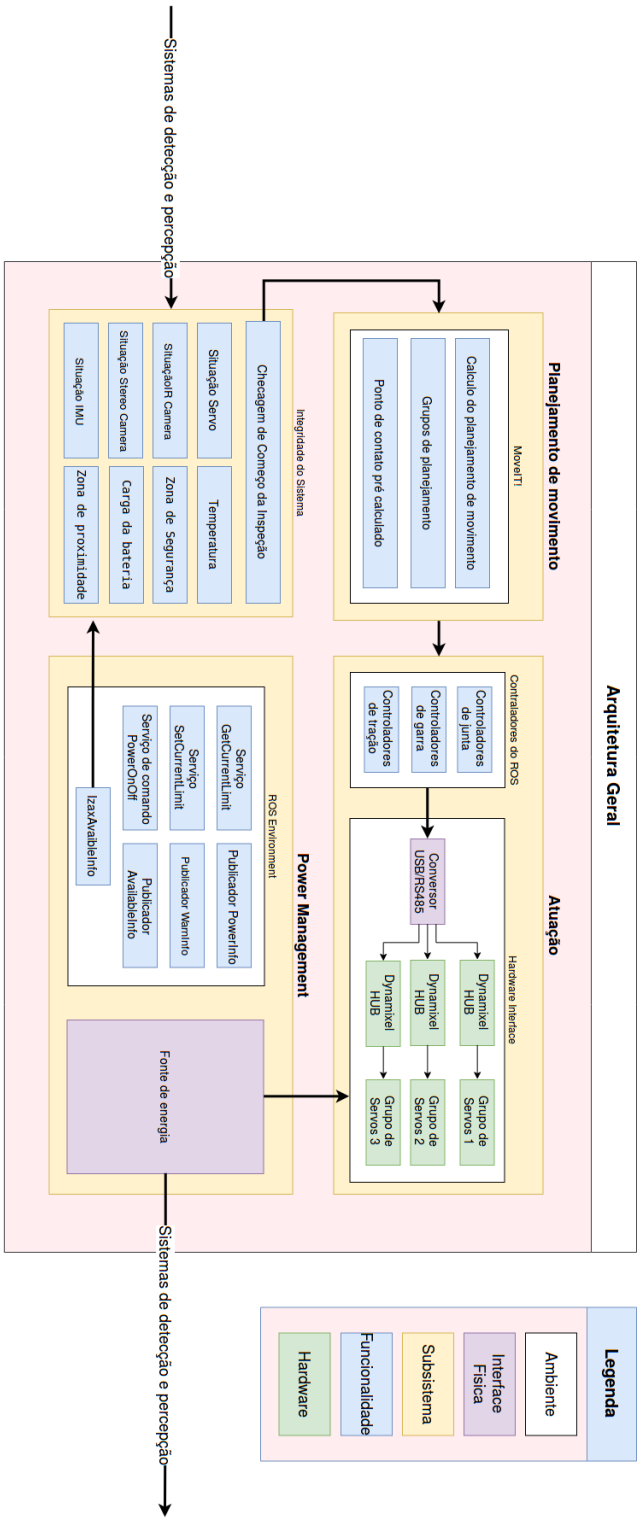


Figura B.1: Arquitetura geral do sistema de movimentação

Apêndice C

Logbook

ELIR

logbook

CONFIGURAÇÃO DOS LIMITES DE GIRO DOS MOTORES

Objetivos

O teste teve como objetivo estabelecer os limites de giro dos motores em seus controladores, com base nos limites físicos da estrutura do robô.

Descrição do teste

É criado um "Controller manager" que conecta os motores e publica em um tópico as informações destes. As juntas do robô são posicionadas manualmente em suas posições maximas e minimas, então o tópico "motor_states" é monitorado para verificar as posições dos motores.

DATA

8 AGOSTO 2018

LOCALIDADE

SENAI CIMATEC
SALVADOR - BAHIA

Mandruvah team

Cleber
Carlos
Ícaro
Davi

17:00

Foram coletados os limites de giro dos motores com id 11, 12, 13, 21, 22 e 23.

17:05

Ajustamos as posições iniciais dos controladores das juntas com base na posição *"home"* da simulação no *MoveIt!*. Nesse momento, verificamos que o valor que é publicado no controlador para mover a junta é a posição em radianos em relação à posição inicial que foi determinada no controlador.

ELIR

logbook

TESTE DE MOVIMENTAÇÃO DOS SERVO-MOTORES

Objetivos

Verificar o comportamento do robô executando alguns movimentos em um dos braços.

Descrição do teste

Os motores são alimentados e seus controladores executados. A partir daí, valores de posição são publicados e o comportamento do robô verificado.

DATA

10 AGOSTO 2018

LOCALIDADE

SENAI CIMATEC
SALVADOR - BAHIA

Mandruvah team

Cleber
Carlos
Ícaro
Davi

15:20

O braço do robo foi levantado até a posição *"home"* com as duas juntas sendo movimentadas ao mesmo tempo. Antes de atingir a posição determinada, o motor com id 21 apresentava erro de overload.

16:10

Quando o robô começa o movimento já proximo da posição final, *"home"*, o braço consegue alcançar o objetivo. Depois de cerca de 5 minutos nessa posição, um erro de overheat é apresentado.

ELIR

logbook

TESTE DE MOVIMENTAÇÃO DO BRAÇO

Objetivos

Verificar possíveis motivos para erro de "Overload" na junta 12-22 apresentado em teste anterior.

Descrição do teste

Os motores são alimentados e seus controladores executados. A partir daí, valores de posição são publicados e o comportamento do robô verificado.

DATA

13 AGOSTO 2018

LOCALIDADE

SENAI CIMATEC
SALVADOR - BAHIA

Mandruvah team

Cleber
Carlos
Ícaro
Davi

16:50

Foi verificado que os motores estão configurados para operar com 100% do torque, não sendo assim essa a causa do problema.

17:15

Percebemos também que o problema acontece com maior frequência quando as juntas do braço e da unidade de tração são acionadas ao mesmo tempo. Quando é acionada uma junta por vez, o "Overload" acontece menos vezes.

17:32

É levantada a suspeita de que a falta do cabo de sincronização nos motores da junta pode ser a causa da falha. Com o cabo conectado, o problema não aconteceu.

ELIR

logbook

TESTE DE CONVERSOR DA PLACA DE POWER MANAGEMENT

Objetivos

Verificar possíveis problemas do conversor da placa de power management e sua resposta de saída .

Descrição do teste

O conversor é retirado da placa e então testado com fonte de alimentação e sua saída observada com um multímetro. O conversor testado é do modelo UWE-12/10-Q12PB-C

DATA

25 SETEMBRO 2018

LOCALIDADE

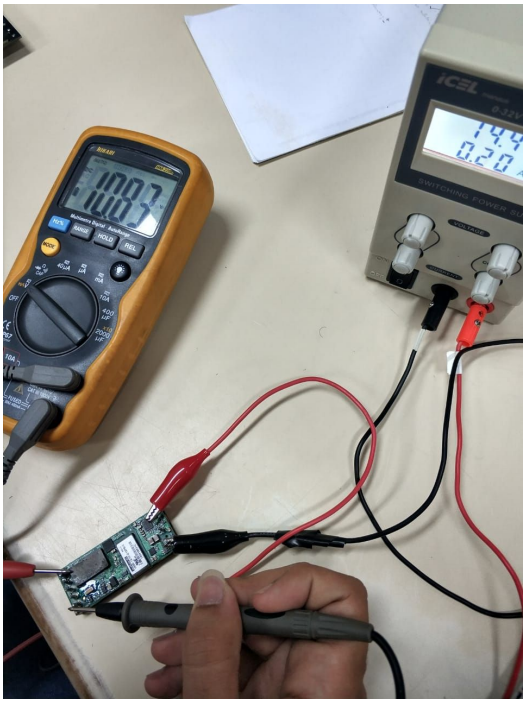
SENAI CIMATEC
SALVADOR - BAHIA

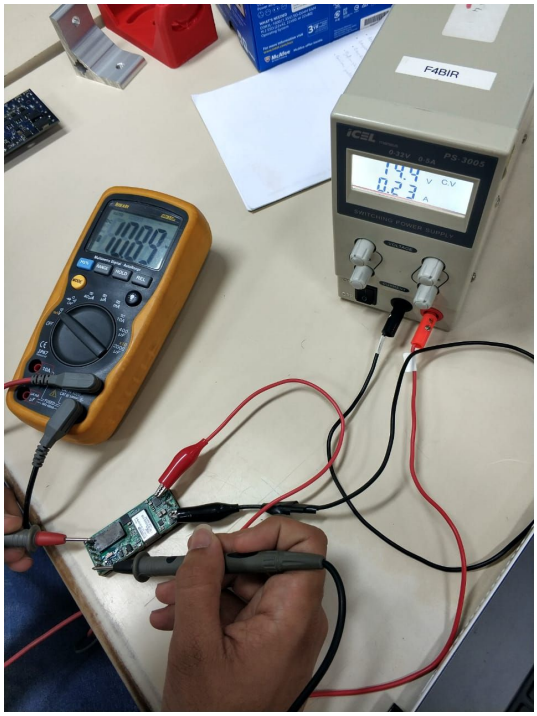
Mandruvah team

Cleber
Carlos
Ícaro
Davi

16:05

O conversor retirado da placa foi testado com alimentação de uma fonte de tensão com 14 Volts, e o mesmo consumiu um valor de cerca de 200 mA. Sua tensão de saída ficou em aproximadamente 10 Volts.



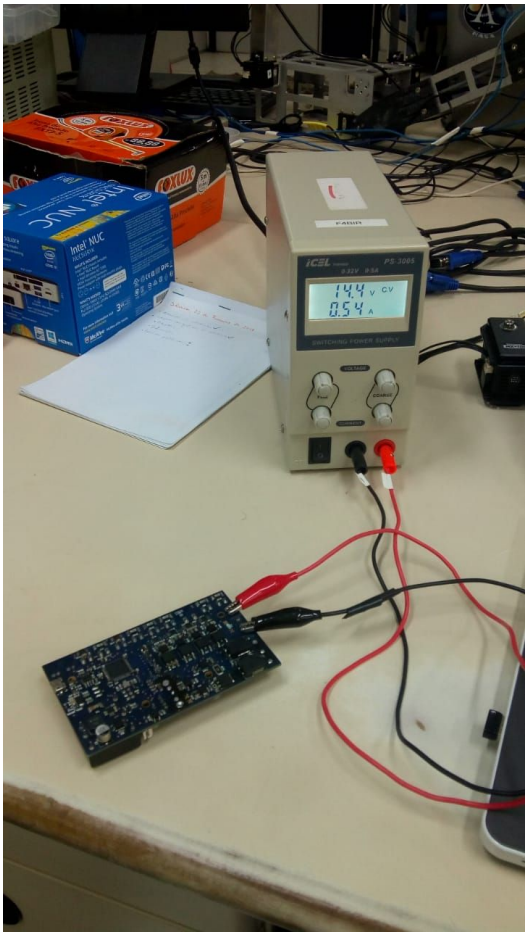


16:12

A corrente requisitada pelo conversor defeituoso oscila entre 200 mA e 250 mA. Sua temperatura, ao contrário do conversor que está funcionando corretamente, não aumenta e o conversor permanece frio.

16:28

O conversor que já está na Power Managemente permanece apresentando funcionamento correto. Sua temperatura aumenta quando permanece ligado.



ELIR

logbook

TESTE DOS MOTORES/CONTROLLER_MANAGER

Objetivos

Identificar se há algum motor defeituoso que pode estar "sujando" a comunicação dos motores.

Descrição do teste

Um motor é conectado e o arquivo controller_manager.launch é executado e verifica-se se o motor foi encontrado. Em seguida, são inseridos os demais motores, um a um, para que se perceba se a comunicação ainda acontece.

DATA

04 OUTUBRO 2018

LOCALIDADE

SENAI CIMATEC
SALVADOR - BAHIA

Mandruvah team

Cleber
Carlos
Ícaro
Davi

13:40

Todos os 18 motores estavam conectados, utilizando apenas os componentes (cabos e hub) da ROBOTIS. Quando o controller_manager.launch foi executado, nenhum motor foi encontrado.

13:42

Com apenas um motor conectado, o controller_manager o encontrou.

13:55

O teste prosseguiu até que, quando o motor de ID 14 foi conectado, o controller_manager não encontrou mais motores.

14:10

O motor de ID 14 foi removido e o teste continuou. O mesmo erro aconteceu quando o motor de ID 3 foi adicionado. Esse motor também foi retirado.

14:15

O teste seguiu até o último motor, nenhum motor aparentemente defeituoso foi encontrado. A comunicação funcionou bem com os 16 motores restantes.

14:45

Dois motores novos foram conectados. O controlador foi executado por volta de 20 vezes, em todos os testes todos os motores foram encontrados.

ELIR

logbook

TESTE DA PLACA DE POWER MANAGEMENT

Objetivos

Verificar possíveis problemas da montagem da placa de power management sem a presença de um dos conversores DC/DC

Descrição do teste

Os capacitores de acoplamento do regulador de tensão para o Atmega 32U4 são soldados e então a placa é alimentada com 14.4 Volts. A temperatura é monitorada com um multímetro com termopar, e os níveis de tensão com um multímetro comum.

DATA

05 OUTUBRO 2018

LOCALIDADE

SENAI CIMATEC
SALVADOR - BAHIA

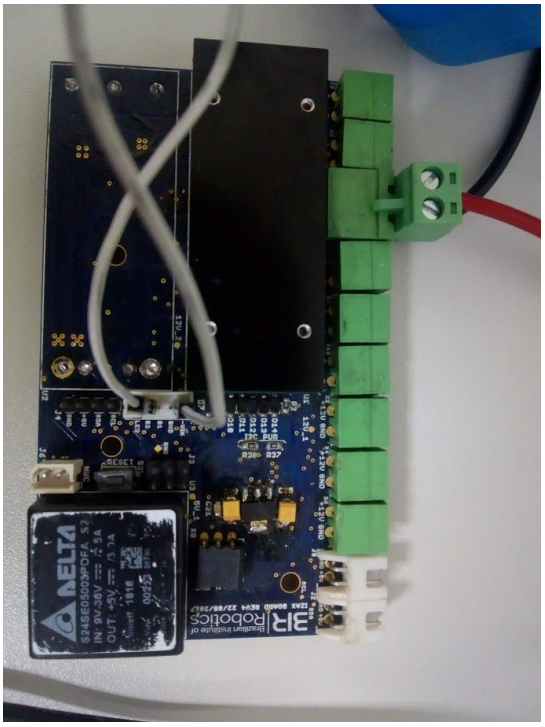
Mandruvah team

Cleber
Carlos
Ícaro
Davi

15:20
Capacitores de acoplamento foram soldados na placa, respeitando a polarização estabelecida no projeto de power management.
Capacitores de tântalo de 10uF de 16 Volts.

15:25
A placa foi alimentada com uma fonte de tensão à 14.4 Volts. Para que haja o funcionamento da placa é necessário realizar um curto entre os pinos -Vin e S2. Desta maneira inicia-se a operação da placa.


DELTA
S24SE05003PDFA S2
IN: 9V-36V
OUT: +5V
RESET
AIR
Robotics

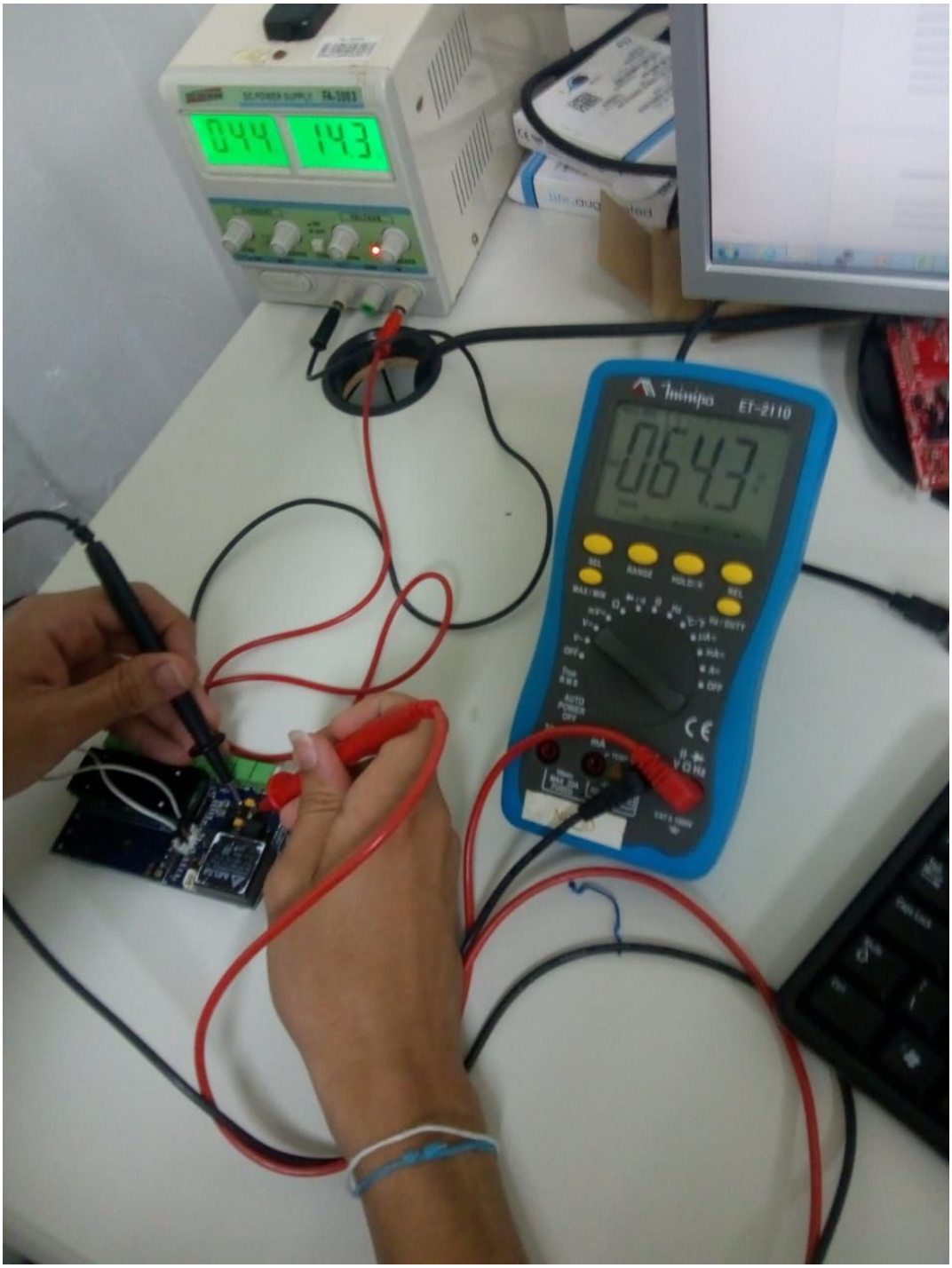

DELTA
S24SE05003PDFA S2
IN: 9V-36V
OUT: +5V
RESET
AIR
Robotics

15:27

A placa apresentou um aumento da temperatura do regulador de tensão 5 Volts, chegando a alcançar temperaturas em cerca de 110°C.

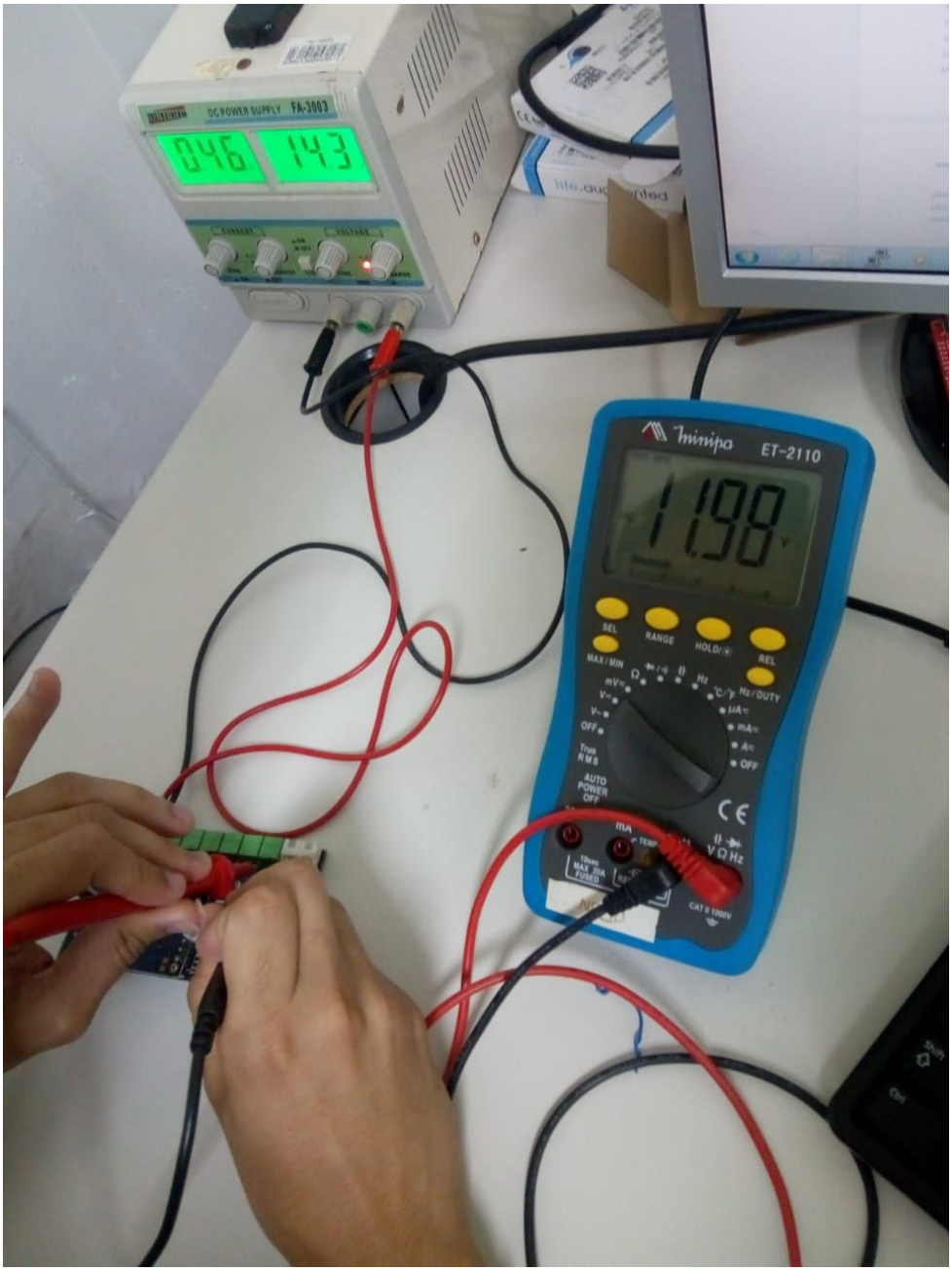
15:28

Os níveis de tensão gerados pelos reguladores também apresentaram níveis fora do esperado. O regulador de 5 volts apresentou tensões extremamente baixas, revelando um possível mal funcionamento, em torno de 70 milivolts



15:30

O conversor ainda presente na placa apresentou um funcionamento normal, gerando 12 Volts nas 3 portas de saída fixa, como esperado.



ELIR

logbook

TESTE COM O ROBÔ NA LINHA

Objetivos

Montar setup do teste, acessar a NUC remotamente e movimentar o robô na linha

Descrição do teste

O robô é posicionado na linha, é estabelecido um acesso remoto, são iniciados os controladores e movimenta-se o robô.

DATA

19 OUTUBRO 2018

LOCALIDADE

SENAI CIMATEC
SALVADOR - BAHIA

Mandruvah team

Cleber
Davi
Ícaro
Carlos

10:40

O robô foi levado até o estacionamento do SENAI CIMATEC onde há uma linha de transmissão de testes. O setup foi montado com uma bateria automotiva, um inversor de frequência 12 - 110V para conectar a fonte da NUC e os motores ligados diretamente à bateria.

11:05

Com o robô na linha e a NUC ligada e conectada a uma rede local exclusiva do teste, conectamos um notebook com o robô via SSH e assim tivemos acesso remoto ao terminal do robô.

11:10

Ao tentar carregar o *controller_manager.launch* nenhum motor foi encontrado. Conferimos então todas as conexões dos hubs de comunicação, nenhum mal contato foi encontrado. Percebemos que um dos terminais do conector do conversor USB-RS485 estava solto da placa.

13:20

Corrigimos o terminal solto do conversor. Novamente carregamos o *controller_manager.launch*, aconteceram erros de "*checksum"* e de *"wrong packet prefix"* e apenas 4 dos 18 motores foram encontrados.

13:40

Após os erros de comunicação, verificamos que quando corrigimos o conversor, os fios D+ e D- foram trocados acidentalmente.

13:45

Ainda depois de corrigir a conexão invertida, não conseguimos encontrar todos os motores, foi levantada a hipótese de que a bateria automotiva era o problema. Medimos a tensão e estava em 12,15V, decidimos voltar ao laboratório e testar com a fonte de bancada.

14:20

Repetimos o teste alimentando o robô com a fonte de bancada e o teste correu bem, nenhum dos erros anteriores aconteceu.

Apêndice D

Lista de componentes

ELIR project - BILL OF MATERIAL

$3.60

id	component	description	brand	part number	power/current	connection	unit cost [R$]	quantity	total cost [R$]
01	interface board		Phidgets	1019_1B	500mA (max)	USB	R$ 272.00	1	R$ 272.00
02	proximity sensor		ETT CO. Ltd	E18-D80NK npn	<25mA	Digital Output	R$ 29.00	5	R$ 145.00
03	temperature sensor		Texas Instruments	LM35	10mA	Analog Output	R$ 7.38	1	R$ 7.38
04	gps		Swift Navigation	Piksi 2.3.1	5V, 500mW	USB		1	R$ 0.00
05	imu		XSENS	Mti-1	44mW	USB		1	R$ 0.00
06	ultrassonic sensor		Maxbotix	EZ-1	5V, 2mA	Analog Output	R$ 107.82	1	R$ 107.82
07	current sensor		Phidgets	1122_0	5V, 10mA	Analog Output	R$ 106.02	3	R$ 318.06
08	lwir camera		FLIR	Lepton 1.0	140mW	I2C		1	R$ 0.00
09	bridge board I		STM	STM32F401 RE	160mA/0.64W	USB	R$ 49.79	1	R$ 49.79
10	bridge board II		STM	STM32L432	140mA/0.56W	USB	R$ 39.56	1	R$ 39.56
11	joint hub		Mandruvah	-	12V	RS485	R$ 25.20	3	R$ 75.60
12	servomotor I		Dynamixel	MX-28	1.4A/16.8W	RS485	R$ 900.00	5	R$ 4,500.00
13	servomotor II		Dynamixel	MX-106	5.2A/62.4W	RS485	R$ 1,800.00	13	R$ 23,400.00
14	adapter 485-usb							1	R$ 0.00
15	battery		Inspired Energy		89Wh/6,2Ah		R$ 879.98	2	R$ 1,759.97
16	power management board		-			USB	R$ 2,200.00	1	R$ 2,200.00
17	multiplex board		Inspired Energy	EB325A	15mA/0.36W	I2C	R$ 1,296.00	1	R$ 1,296.00
18	central processing		Intel	NUC515RYK			R$ 4,300.00	1	R$ 4,300.00
19	stereo camera		Stereolabs	ZED camera	380mA	USB	R$ 1,800.00	1	R$ 1,800.00
20	cabo usb								

Referências Bibliográficas

GONçALVES, P. C. *Protótipo de um Robô Móvel de Baixo Custo para Uso Educacional*. 2007. Disponível em: ⟨http://www.din.uem.br/~mestrado/diss/2007/goncalves.pdf⟩. 1

Sá, I. M. de A. *A Educação a Distância: Processo Contínuo de Inclusão Social*. [S.l.]: CEC, 1998. 1

Desenvolvimento do robô de inspeção.

Bruno Rodrigues
Bruno de Sousa
Frederico Garcia
Leandro S O Nozela

Salvador, Dezembro de 2019.