Universidade Federal do Maranhão
Centro de Ciências Exatas e Tecnologia
Curso de Ciência da Computação

MATHEUS CHAVES MENEZES

ESTUDO DA PLATAFORMA ROBODECK PARA APLICAÇÕES DE LOCALIZAÇÃO E MAPEAMENTO

São Luís
2017

MATHEUS CHAVES MENEZES

ESTUDO DA PLATAFORMA ROBODECK PARA APLICAÇÕES DE LOCALIZAÇÃO E MAPEAMENTO

Monografia apresentada ao curso de Ciência da Computação da Universidade Federal do Maranhão, como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Profº Dr. Alexandre César Muniz de Oliveira

São Luís

2017

Ficha gerada por meio do SIGAA/Biblioteca com dados fornecidos pelo(a) autor(a).
Núcleo Integrado de Bibliotecas/UFMA

Menezes, Matheus Chaves.
Estudo da Plataforma RoboDeck para Aplicações de Localização e Mapeamento / Matheus Chaves Menezes. – 2017.
46 f.

Orientador(a): Alexandre César Muniz de Oliveira.
Monografia (Graduação) – Curso de Ciência da Computação, Universidade Federal do Maranhão, São Luís, 2017.

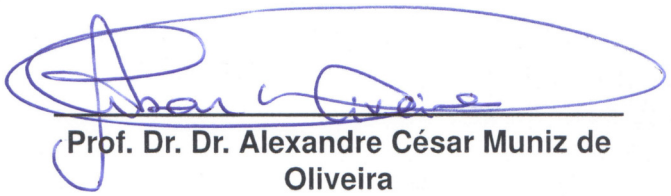
1. Localização. 2. Mapeamento. 3. RoboDeck. 4. Robótica Móvel. 5. Sistema Operacional Robótico. I. Muniz de Oliveira, Alexandre César. II. Título.

Matheus Chaves Menezes

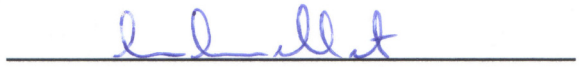
Estudo da Plataforma RoboDeck para Aplicações de Localização e Mapeamento

Monografia apresentada ao curso de Ciência da Computação da Universidade Federal do Maranhão, como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciência da Computação.

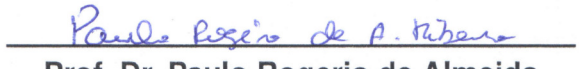
Trabalho aprovado. São Luís, 24 de janeiro de 2017:



Prof. Dr. Dr. Alexandre César Muniz de Oliveira
Orientador
Universidade Federal do Maranhão



Prof. Dr. Areolino de Almeida Neto
Universidade Federal do Maranhão



Prof. Dr. Paulo Rogerio de Almeida Ribeiro
Universidade Federal do Maranhão

São Luís
2017

Dedico este trabalho aos meus pais, Jorge e Coracy, que sempre me apoiaram, me ouviram e nunca deixaram faltar amor.

Agradecimentos

Agradeço primeiramente e principalmente aos meus pais. Sem eles, eu não seria quem sou, e muito menos chegaria aonde cheguei. Não cabe neste agradecimento o amor que sinto por eles, pois são as pessoas mais importantes da minha vida. Agradeço pelo esforço e paciência que sempre tiveram comigo, especialmente ao meu pai, pois sempre me incentivou a estudar e me mostrou que o conhecimento é a luz em meio à escuridão.

Agradeço também a minha família, pois sempre acreditaram em mim e me incentivaram a fazer o bem.

Agradeço ao meu orientador Prof. Dr. Alexandre César Muniz de Oliveira por todas as oportunidades oferecidas dentro da Universidade. Obrigado por me acolher desde o primeiro período e ter colaborado tanto para meu crescimento acadêmico e pessoal.

Agradeço aos meus amigos, e aos que hoje são colegas, por essa longa jornada desde o ensino fundamental até aqui. Em especial, agradeço aos meus amigos do IFMA.

Também agradeço a todos os meus amigos do LACMOR pela ajuda e conversas jogadas fora, ao meus sempre companheiros do PET e a tão especial turma de 2012.1. Aos meus amigos Marvin, Gilvan, Fernando, Felipe, Italo, Chrystian, Luann, Alex, Dayson e Eduardo que sempre estiveram dispostos a me ouvir e ajudar de alguma forma.

Agradeço em especial a Eliana Costa por toda ajuda durante minha jornada na universidade. Tudo me permitiu crescer e amadurecer. São coisas que levarei para o resto da minha vida. Os meus mais sinceros agradecimentos.

Agradeço à minha turma de Taekowndo pelos ensinamentos que vivi e vivo até hoje. A filosofia de vida nunca morrerá.

Por último, agradeço à você que estiver lendo esta monografia, pois quando nós partimos, o nosso legado é o que nos mantém vivos.

“Nossa existência deforma o universo. Isso é responsabilidade” - Neil Gaiman

RESUMO

Exploração autônoma não é uma tarefa trivial quando esta é destinada a robôs móveis. Dois fatores são determinantes para o sucesso dessa tarefa: sistema de localização e construção do mapa para navegação. Este trabalho apresenta um estudo sobre a plataforma robótica RoboDeck, em especial, no tocante às rotinas computacionais específicas necessárias para sua utilização na tarefa de exploração autônoma, com suporte do Sistema Operacional Robótico (*Robot Operating System* – ROS) e sensores de baixo custo. A arquitetura implementada através do ROS permite a substituição de sensores ou mesmo do kit robótico inteiro sem a necessidade de rescrita das rotinas. São realizados experimentos para dois objetivos principais, sendo eles a criação de mapas de um ambiente fechado e controlado e a modelagem da odometria para movimentação linear do robô. O mapa gerado pelo primeiro experimento apresenta características de um ambiente fechado, mas existem instabilidades em seu formato. A modelagem da odometria gerou resultados que diminuíram os erros entre distâncias percorridas pelo robô e estimadas pela odometria, mostrados por método estatístico de análise de variância (ANOVA).

Palavras-chaves: Robótica Móvel. Mapeamento. Localização. RoboDeck. Sistema Operacional Robótico.

ABSTRACT

Autonomous exploration is not a trivial task when it is intended for mobile robots. Two factors are decisive for the success of this task: localization and construction of the map for navigation. This paper presents a study about the robotic platform RoboDeck, in particular, with respect to the specific computational routines necessary for its use in the task of autonomous exploration, with support of the Robot Operating System (ROS) and low cost sensors. The architecture implemented through the ROS allows the replacement of sensors or even the entire robotic kit without the need of rewriting routines. Experiments are performed for two main objectives, being the creation of maps of a closed and controlled environment and the modeling of the odometry for linear movement of the robot. The map generated by the first experiment has characteristics of a closed environment, but there are instabilities in its format. The modeling of the odometry generated results that reduced the errors between distances traveled by the robot and estimated by odometry, shown by statistical analysis of variance method (ANOVA).

Keywords: Mobile Robotics. Mapping. Localization. RoboDeck. Robot Operating System.

Lista de ilustrações

Figura 1 – Visão geral do robô RoboDeck 16
Figura 2 – Visão geral da arquitetura dos Módulos do RoboDeck 17
Figura 3 – Comunicação entre Nós através de tópicos e serviços 19
Figura 4 – Diagrama de comunicação do *middleware* 20
Figura 5 – Exemplos de reflexões indesejadas. 22
Figura 6 – Modelagem do sonar 23
Figura 7 – Representação das células no mapa de grades de ocupação 24
Figura 8 – Mensagens publicadas no nó de odometria 27
Figura 9 – Mensagens publicadas no nó de odometria. 28
Figura 10 – Falha na comunicação entre RoboDeck e ROS 30
Figura 11 – Ambiente fechado e controlado utilizado para mapeamento. 31
Figura 12 – Proposta para criação do mapa global. 32
Figura 13 – Mapas individuais de cada quadrante do ambiente. 33
Figura 14 – Mapa global do ambiente. 34
Figura 15 – Médias das distâncias exatas e aproximadas do robô. 36
Figura 16 – Média do Erro Absoluto entre Distâncias Percorridas e Estimadas. . 38
Figura 17 – Gráficos de barras dos Erros Absolutos das amostras. 38
Figura 18 – Média do Erro Absoluto entre Distâncias Percorridas e Estimadas. . 39
Figura 19 – Médias das distâncias percorridas e estimadas do robô. 41
Figura 20 – Média do erros após correção. 41
Figura 21 – Gráficos de barras dos Erros Absolutos das amostras após correção 43

Lista de tabelas

Tabela 1 – Distâncias percorridas pelo robô para 100 cm. 29
Tabela 2 – Média dos valores de 10 experimentos para cada comando 36
Tabela 3 – Análise de variância (ANOVA) entre as médias dos erros 37
Tabela 4 – Análise de *Post-hoc* entre as médias dos erros 37
Tabela 5 – Medidas das distâncias com 10 experimentos para cada comando . 40
Tabela 6 – Teste t pareado entre os erros absolutos 42
Tabela 7 – Análise de variância (ANOVA) entre as médias dos erros 42
Tabela 8 – Teste *Post-hoc* entre os erros após correção 43

Lista de abreviaturas e siglas

API	*Application Programming Interface*
DE	Distância Estimada
DP	Desvio Padrão
DPe	Distância Percorrida
JDK	*Java Development Kit*
MCC	Módulo de Controle e Comunicação
MCR	Módulo de Controle Robótico
MCS	Módulo de Controle e Sessão
ROS	*Robot Operating System*
TOF	*time-of-flight*
UART	*Universal Asynchronous Receiver/Transmitter*
USB	*Universal Serial Bus*

Sumário

1 INTRODUÇÃO 13
1.1 Breve Revisão Bibliográfica 14
1.2 Objetivos 15
1.3 Organização do Trabalho 15

2 MATERIAIS E MÉTODOS 16
2.1 RoboDeck 16
2.1.1 Módulos de Controle e Alta Performance 17
2.2 ROS (Robot Operating System) 18
2.2.1 Conceitos do ROS 18
2.3 *Middleware* de Integração ROS-RoboDeck 19
2.4 Sensores Ultrassônicos 21
2.4.1 Modelo de Distribuição do Sensor Ultrassônico 22
2.5 Odometria 22
2.6 Mapa com Grades de Ocupação 24

3 PROCEDIMENTOS EXPERIMENTAIS 25
3.1 Odometria para RoboDeck 25
3.2 Mapeamento 30

4 MODELAGEM DO ERRO DA ODOMETRIA 35
4.1 Definição do experimento 35
4.2 Calculo e Modelagem do Erro 35
4.3 Teste de Correção do Erro 40

5 CONCLUSÃO 44
5.1 Trabalhos Futuros 45

REFERÊNCIAS 46

1 Introdução

Segundo (WOLF et al., 2009), o estudo da robótica móvel é um tema relevante e atual para nossa sociedade. O seu estudo, pesquisa e desenvolvimento tiveram grandes avanços nas últimas duas décadas, apresentando-se cada vez mais em atividades não só industriais, mas também domésticas e militares. Seu estudo possui ainda um contexto multidisciplinar, necessitando de conhecimento na área eletrônica, elétrica e computacional, sendo esta última muito importante para provê ao robô a capacidade de possuir sistema mais seguros, robustos, inteligentes e autônomos (WOLF et al., 2009).

A construção de um mapa é um processo muito importante na navegação de um robô móvel. Ele é responsável por permitir que o robô alcance objetivos ou mapeie áreas inexploradas ou de difícil acesso para seres humanos. Durante o processo de mapeamento, informações necessárias sobre a localização, como posição e rotação do robô, podem ser obtidas através de odometria, por exemplo, e a detecção de objetos ao longo da construção do mapa pode ser realizada por diversos sensores, desde baratos e de simples manuseio, como os sensores ultrassônicos (VARVEROPOULOS, 2005) (YENILMEZ; TEMELTAS, 2002), aos mais avançados, de maior precisão, como os *laser scanners* (JUANG; WANG, 2015) (GUIVANT; NEBOT; BAIKER, 2000).

Segundo (BEZERRA; ALSINA; MEDEIROS, 2004), os métodos de localização de robôs móveis podem ser classificados em duas categorias: métodos de localização relativa e métodos de localização absoluta. Os métodos de localização reativa utilizam as posições anteriormente obtidas para definir a nova posição do robô. A odometria é um exemplo de método de localização relativa e os *encoders* óticos são os sensores mais utilizados, porém, alguns fatores influenciam para o acúmulo de erros da posição e orientação do robô ao longo do tempo. Já o método de localização absoluta utiliza dados atuais dos sensores para localizar o robô em relação a um referencial fixo absoluto. O reconhecimento de imagens de marcadores globais é um exemplo do método de localização absoluta.

Os métodos de localização relativa possuem a vantagem se serem mais rápidos, porém, como dito anteriormente, a odometria produz erros que se propagam durante o tempo, já os métodos de localização absoluta, apesar de levarem mais tempo de processamento computacional, retornam uma localização do robô mais precisa.

Neste trabalho, a plataforma robótica RoboDeck, fabricada pela empresa nacional XBOT e adquirida pela Universidade Federal do Maranhão, é alvo de um estudo visando a empregá-la para a tarefa de exploração autônoma. O RoboDeck foi projetado com a intenção de promover o desenvolvimento educacional na área da pesquisa e

ensino em robótica. Possui diversos sensores, como acelerômetro, bússola, *encoders*, sensores infravermelhos e ultrassônicos (XBOT, 2011).

O *framework* ROS (*Robot Operating System*) é utilizado para a criação de algoritmos de odometria do robô, além de rotinas especificas de movimento, rotação e captura de dados dos sensores. O ROS foi idealizado para facilitar no gerenciamento e na escrita de algoritmos robóticos, tendo compatibilidade com vários robôs no mercado (ROS. . . , 2014b). O ROS ainda oferece funcionalidades esperadas de um sistema operacional, como abstração de hardware, comunicação entre processos, controle de dispositivos de baixo nível, etc. Devido ao seu nível de abstração de hardware, o ROS permite a integração de diversos dispositivos conectados a ele de forma simples (ROS. . . , 2014b).

A integração entre o RoboDeck e o ROS é feita por um *middleware* que provê uma API (*Application Programming Interface*) para comunicação entre ambos. O método de comunicação é baseado em requisição/resposta, onde a aplicação no ROS requisita determinada execução de comando no robô, e a resposta é retornada para a aplicação (GASHU, 2014).

1.1 Breve Revisão Bibliográfica

Em (OGISO et al., 2015), é proposta uma metodologia de localização utilizando odometria baseada em *encoders* óticos e microfones para captar frequências específicas de sons com posições conhecidas para correção dos erros gerados pela odometria, onde os resultados mostraram uma diminuição significante do erro de odometria com baixo custo computacional.

Em (BURGARD et al., 1999), é apresentada uma metodologia com a proposta de resolver o problema de mapeamento e localização usando um algoritmo de Maximização de Expectativa (*expectation–maximization algorithm*, em inglês). O trabalho apresenta a construção de um mapa global formado por vários mapas menores locais, apropriando-se do fato que os erros da odometria a curto prazo são geralmente pequenos. A abordagem é cabível quando se utiliza sensores de pouca precisão para construção de mapas em ambientes grandes.

Um sistema robusto para resolver o problema de mapeamento e localização utilizando sensores ultrassônicos é apresentada por (TARDÓS et al., 2002). No trabalho, é utilizado um anel de sonares para construção de mapas locais e sensores de odometria para definir a localização do robô. É aplicada a transformada de Hough aos retornos do sonar obtidos ao longo de curtas distâncias (cerca de 1-2 m de distância), a fim de manter os erros de odometria pequenos.

1.2 Objetivos

O objetivo deste trabalho é o desenvolvimento de rotinas específicas para interface entre o *framework* ROS e o *kit* robótico RoboDeck que permitam utilizar o kit para a tarefa de exploração autônoma. Para tanto serão realizadas as seguintes atividades:

i Levantamento bibliográfico sobre trajetória em tempo real e exploração autônoma de robôs móveis;

ii Estudo do *kit* robótico RoboDeck, em especial aos seus sensores ultrassônicos, *encoders*, e sua comunicação com outros dispositivos por *Wi-fi*, bem como seus comandos de alto nível;

iii Estudo ROS e do *middleware* que provê comunicação entre ROS e o RoboDeck;

iv Ampliação das funcionalidades *middleware* com comandos de leitura e sobrescrita dos *encoders*;

v Desenvolvimento de algoritmos para odometria e mapeamento, bem como sua validação;

vi Modelagem do erro da odometria;

1.3 Organização do Trabalho

Este trabalho está organizado em 5 capítulos, sendo este o primeiro.

No Capítulo 2, estão descritos os materiais, bem como a conceitualização do kit robótico e das ferramentas computacionais, e os fundamentos teóricos para entendimento deste trabalho.

O Capítulo 3 apresenta os procedimentos experimentais da primeira parte do trabalho. Nela, são apresentadas as equações para movimento linear e rotacional do RoboDeck e alguns experimentos realizados para validação dos algoritmos. Também estão presentes a proposta de mapeamento do ambiente físico e controlado, os experimentos de mapeamento e seus resultados.

No Capítulo 4, é realizada a modelagem do erro linear presente no sistema de odometria para o RoboDeck. É proposta uma equação para diminuição desse erro e uma análise estatística desse erro.

O Capítulo 5 apresenta as conclusões do trabalho e sugestões para trabalhos futuros.

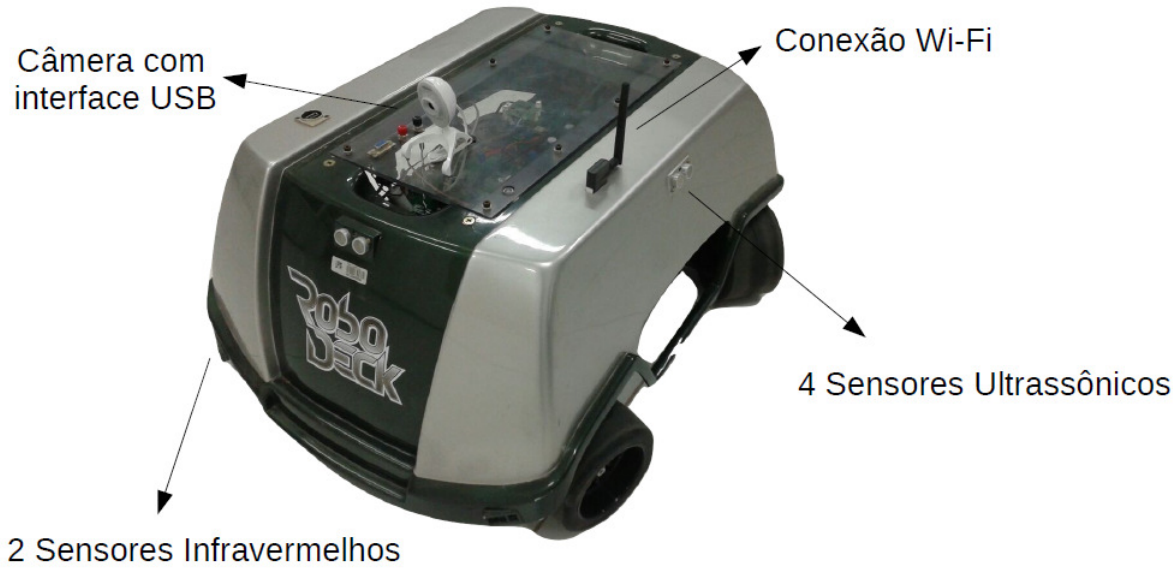
2 Materiais e Métodos

Este Capítulo apresenta os materiais, ferramentas computacionais e fundamentos teóricos utilizados no desenvolvimento deste trabalho.

2.1 RoboDeck

O RoboDeck é uma plataforma robótica educacional de código aberto produzido pela empresa nacional Xbot. Foi idealizado para promover o desenvolvimento educacional, sobretudo na área da pesquisa (XBOT, 2011). O pacote educacional conta com a plataforma robótica móvel, o robô, um SDK (*software Development kit*) para desenvolvimentos de aplicações, um ambiente de programação baseado na linguagem C/C++ e um *software* para testes. O robô, ilustrado na Figura 1, possui quatro sensores ultrassônicos distribuídos no centro de cada lado. Esses sensores são utilizados neste trabalho como extração do vetor de medidas para o mapeamento do ambiente. Além disso, o robô possui infravermelho, câmera USB, acelerômetro, bússola, *encoders* nos motores de tração e GPS. A comunicação como o RoboDeck pode ser feita por *WiFi* ou *ZigBee*.

Figura 1 – Visão geral do robô RoboDeck

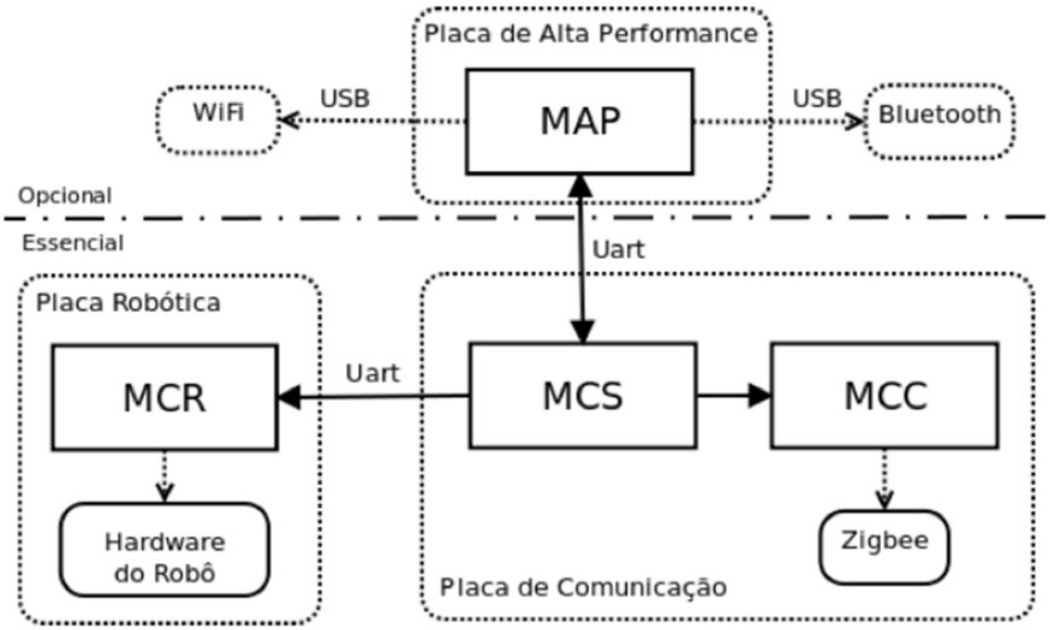


Cada *encoder* do RoboDeck possui resolução de pulsos que variam de 0 a 400 por uma volta completa da roda, ou seja, uma volta completa da roda é igual a 400

pulsos do *encoder*. A resolução do *encoder* é a quantidade de pulsos que ele gera antes de voltar a posição inicial (ZANOLLA et al., 2014).

O RoboDeck foi desenvolvido para ser um robô modular que permite a expansão tanto de seu código quanto do seu *hardware*. Com essa ideia de modularidade, o *hardware* básico do robô é dividido em duas partes: o essencial e o opcional. O *hardware* essencial possui dois microcontroladores que se comunicam por uma UART. Nele, rodam *softwares* que não se baseiam em sistemas operacionais e são escritos na linguagem C. Já o *hardware* opcional é composto por uma placa capaz de executar o sistema operacional Linux, e os *softwares* que rodam neste *hardware* são escritos na linguagem C++. A Figura 2 apresenta as duas principais divisões de *hardware* e *software* do robô (MUñOZ, 2011).

Figura 2 – Visão geral da arquitetura dos Módulos do RoboDeck



Fonte: (MUñOZ, 2011)

2.1.1 Módulos de Controle e Alta Performance

Como apresentado na figura 2, o RoboDeck possui em seu *hardware* essencial, módulos responsáveis pela parte física, comunicação, identidade e autenticação dos controladores do robô. Tais módulos de controle são apresentados como:

- *Módulo de Controle Robótico* (MCR)
- *Módulo de Controle e Comunicação* (MCC)

- *Módulo de Controle de Sessão* (MCS)

o MCR é o *software* responsável pelo gerenciamento do *hardware* relacionado aos sensores e atuadores do robô. Ele executa comandos seriais que chegam pela UART do MCS. O MCC é responsável pela comunicação de controle do RoboDeck e é implementada pelo controle *Zigbee*. O MCS é responsável pela identidade do robô, implementando conceitos de sessão aos comandos vindos dos controladores do robô, impedindo o robô de responder a dois comandos ao mesmo tempo (MUñOZ, 2011).

Em seu *hardware* opcional, o RoboDeck possui um *Módulo de Alta Performance* (MAP) designado para dar autonomia ao robô e comunicação de banda larga com o meio externo. O MAP foi desenvolvido para rodar em uma placa microprocessada capaz de rodar um sistema operacional Linux qualquer e que possua interfaces UART e USBs. Ele permite que sejam desenvolvidos aplicativos robóticos capazes de expandir os comandos robóticos que podem ser acessados pelos controladores do robô (MUñOZ, 2011).

2.2 ROS (Robot Operating System)

ROS é um meta sistema operacional de código aberto utilizado no gerenciamento de sistemas robóticos. Ele fornece os serviços de um sistema operacional, incluindo abstração de *hardware*, controle de dispositivos de baixo nível, implementação de funcionalidades usualmente utilizadas, troca de mensagens entre processos e gerenciamento de pacotes. Ele também fornece ferramentas e bibliotecas para a obtenção, construção, escrita e execução de códigos em vários dispositivos (ROS. . . , 2014b). Contudo, o ROS atua em conjunto com outro sistema operacional Linux, não podendo ser instalado em máquinas nativamente (O'KANE, 2014).

O ROS *runtime* é uma rede peer-to-peer de processos que são livremente acopladas utilizando a infraestrutura de comunicação do ROS, onde essa comunicação pode ser síncrona ou assíncrona. Os processos executados dentro do ROS também são conhecidos como *Nós* (ROS. . . , 2014b).

2.2.1 Conceitos do ROS

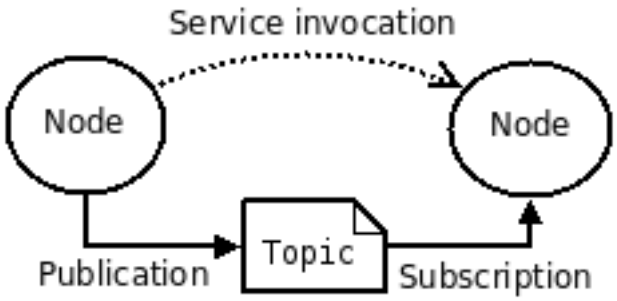
ROS possui três níveis de conceitos: O sistema de arquivos, o grafo de computação e o nível de comunicação (ROS. . . , 2014a). Abaixo, são apresentados alguns conceitos do nível de grafo de computação:

- **Nós:** Nós são processos executáveis em nível computacional. Um sistema de controle robótico possui muitas tarefas executáveis, como controle das rodas,

obtenção de dados dos sensores e etc. Um nó é atribuído para cada tarefa. Estruturas de nós bastante utilizadas são os *Publishers* e *Subscribers*. A estrutura de um nó*publisher* é basicamente enviar mensagens a um ou mais nós *subscribers*. A comunicação é feita através da troca de mensagens.

- **Mensagens:** Nós se comunicam uns com ou outros através da troca de mensagens. Uma mensagem é uma estrutura de dados simples que possui campos digitáveis com tipos primitivos padrão, como inteiro, ponto flutuante, booleano e etc. As mensagens podem incluir estruturas e matrizes com os mesmos tipos primitivos.
- **Tópicos:** Nós enviam mensagens através da sua publicação a um determinado tópico. O tópico é um nome utilizado para identificar o conteúdo da mensagem. Nós que que se interessem por determinado tipo de mensagem devem fazer a leitura do tópico correto. Essa estrutura permite que vários nós possam publicar no mesmo tópico, assim como vários nós podem ler do mesmo tópico. A Figura 3 mostra como a comunicação através de tópicos é realizada.

Figura 3 – Comunicação entre Nós através de tópicos e serviços



Fonte: (ROS. . . , 2014a)

- **Serviços:** Requisições e respostas entre nós são feitas através de serviços, Serviços são definidos por um par de estruturas de mensagem: um para o pedido e um para a resposta. Um nó oferece um serviço com um nome e um cliente utiliza o serviço enviando um mensagem de pedido e aguarda a resposta. A Figura 3 também demostra superficialmente como a comunicação é realizada através de serviços.

2.3 *Middleware* de Integração ROS-RoboDeck

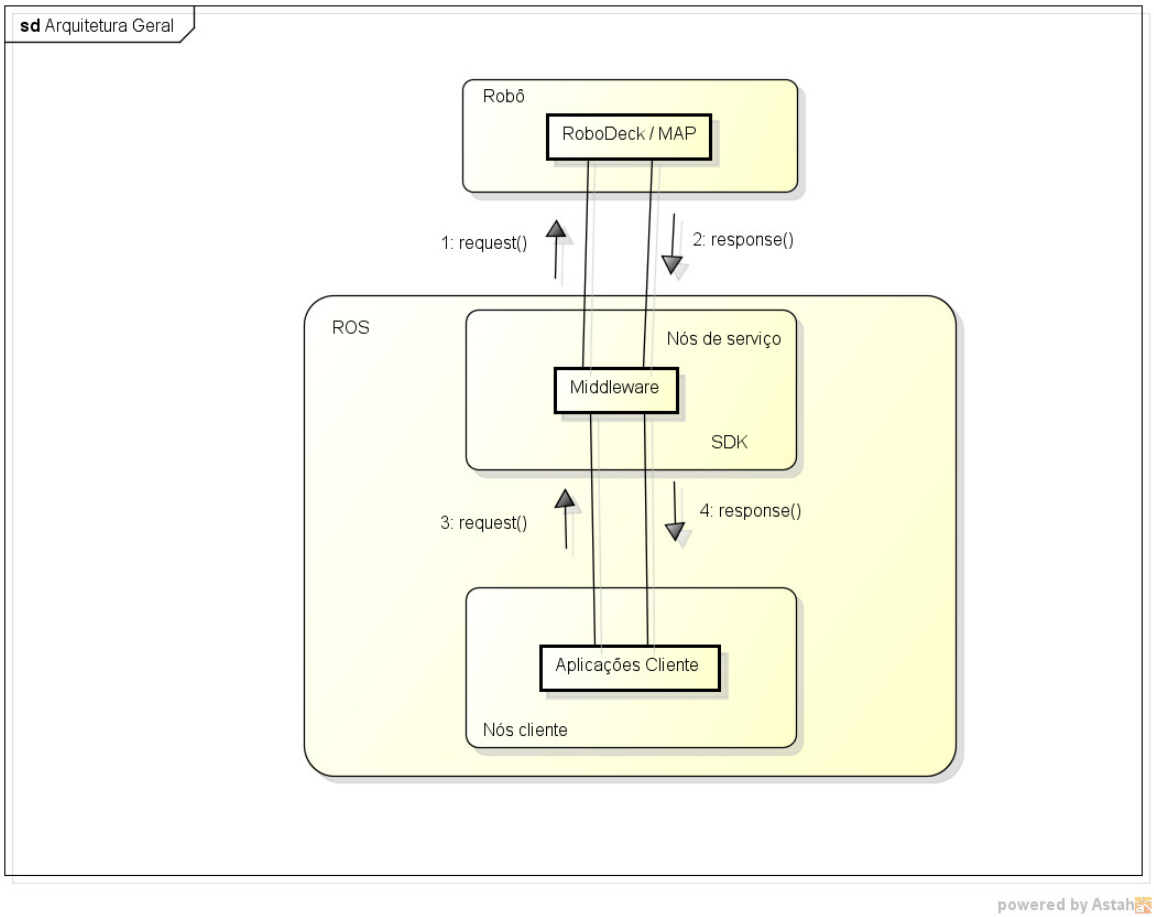
O *middleware* desenvolvido por (GASHU, 2014) funciona como um intermediador da comunicação dos programas clientes escritos no ROS com o RoboDeck. Ele foi

desenvolvido pra se comunicar com a placa MAP do RoboDeck e seu objetivo é oferecer uma API por meio do ROS para esta comunicação.

O *middleware* foi desenvolvido utilizando o *Java Development Kit* (JDK) disponibilizado pela XBOT para o RoboDeck e o ROSJava. O ROSJava é um pacote disponível para o ROS e oferece serviços para a escrita de algoritmos (nós) na linguagem Java e criação de projetos para integração com outras plataformas, como o sistema operacional *Android* (ROS. . . , 2013).

A comunicação entre o robô e o ROS funciona através de requisição/resposta. O cliente que quer se comunicar com o robô deve solicitar uma requisição através de um nó do ROS. Ele faz a tradução tanto da requisição do cliente para o robô quanto da resposta do robô para o cliente. Tanto o *middleware* quanto as aplicações clientes utilizam o ROS para permitir a comunicação, assim, eles devem ser executados como nós de rede do ROS. (GASHU, 2014). A Figura 4 mostra o diagrama de comunicação do *middleware*.

Figura 4 – Diagrama de comunicação do *middleware*



Fonte: (GASHU, 2014)

2.4 Sensores Ultrassônicos

Sensores ultrassônicos, também conhecido como sonares, são vastamente utilizados na robótica móvel. Eles emitem pulsos acústicos por um transdutor, e as reflexões dos pulsos para determinar a distancia do sensor para um objeto. Conhecendo-se a velocidade do som, a distancia para o objeto é proporcional ao tempo de viagem da partícula. Esta técnica também é conhecida como *time-of-flight* (TOF)(KLEEMAN; KUC, 2008). A Equação 2.1 abaixo representa o cálculo da distância *d* retornada por um sensor ultrassônico:

$$d = \frac{v \cdot t}{2} \tag{2.1}$$

Onde v é a velocidade de propagação do som (aproximadamente 343 m/s) e t é o tempo de viagem da partícula em segundos. Deve-se considerar o tempo da partícula até o objeto e seu eco até o sensor, por esse motivo, há divisão do tempo por dois.

As principais vantagens da utilização desses sensores são seu baixo custo, vasta área de atuação, manuseio simples e, normalmente, leituras precisas (ARAUJO; GRUPEN, 1998). Os sonares são capazes de dar uma aproximação com alta fidelidade de distância, e estas informações geralmente são utilizadas para evitar colisão entre o robô e obstáculos, e tais medidas ainda podem ser difíceis de se obter por outros métodos. (HEINEN; OSÓRIO, 2002)

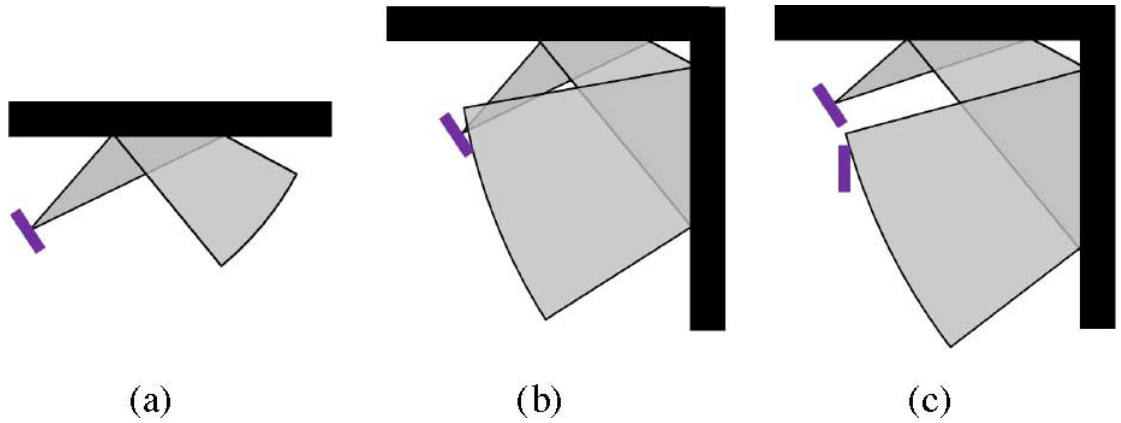
Também há a possibilidade de utilização desses sensores como solução para localização de robôs, porém, os sistemas de ultrassom são adequados somente em pequenas áresas de trabalho e se houver a inexistência de obstáculos que possam interferir com a propagação do sinal (SECCHI, 2012).

Apesar das suas vantagens, sensores ultrassônicos sofrem de dois problemas conhecidos: medidas incorretas e incerteza angular(LEE; CHUNG, 2009).

1. *Medidas incorretas*: sensores de sonar frequentemente não conseguem detectar o obstáculo mais próximo por causa dos reflexos indesejáveis. A reflexão faz uma leitura errada que pode formar obstáculos fantasmas ou omitir obstáculos reais do mapa. A Figura 5 apresenta situações onde ocorrem reflexões indesejadas: (a) *Reflexão especular* se refere à resposta do sonar que não é refletida de volta diretamente do objeto alvo (ZAKI; ARAFA, 2012); (b) *Reflexão de muitas ordens* ocorre quando a partícula atinge dois ou mais obstáculos antes de retornar ao sensor (DUDEK; JENKIN, 2010); (c) *Crosstalk* ocorre quando a partícula vem de um sensor diferente (BANK; KAMPKE, 2007).

2. *Incerteza angular*: Sonares fornecem informações diretamente sobre o obstáculo mais próximo, porém não dão informações sobre a angulação deste objeto(CHOSET; NAGATANI; LAZAR, 2003). A incerteza angular pode esconder aberturas estreitas e distorcer o mapa(KLEEMAN; KUC, 2008).

Figura 5 – Exemplos de reflexões indesejadas.



Fonte: (LEE; CHUNG, 2009)

2.4.1 Modelo de Distribuição do Sensor Ultrassônico

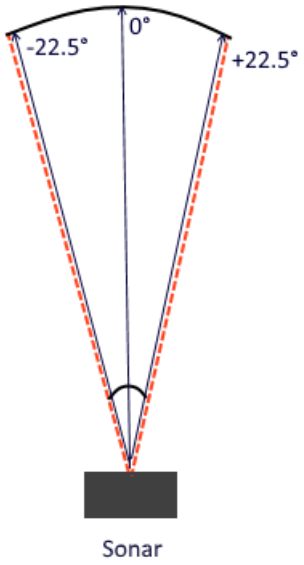
A energia irradiada pelo transdutor do sonar não está focada em uma linha reta, mas distribuída em um cone. O tamanho do cone é medida pelo ângulo do feixe de largura do transdutor de sonar (VARVEROPOULOS, 2005). Nas especificações do fabricante do RoboDeck, o sensor cobre um feixe cônico de aproximadamente 45°. A Figura 6 mostra a distribuição do feixe cônico do sonar utilizado neste trabalho.

2.5 Odometria

Odometria é um método de navegação utilizado para posicionamento de um robô móvel, obtida calculando a posição do robô utilizando sensores quem medem deslocamento. Os *encoders* são os sensores comumente usados para esta função, especialmente os *encoders* rotativos, pois transformam o movimento de rotação dos motores em pulsos ou sinais digitais.

A ideia básica da odometria é a integração de incremento da informação de movimento sobre o tempo. Segundo (ZAKI; ARAFA, 2012), é bem conhecido que odometria proporciona boa precisão a curto prazo, tem baixo custo computacional e permite taxas de amostragem muito elevadas. Porém, seu modelo gera erros inevitáveis ao longo do tempo. Quanto maior for a distância percorrida pelo robô móvel, maiores são os

Figura 6 – Modelagem do sonar



erros acumulados. Contudo, (THRUN; BURGARD; FOX, 2005) afirma que experiências práticas mostram que a odometria, apesar de apresentar erros, é usualmente mais precisa que modelos de velocidade.

Erros odométricos são causados por diversos fatores e podem ser classificados como sistemáticos e não-sistemáticos (ZAKI; ARAFA, 2012). Geralmente, erros sistemáticos são mais graves, pois o erro acumulado é constante. Em ambientes fechados com poucas irregularidades no solo, os erros sistemáticos contribuem mais que erros não-sistemáticos para os erros odométricos. Porém, em superfícies irregulares, erros não-sistemáticos são predominantes.

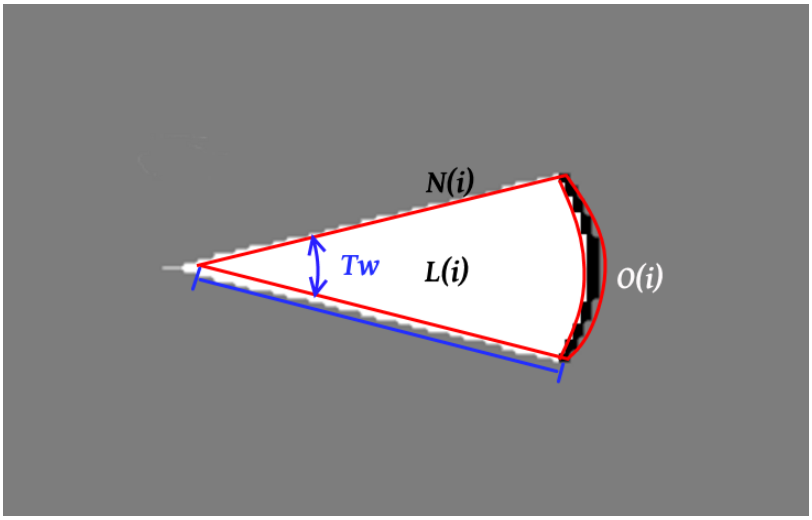
1. Erros Sistemáticos:
    - Desalinhamento das rodas;
    - Diâmetros das rodas diferentes;
    - *Encoders* com resolução finita;
    - Taxa de amostragem dos *Encoders* finito;
    - Distância entre eixos real das rodas diferente da distância entre eixos nominal;
    - Desbalanceamento de cargas.
2. Erros Não-sistemáticos:

- Deslocamento em pisos irregulares;
- Passagem por objetos inesperados no piso;
- Derrapagem nas rodas por diversos fatores.

2.6 Mapa com Grades de Ocupação

Segundo (ELFES, 1989), a grade de ocupação é uma área aleatória multidimensional que mantém estimativas estocásticas do estado de ocupação das células em uma grade espacial. Para construir um mapa derivado de sensor do robô, as estimativas do estado da célula são obtidas através da interpretação das leituras de entrada usando os modelos dos sensores. Neste trabalho, o modelo do sonar foi utilizado para a criação dos mapas, e os valores das células são discretizado entre 0 e 1, com intervalo de 0,5 unidade, onde o valor 0 representa livre, 1 representa ocupado e 0,5 representa estado indefinido. A atualização de cada célula no mapa global é feita com dados sobre a posição do robô e os dados dos sensores. A Figura 7 demostra os estados das células dentro do mapa. *N(i)* representa células desconhecidas (cinza), *L(i)* são células livres (branco), *O(i)* são as células ocupadas (preto) e *Tw* é o ângulo de abertura do cone.

Figura 7 – Representação das células no mapa de grades de ocupação



3 Procedimentos Experimentais

Neste Capítulo, apresentam-se os procedimentos experimentais deste trabalho. Inicialmente, são apresentadas algumas observações sobre os testes do *middleware* ao início deste trabalho. Na Seção 3.1, são definidas equações de movimento linear e rotacional para odometria do RoboDeck. Depois, são realizadas alguns experimentos sobre o movimento linear do robô e o sistema de odometria. Na seção 3.2, é apresentado um experimento de mapeamento utilizando os sensores ultrassônicos do RoboDeck.

Foram testados todos os comandos e funcionalidades do RoboDeck disponíveis com o software de testes da XBOT, bem como movimentação para frente e para trás, rotação, curvas para direita e esquerda, leitura dos *encoders*, comando de parada do robô, leitura dos sensores ultrassônicos, etc. De todos os comandos disponíveis, somente o GPS do robô não respondeu ao comando de inicialização. Foi observado que, ao executar o comando de rotação, o robô gira em seu próprio eixo, tornando possível, apesar de possuir somente 4 sensores ultrassônicos, a obtenção de mais dados dos sonares na mesma posição. Também foi observado que os comandos enviados ao RoboDeck são executados constantemente, sendo necessário o envio de outro comando para interromper o anterior, com exceção dos comandos de leituras de sensores.

A comunicação entre o RoboDeck e o ROS é realizado através de uma rede local *Wi-fi* do RoboDeck, sendo necessária uma senha disponibilizada para o acesso à rede.

A primeira etapa do procedimento para a criação de rotinas específicas para o RoboDeck foi sua integração utilizado o *middleware* de integração RoboDeck-ROS. O sistema operacional utilizado nos experimentos foi Linux na sua distribuição Ubuntu em sua versão 14.04 LTS, e a versão do ROS foi a "Indigo". Após estabelecer a comunicação entre o robô e o *middleware*, foram testadas as rotinas já desenvolvidas e disponibilizadas junto com software. Foi detectado que algumas requisições não funcionavam e/ou não foram implementadas, como os comandos de leitura dos *encoders* do RoboDeck, bem como sua sobrescrita.

3.1 Odometria para RoboDeck

Para que fosse possível a criação do algoritmo de odometria, era necessário realizar chamadas aos *encoders* do robô, porém as mesmas não existiam e necessitavam ser acrescentadas ao *middleware*. Após a implementação das chamadas de

leitura e sobrescrita dos *encoders*, permitiu-se a criação de rotinas de movimentação linear para robô com determinada restrição de distância a ser percorrida, assim como a criação de rotinas de rotação com determinada restrição de ângulo a ser rotacionado, além de dar contribuição ao trabalho de desenvolvimento do *middleware*.

Em experimentos, notou-se que algumas chamadas para leitura do *encoder* do motor de tração esquerdo não aconteciam de forma bem sucedida. A medida adotada foi realizar somente cálculos utilizando somente o *encoder* do motor direito. Assume-se que erros nos cálculos de distância podem acontecer tendo como base somente a leitura de um *encoder*.

O cálculo da distância percorrida pelo robô foi realizado utilizando seu *encoder* ótico incremental. Como o diâmetro da roda é conhecido, e também sabendo que 400 pulsos do *encoder* representam uma revolução da roda, a distância percorrida pelo robô em um instante t é representada na Equação 3.1, onde E_t é a leitura efetuada por um dos *encoder* no tempo t e r é o valor do raio das rodas.

$$D(t) = \frac{E(t) \cdot 2 \cdot \pi \cdot r}{400} \tag{3.1}$$

O cálculo do ângulo rotacionado pelo robô é feita de forma semelhante. Sabendo o valor do maior comprimento do robô, e que o movimento de rotação permite ao robô girar no próprio eixo, a distância de uma rotação completa do robô é dada pela Equação 3.2. R é a metade do maior do comprimento do robô (37 cm).

$$C = 2 \cdot \pi \cdot R \tag{3.2}$$

A distância C corresponde a um giro de 360°do robô. Assim, dado um determinado ângulo α (medido em graus), a distância correspondente é dada pela Equação 3.3. Após a conversão do ângulo α desejado para rotação em uma distância C, é aplicada a equação 3.1 ao comando de rotação do robô, assim, o robô rotaciona uma distância correspondente ao ângulo.

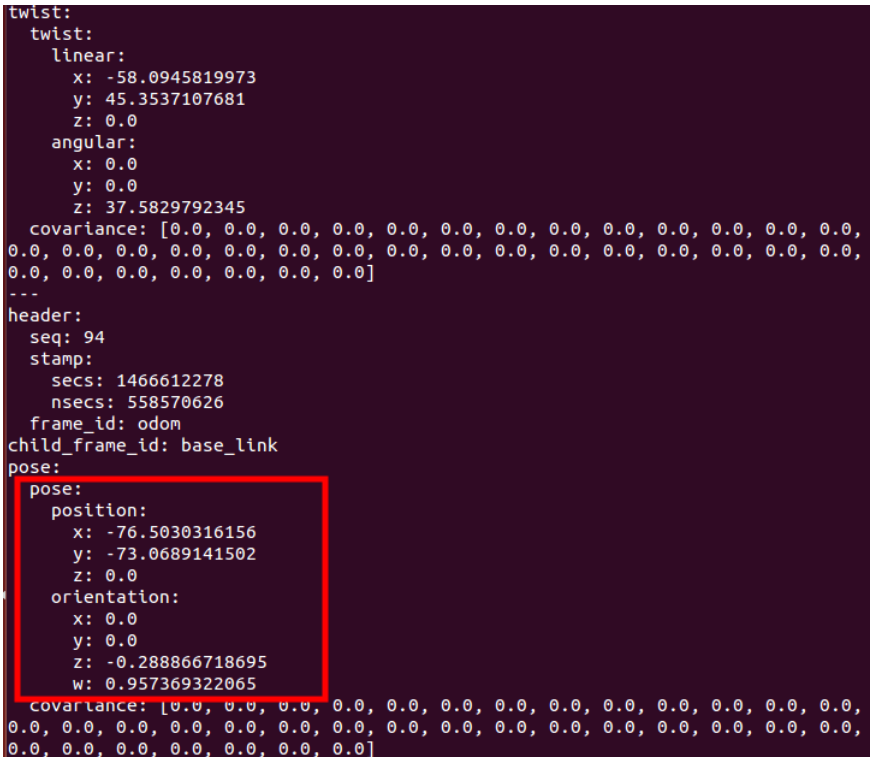
$$C(\alpha) = \frac{\pi \cdot \alpha \cdot R}{180} \tag{3.3}$$

Com as equações acima definidas, elas foram aplicadas em um nó chamado "pub-robodeck"no ROS. Esse nó tem o propósito de enviar comandos de movimentação linear e rotacional específicas ao RoboDeck.

Dois tópicos intitulados “/distance” e “/angle” são criados para que os nós do ROS possam escrever e ler mensagens sobre eles. O nó "pub-robodeck"constantemente publica nesses dois tópicos os valores da distância linear do robô e o seu ângulo rotacional.

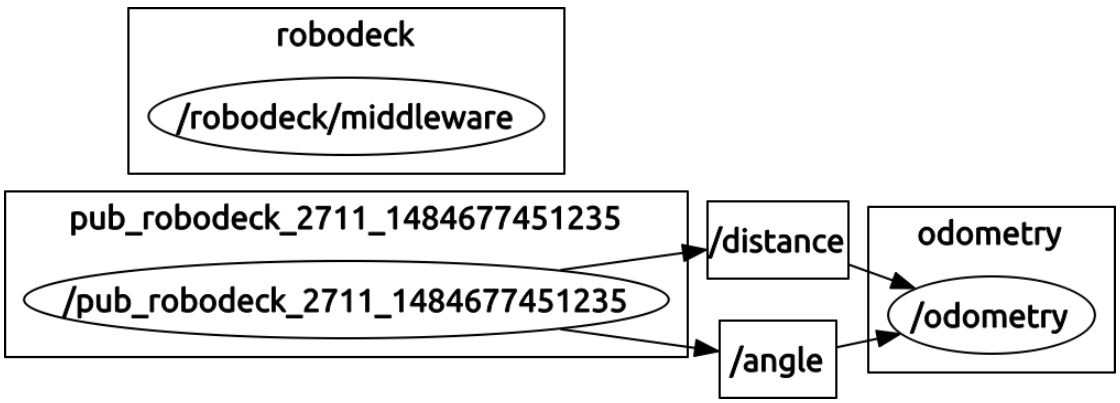
Após a etapa acima concluída, foi escrito um algoritmo nó intitulado “odom-robodeck” que publica mensagens do tipo *odometry* em um tópico “odometry” no ROS. Mensagens do tipo *odometry* informa com mais detalhes a posição e velocidade do robô, incluindo velocidade linear, angular e posição *x,y* do robô no plano cartesiano. A Figura 8 exemplifica os dados informados em um tópico com o tipo de mensagem *odometry*. O nó “odom-robodeck” lê informações dos tópicos “/distance” e “/angle” e, através de análises geométricas, converte as informações desses tópicos em informações do tipo *odometry*. Na figura 9, é apresentado o esquema de comunicação entre o nó “odom-robodeck” e o nó “pub-robodeck”. O nó “robodeck” inicializa todas os comandos do RoboDeck para serem chamados via serviços do ROS. Este procedimento de criação do nó “odom-robodeck” foi realizado para que futuras aplicações não relacionadas ao RoboDeck precisem acessar somente o tópico “odometry” para usufruir de sua posição e orientação.

Figura 8 – Mensagens publicadas no nó de odometria



O sistema odométrico implementado necessita de constante comunicação com o robô para cálculos da distância percorrida. Desta forma, sabe-se que o valor requisi-

Figura 9 – Mensagens publicadas no nó de odometria.



tado para locomoção do robô não será o mesmo de sua locomoção real. A Tabela 1 apresenta o resultado de um teste controlado de movimentação linear do robô com 10 repetições. O comando enviado ao robô representa a distância em centímetros que o robô deve se movimentar. A distância percorrida (DPe) é a distância medida (usando uma fita métrica) que o robô andou e a distância estimada (DE) é a distância que a odometria informa que o robô andou.

Tabela 1 – Distâncias percorridas pelo robô para 100 cm.

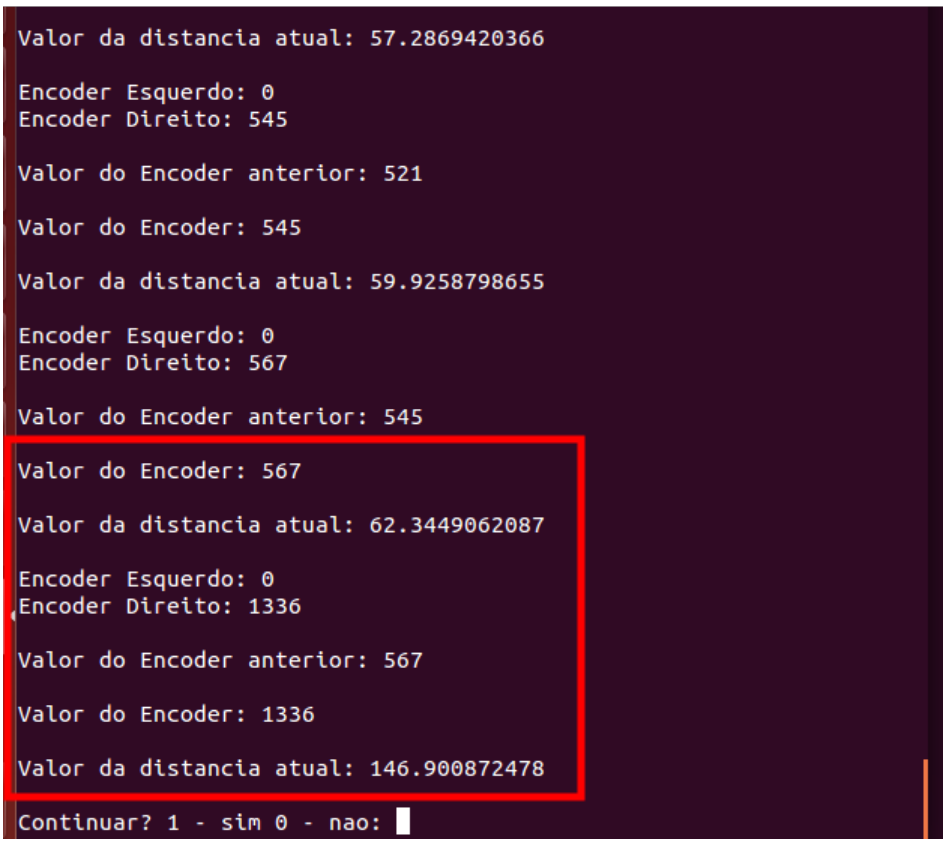
Comando enviado ao robô	Distância percorrida (DPe)(cm)	Distância estimada (DE)(cm)
100	84,0	101,5991
100	81,5	101,5983
100	88,0	109,6259
100	84,5	100,9394
100	87,7	106,9869
100	87,5	105,5575
100	83,8	102,9186
100	83,4	101,1593
100	85,5	102,3688
100	88,5	102,5887
Média $\pm$ DP	85,44 $\pm$ 2,37	103,53 $\pm$ 2,90

Analisando a Tabela 1, observa-se que existe um erro de odometria. A distância percorrida foi, neste experimento, sempre menor que a distância estimada pela odometria. Quando se trata de exploração autônoma para um robô móvel, faz-se necessário a modelagem do erro da odometria para correção da posição e orientação do robô no mapa. Para que este erro seja modelado, são necessários testes controlados, maior quantidade de dados para experimento e análises sobre esses dados. A Seção 4 apresenta este estudo sobre o erro do movimento linear do RoboDeck.

Notou-se, após diversas utilizações, que podem ocorrer falhas na comunicação entre o robô e a máquina operante do ROS, resultando no atraso de envio e recebimento das mensagens trocadas entres os dois dispositivos. Por esse motivo, existem diferenças significativas entre os resultados esperados e o recebidos ao enviar um comando de movimento ao robô. A Figura 10 mostra, em destaque, a diferença entre dois valores da distância calculados entre uma leitura dos *encoders* e a outra leitura

depois de um atraso de comunicação. O valor esperado da distância era próximo de 100 (em centímetros), no entanto, o retornado foi de 146,90 cm.

Figura 10 – Falha na comunicação entre RoboDeck e ROS



3.2 Mapeamento

Neste trabalho, valores específicos de ângulos foram usados para rotacionar o robô e adquirir dados dos sensores ultrassônicos para o mapeamento. O ângulo definido para a rotação do robô foi de 30°. O objetivo do robô é mapear o ambiente em 360°, o que gera 48 dados (distâncias) dos sensores ultrassônicos para uma posição do robô (giro no próprio eixo).

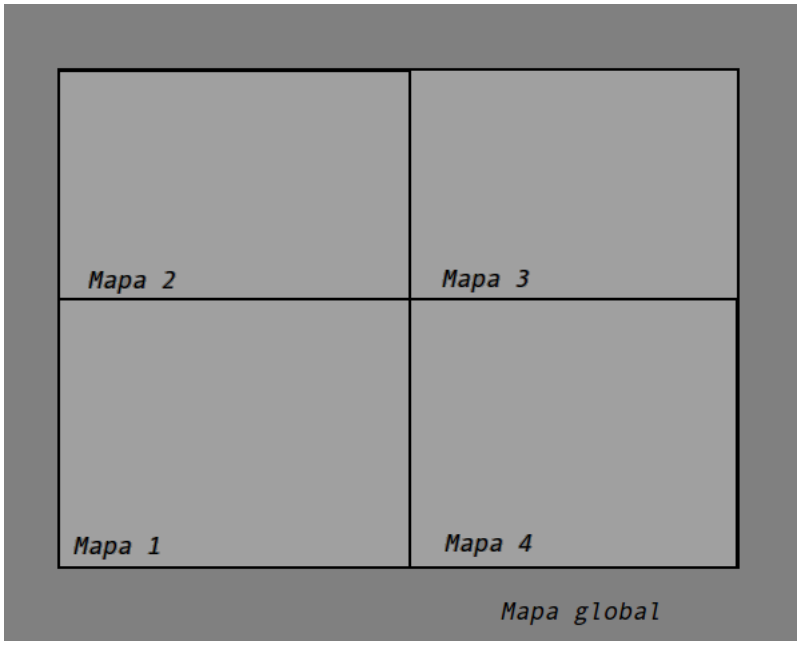
A Figura 11 detalha o ambiente físico utilizado para o mapeamento, possuindo dimensões de 270 cm x 264 cm x 45 cm (largura x profundidade x altura). O material utilizado como revestimento foi testado para verificar se há ou não interferência na leitura dos sensores, onde o mesmo não apresentou quaisquer problemas de leituras. Porém, há instabilidade nas medidas quando a distância do obstáculo é maior que 90 cm, onde o motivo ainda é alvo de estudo e sem explicação concreta. Por esse motivo, foram definidos que: (a) os dados confiáveis para o mapeamento deveriam possuir

valores maiores que 5 cm e menores que 90 cm e; (b) com a limitação dos sensores, o mapa global seria formado por mapas secundários.

Figura 11 – Ambiente fechado e controlado utilizado para mapeamento.

O algoritmo do mapa com grades de ocupação deste trabalho foi desenvolvido para receber uma matriz como entrada. A matriz contém as leituras dos sonares, cada sonar com seu ângulo em relação a posição inicial do robô e o valor da distância. Como já exposto, vários mapas secundários são construídos e, ao fim, um mapa global é gerado com a soma dos mapas secundários. As dimensões do mapa secundário são 150 cm x 150 cm (comprimento x largura), e as dimensões do mapa global são de 300 cm x 300 cm (comprimento por largura). Cada mapa secundário tem uma posição no plano cartesiano *(x, y)* e uma rotação θ. Os dados de posição e rotação de cada mapa secundário são utilizados para posicioná-los no mapa global. A Figura 12 apresenta a proposta explicada neste parágrafo.

Figura 12 – Proposta para criação do mapa global.

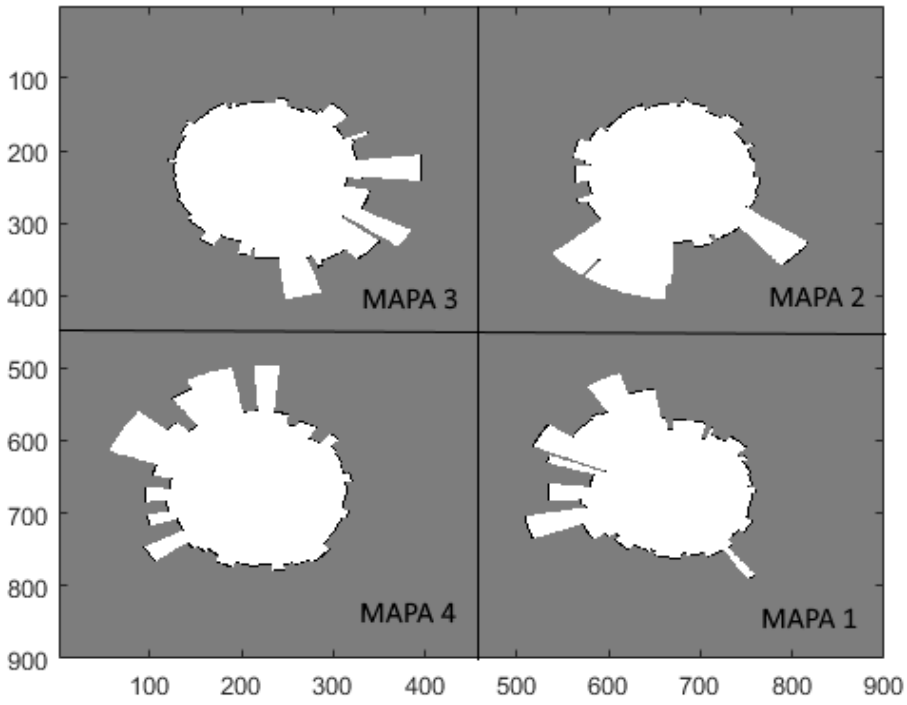


O procedimento do mapeamento do ambiente ocorreu em 9 etapas, onde cada etapa consistia em comandos enviados ao robô pelo ROS. O mapeamento total do ambiente consiste em 4 mapeamentos menores, onde o primeiro mapa começou pelo canto inferior direito (“Mapa 4”, tomando a primeira imagem da Figura 11 como referência), seguindo ao canto superior direito, canto superior esquerdo e, por fim, canto inferior esquerdo. Foram realizados comandos de mapeamento, movimento linear e rotação em determinado ângulo pelo robô:

1. Mapeamento;
2. Movimentação linear de 100 cm;
3. Mapeamento;
4. Rotação em 90°;
5. Movimentação linear de 100 cm;
6. Mapeamento;
7. Rotação em 90°;
8. Movimentação linear de 100 cm;
9. Mapeamento.

A Figura 13 ilustra os mapas secundários extraídos de cada extremidade do ambiente real. Os resultados deste experimento possuem erros de odometria, e seu propósito ainda não é gerar um mapa para navegação, mas mostrar que é possível seu desenvolvimento caso os erros sejam corrigíveis. São extraídas 48 medidas de distância por cada mapeamento, somando a quantidade de medidas extraídas de todos os sensores. Nota-se que as regiões melhores delimitadas são as distâncias do ambiente físico mais próximas do robô. À proporção que essas distâncias aumentam, há maior deformidade no mapa, com medidas incorretas e/ou inconsistentes. O modelo de representação do sonar e a quantidade de medidas extraídas por cada mapeamento colaboram para que o mapa possua um formato circular. Percebe-se que alguns ruídos se manifestaram nos mapas, mas há a possibilidade da utilização de algoritmos de filtragem, como o *Filtro de Kalman*, para melhorar a representação dos mapas, diminuindo ou eliminando ruídos e outras incertezas do ambiente.

Figura 13 – Mapas individuais de cada quadrante do ambiente.

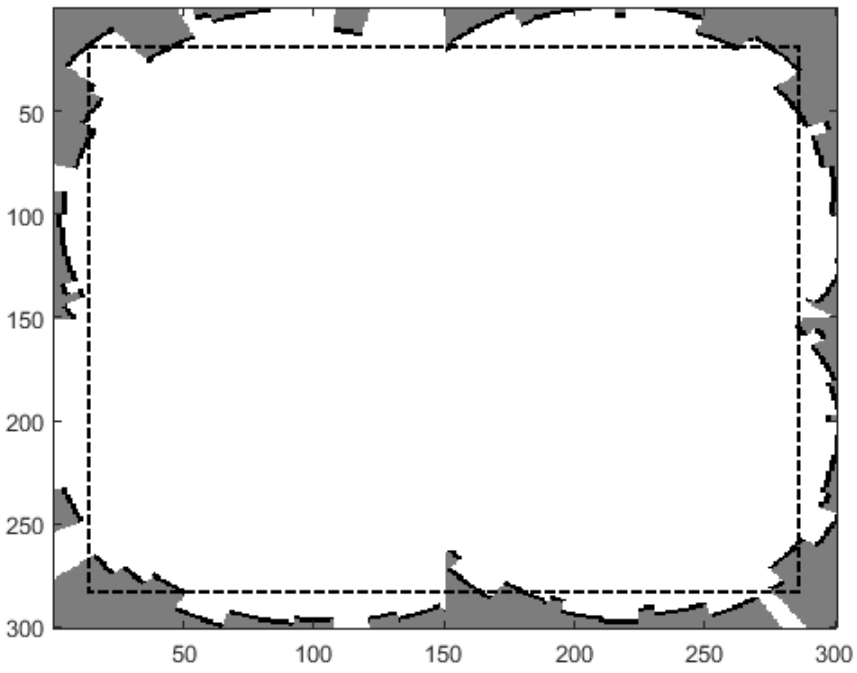


Por fim, a proposta da união dos mapas secundários para a formação de um mapa global é apresentada na Figura 14, onde a região tracejada representa as limitações do ambiente real. Percebe-se que o mapa possui padrões de um ambiente fechado, viabilizando a utilização dos sonares para mapeamento com o RoboDeck. Contudo, assim como os mapas secundários, o mapa global exibe um formato circular

em seus cantos.

Notam-se novamente os ruídos e as medidas incorretas que foram apresentados nos mapas secundários, entretanto, há uma diminuição desses erros no mapa global devido à seleção das regiões de interesse para a sua construção, ressaltando que há possibilidade da utilização de algoritmos de filtragem para melhorar a representação do mapa global.

Figura 14 – Mapa global do ambiente.



4 Modelagem do Erro da Odometria

Neste capítulo, é apresentada a modelagem do erro de movimento linear da odometria do RoboDeck. Para tanto, é necessário um conjunto de dados, obtidos a partir de experimentos controlados, para posterior análise. Em se tratando de erro de odometria, devem ser realizadas medições de distância percorrida pela robô e estimada pela odometria.

4.1 Definição do experimento

O experimento consiste de repetições de comandos de movimentação linear, com diferentes comandos de distância a ser percorrida pelo robô, para obtenção da distância percorrida real e a obtida pela odometria. Em cima desses valores, são calculados e analisados estatisticamente os erros.

Os comandos de distância definidos são: 50, 100, 150 e 200 centímetros; para cada comando, 10 amostras de distância devem ser adquiridas, sendo cada amostra um par de valores composto pela distância percorrida e pela distância estimada. São definidos 4 grupos, sendo eles: X050, X100, X150, X200. Um grupo é formado pelos valores dos erros absolutos (ver Equação 4.1) entre as distância percorridas e estimadas das amostras, fazendo que cada grupo possua 10 valores de erros.

O método ANOVA é um procedimento estatístico de análise de variância usado para determinar se existem diferenças estatisticamente significativas entre as médias de três ou mais grupos independentes (MCDONALD, 2009). As comparações feitas entre os grupos usam este método, seguidos por um teste post-hoc, neste caso o *Turkey HSD*.

Como resultado do experimento, obteve-se a Tabela 2, que exibe a média das distâncias dos 10 dados de cada grupo. A Figura 15 apresenta um comparativo entre a média das distâncias.

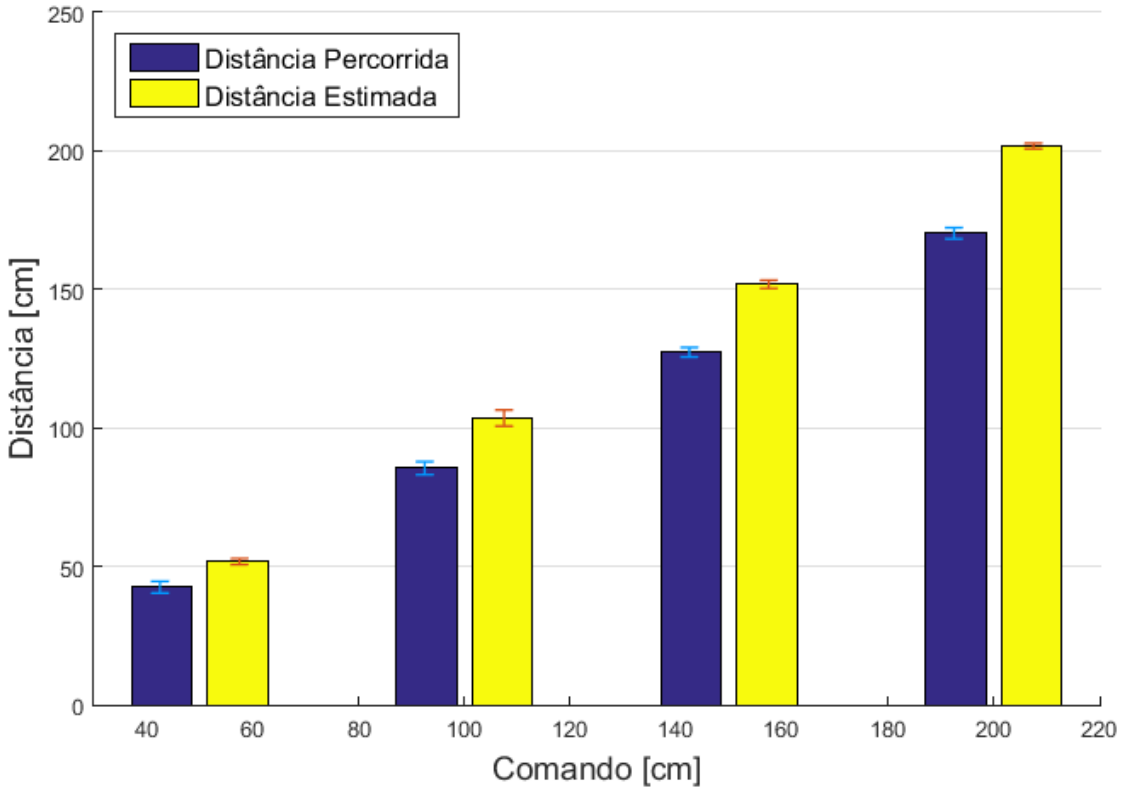
4.2 Calculo e Modelagem do Erro

O erro de cada amostra do experimento é calculado como apresentado na Equação 4.1, sendo esta a equação do erro absoluto. A distância percorrida pelo robô

Tabela 2 – Média dos valores de 10 experimentos para cada comando

Comando Enviado	DE $\pm$ DP (cm)	DPe $\pm$ DP (cm)	Erro absoluto $\pm$ DP (cm)
50	51,71 $\pm$ 1,10	42,45 $\pm$ 2,11	9,26 $\pm$ 2,27
100	103,53 $\pm$ 2,89	85,44 $\pm$ 2,37	18,09 $\pm$ 2,09
150	151,83 $\pm$ 1,49	127,3 $\pm$ 1,73	24,52 $\pm$ 1,39
200	201,75 $\pm$ 0,98	170.23 $\pm$ 2,00	31,51 $\pm$ 2,10

Figura 15 – Médias das distâncias exatas e aproximadas do robô.



é representada por d_p e d_e é a distância estimada pela odometria

$$erro = |d_p - d_e| \tag{4.1}$$

Percebe-se, analisando a Tabela 2, que a média dos erros cresce conforme o comando de distância enviado aumenta. A Figura 16 apresenta visualmente o crescimento deste erro e a Tabela 3 apresenta análise de variância para esses erros, onde valor de

Tabela 3 – Análise de variância (ANOVA) entre as médias dos erros

	Df	*Sum* Sq	*Mean* Sq	Valor F	Pr
Grupos	3	2507,7	835,9	232,3	2e-16
Resíduos	32	115,2	3,6	-	-

Pr menores ou iguais a 0,05 indicam que as médias comparadas são estatisticamente diferentes. A Tabela 4 exibe uma análise de post-hoc entre pares dos grupos grupos, e os valores de p menores que 0,05 indicam que dois grupos são estatisticamente diferentes. A Figura 17 mostra graficamente a distribuição das médias dos grupos.

Tabela 4 – Análise de *Post-hoc* entre as médias dos erros

Grupos	Diferença	Menor	Maior	p
X150-X100	6,767302	4,344427	9,190176	1e-07
X200-X100	13,757244	11,334369	16,180118	0e+00
X050-X100	-8,821235	-11,24410	-6,398360	0e+00
X200-X150	6,989942	4,567067	9,412816	0e+00
X050-X150	-15,58853	-18,01141	-13,1656	0e+00
X050-X200	-22,57847	-25,00135	-20,15560	0e+00

Figura 16 – Média do Erro Absoluto entre Distâncias Percorridas e Estimadas.

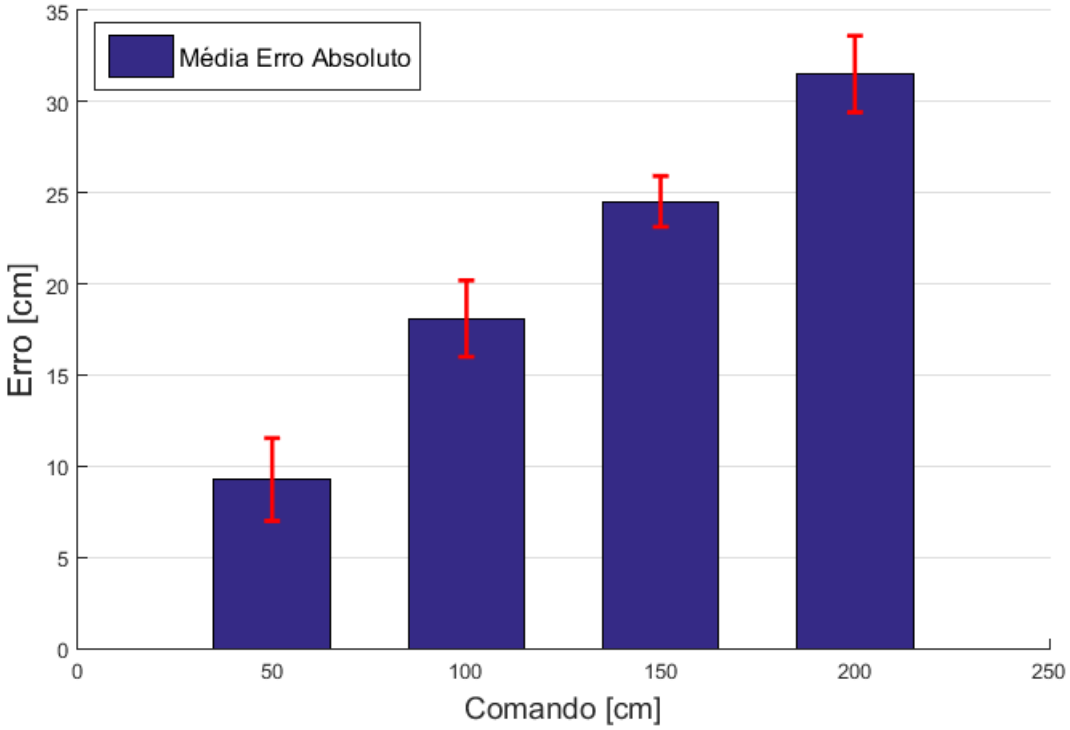
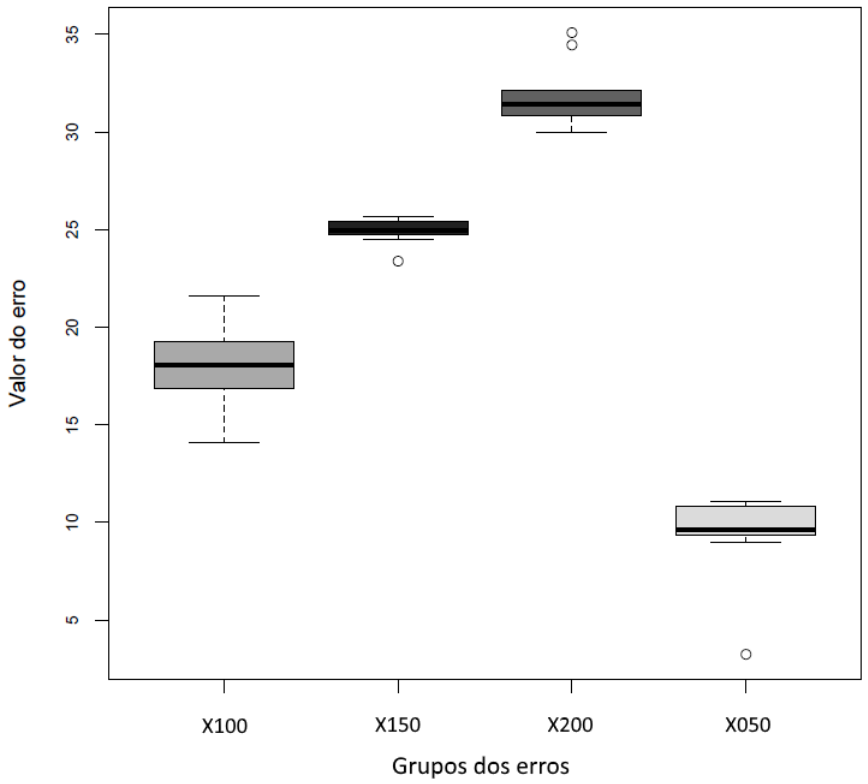


Figura 17 – Gráficos de barras dos Erros Absolutos das amostras.

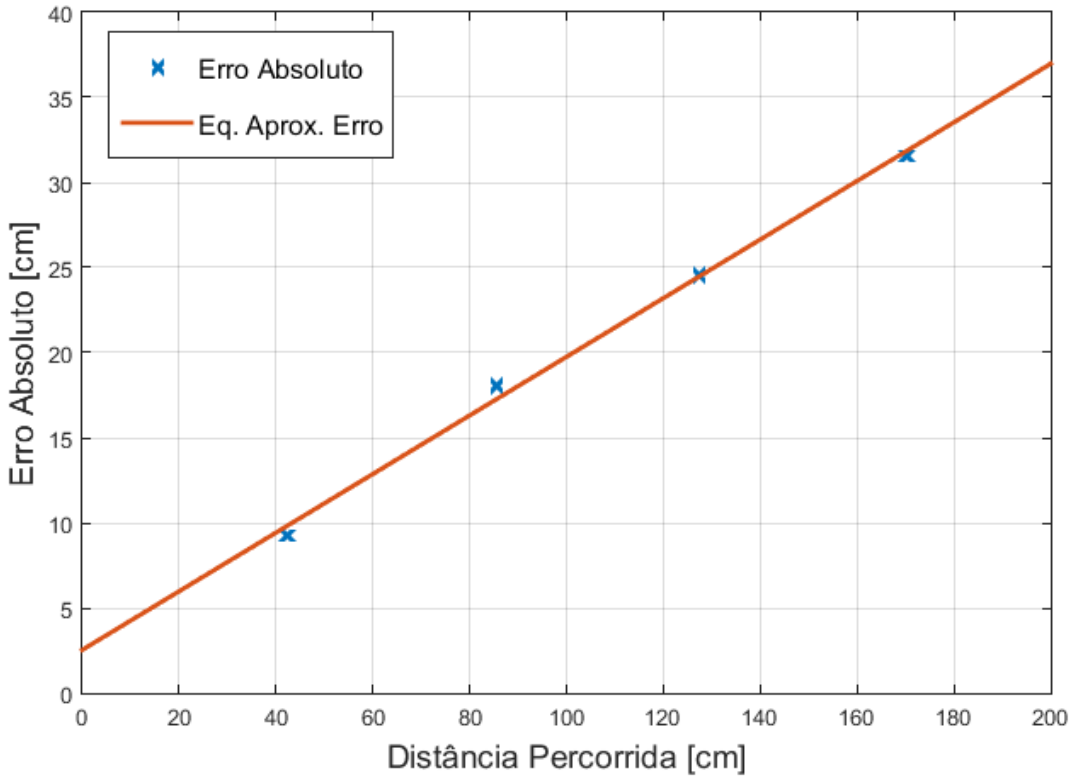


A distância aproximada da odometria é maior que a distância percorrida pelo robô, assim, para que essas distâncias se igualem, é necessário que um ajuste seja adicionado à igualdade das distâncias. Dado d como a distância percorrida pelo robô, d_{odom}, a distância estimada pela odometria e um ϵ como um valor de erro, a distância percorrida pelo robô apresenta-se na equação 4.2. Dois fatores são analisados: (1) O comando enviado ao robô (representado por d_{odom}) deve ser equiparada à distância percorrida pelo robô, portanto, para que isto seja verdadeiro, d_{odom} deverá ser maior que d e; (2) o erro ϵ deve estar em função de d, pois d é a única entrada para se calcular o valor de d_{odom}.

$$d = d_{odom} - \epsilon \tag{4.2}$$

Para efetuar o calculo do erro em função de d, as médias dos erros foram projetadas em um gráfico para análise de comportamento, como ilustra a Figura 18. Após análise e por apresentar uma tendência linear de crescimento, aplicou-se o método matemático de regressão linear aos pontos, que representam as médias dos erros. Assim, encontrou-se uma equação linear em função da distância percorrida d, cujo termos são exibidos na Equação 4.3.

Figura 18 – Média do Erro Absoluto entre Distâncias Percorridas e Estimadas.



$$f_{erro}(d) = 0.1722 \cdot d + 2.5378 \tag{4.3}$$

Reorganizando os fatores e colocando todos os termos em função de d, o termo d_{odom} é calculado seguindo a Equação 4.4. Esta Equação calcula um d_{odom} e $f_{erro}(d)$ para qualquer distância d desejada para movimentação linear do robô.

$$d_{odom}(d) = d + f_{erro}(d) \tag{4.4}$$

4.3 Teste de Correção do Erro

Após a implementação da Equação 4.4 à odometria do RoboDeck, um novo experimento foi realizado para validação da proposta. Assim como o experimento anterior, 10 medidas para cada comando discretizados em 50, 100, 150 e 200 cm são obtidas para analise. A Tabela 5 expõe os novos valores das médias das distâncias percorridas e estimadas, sendo comparadas por gráfico de barras na Figura 19.

Os novos erros calculados são apresentados também na Tabela 5 e ilustrados na Figura 20. A Tabela 6 faz uma comparação estatística usando o teste t pareado entre a média dos erros antes e depois da implementação da Equação 4.4. O teste t pareado é usando quando se necessita comparar duas medidas ou valores. O cenário mais comum é que o teste t pareado analise o “antes” e “depois” de algum tratamento (MCDONALD, 2009). Com o valor de p menor que 0,05, o teste expõe que os valores dos erros antes e depois não são iguais e que os erros depois do experimento diminuíram em relação aos anteriores.

Tabela 5 – Medidas das distâncias com 10 experimentos para cada comando

Comando Enviado	DE $\pm$ DP (cm)	DPe $\pm$ DP (cm)	Erro Absoluto $\pm$ DP (cm)
50	51,91 $\pm$ 1,35	49,58 $\pm$ 0,98	2,33 $\pm$ 1,13
100	101,65 $\pm$ 1,31	98,03 $\pm$ 1,09	3,62 $\pm$ 1,57
150	151,78 $\pm$ 1,35	144,06 $\pm$ 3,14	7,72 $\pm$ 4,05
200	204,29 $\pm$ 3,75	195,69 $\pm$ 1,78	8,60 $\pm$ 3,59

Figura 19 – Médias das distâncias percorridas e estimadas do robô.

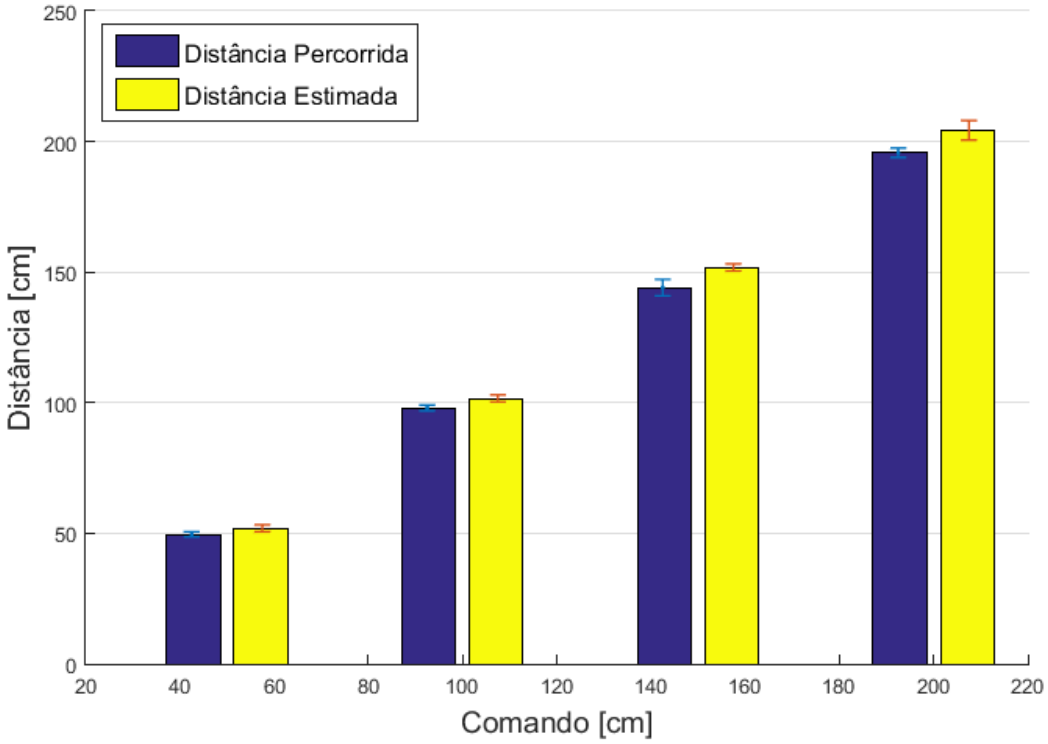


Figura 20 – Média do erros após correção.

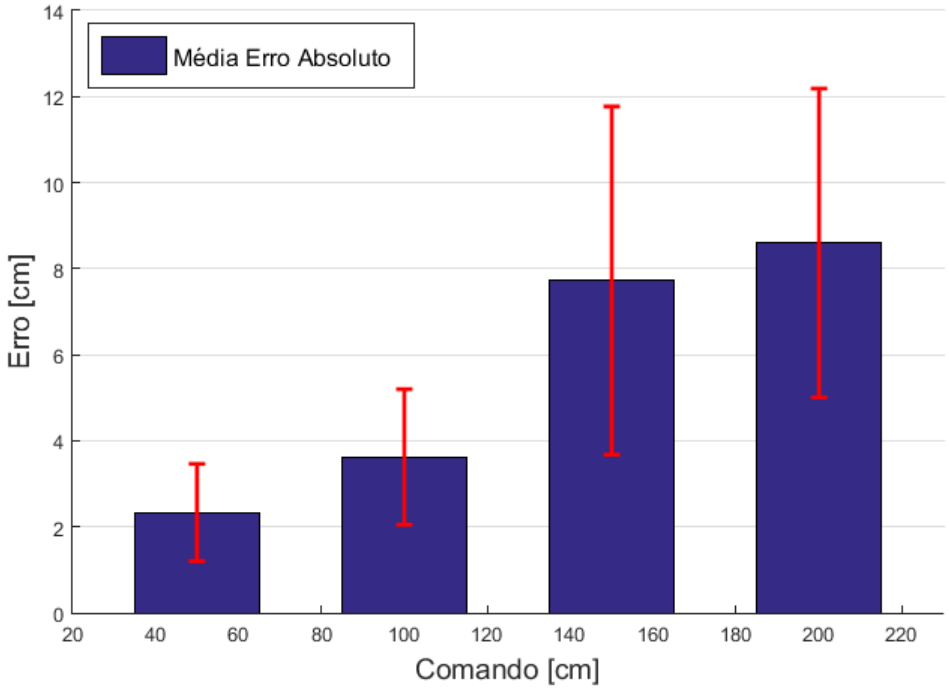


Tabela 6 – Teste t pareado entre os erros absolutos

TESTE T PAREADO PRE x POS			
d	t	df	p
50	59,71	9	5,223e-13
100	14,998	9	1,13e-07
150	13,31	9	3,171e-07
200	20,094	9	8,713e-09

Por fim, é realizada uma nova avaliação da média do erro usando análise de variância, como mostrada na Tabela 7. Analisando os resultados, nota-se pelo valor de Pr que existem diferenças estatísticas nas médias dos erros dos grupos. A análise de post-hoc na Tabela 8, o valor de p para os grupos X100 e X050 e entre os grupos X200 e X150 não são estatisticamente diferentes, o que corrobora com a hipótese que o erro corrigido tende a não depender mais da distância a ser percorrida. A Figura 21 mostra graficamente a distribuição das médias dos grupos.

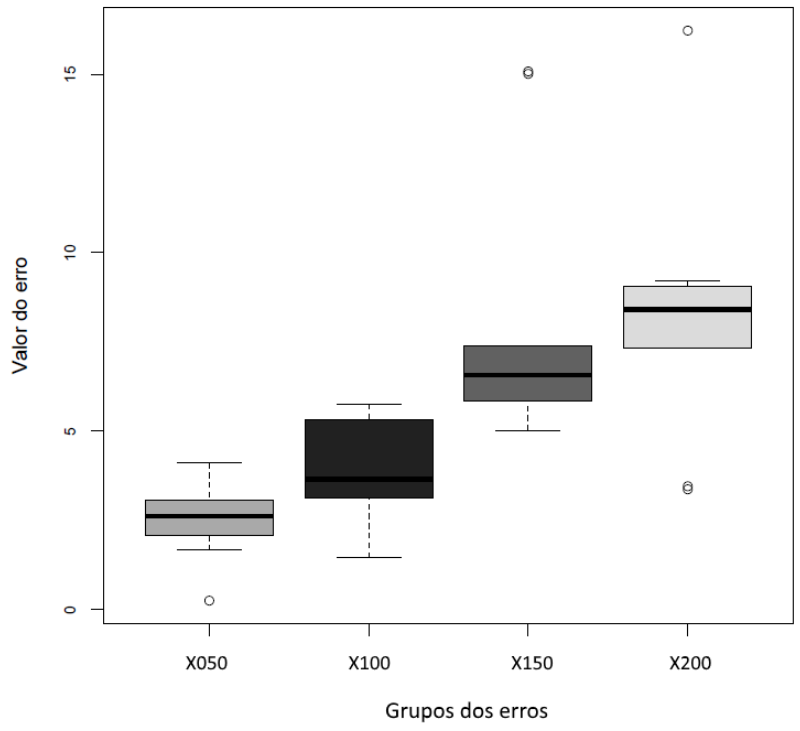
Tabela 7 – Análise de variância (ANOVA) entre as médias dos erros

	Df	*Sum* Sq	*Mean* Sq	Valor F	Pr
Grupos	3	238,7	79,56	9,694	0,000106
Resíduos	32	262,6	8,21	-	-

Tabela 8 – Teste *Post-hoc* entre os erros após correção

Grupos	Diferença	Menor	Maior	p-adj
X100-X050	1,41320063	-2,245776	5,072177	0,7237623
X150-X050	5,76954281	2,1105661	9,428520	0,0008909
X200-X050	5,74726301	2,0882863	9,406240	0,0009329
X150-X100	4,35634218	0,6973654	8,015319	0,0145942
X200-X100	4,33406239	0,6750856	7,993039	0,0152089
X200-X150	0,02227979	-3,681256	3,636697	0,9999984

Figura 21 – Gráficos de barras dos Erros Absolutos das amostras após correção



5 Conclusão

Exploração autônoma não é uma tarefa trivial quando esta é destinada a robôs móveis, que necessitam integrar o sistema de localização e construção do mapa do ambiente com um processo cognitivo capaz de guiar o robô usando as informações incrementalmente colhidas durante a exploração. Este trabalho apresenta um estudo de caso sobre a plataforma robótica RoboDeck visando desenvolver rotinas computacionais específicas necessárias para sua utilização na tarefa de exploração autônoma, com suporte do Sistema Operacional Robótico (*Robot Operating System* – ROS) e sensores de baixo custo.

A arquitetura implementada através do ROS permite a substituição de sensores ou mesmo do kit robótico inteiro sem a necessidade de rescrita das rotinas. Experimentos são apresentados, nos quais mapas de um ambiente fechado e controlado são gerados satisfatoriamente a partir de sensores ultrassônicos de baixo custo que equipam a plataforma.

Os algoritmos de odometria desenvolvidos para o robô contribuem para futuras aplicações de localização através do ROS, pois os dados da odometria estão adequados ao mesmo tipo de dados de outras aplicações do sistema operacional robótico.

Os mapas produzidos apresentaram instabilidades à proporção que as distâncias envolvidas entres o robô e os obstáculos se tornaram maiores. Concluiu-se que não seria recomendado utilizar os sensores ultrassônicos para exploração como única fonte de visão computacional. Deste modo, o recomendado seria utilizar outros sensores para se obter outras informações do ambiente de forma a construir mapas mais robustos.

A odometria do robô apresentou erros significativos quando testes de movimento linear foram realizados. Os resultados do teste Análise de Variância (ANOVA) mostraram que a média dos erros varia de acordo com a distância para os primeiros erros calculados. Após realizada a modelagem do erro implementada ao algoritmo de odometria no intuito de verificar se há diminuição dos mesmos, os novos testes do ANOVA exibiram que não existiram diferenças entre os dois primeiros os dois últimos grupos.

5.1 Trabalhos Futuros

Visando futuras aplicações de exploração autônoma ao RoboDeck, um estudo mais aprofundado sobre os erros da odometria para comandos de movimento linear são cabíveis, além do estudo sobre os erros para comandos de movimento rotacional do robô.

Sugere-se a utilização de outros sensores presentes no RoboDeck, como a câmera do robô, para auxílio tanto na construção e aperfeiçoamento do mapa quanto em um sistema de localização global, pois esses sistemas são bem usados para correção de erros de odometria.

Referências

ARAUJO, E.; GRUPEN, R. A. Feature detection and identification using a sonar-array. In: IEEE. *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*. [S.l.], 1998. v. 2, p. 1584–1589. Citado na página 21.

BANK, D.; KAMPKE, T. High-resolution ultrasonic environment imaging. *IEEE transactions on robotics*, IEEE, v. 23, n. 2, p. 370–381, 2007. Citado na página 21.

BEZERRA, C. G.; ALSINA, P. J.; MEDEIROS, A. A. dantas de. *Um sistema de localizaçao para um robô móvel baseado em odometria e marcos naturais*. Tese (Doutorado), 2004. Citado na página 13.

BURGARD, W.; FOX, D.; JANS, H.; MATENAR, C.; THRUN, S. Sonar-based mapping with mobile robots using em. In: MORGAN KAUFMANN PUBLISHERS, INC. *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-*. [S.l.], 1999. p. 67–76. Citado na página 14.

CHOSET, H.; NAGATANI, K.; LAZAR, N. A. The arc-transversal median algorithm: a geometric approach to increasing ultrasonic sensor azimuth accuracy. *IEEE Transactions on Robotics and Automation*, IEEE, v. 19, n. 3, p. 513–521, 2003. Citado na página 22.

DUDEK, G.; JENKIN, M. *Computational principles of mobile robotics*. [S.l.]: Cambridge university press, 2010. Citado na página 21.

ELFES, A. Using occupancy grids for mobile robot perception and navigation. *Computer*, IEEE, v. 22, n. 6, p. 46–57, 1989. Citado na página 24.

GASHU, T. Y. *Integração do sistema operacional ROS com o módulo de alta performance Robodeck*. São José dos Campos, SP: [s.n.], 2014. Citado 3 vezes nas páginas 14, 19 e 20.

GUIVANT, J.; NEBOT, E.; BAIKER, S. Autonomous navigation and map building using laser range sensors in outdoor applications. *Journal of robotic systems*, Citeseer, v. 17, n. 10, p. 565–583, 2000. Citado na página 13.

HEINEN, F. J.; OSÓRIO, F. Sistema de controle híbrido para robôs móveis autônomos. *Unisinos–PIPCA–Dissertação de Mestrado em Computação Aplicada*, 2002. Citado na página 21.

JUANG, J. G.; WANG, J. A. Indoor map building by laser sensor and positioning algorithms. In: TRANS TECH PUBL. *Applied Mechanics and Materials*. [S.l.], 2015. v. 764, p. 752–756. Citado na página 13.

KLEEMAN, L.; KUC, R. Sonar sensing. In: *Springer Handbook of Robotics*. [S.l.]: Springer, 2008. p. 491–519. Citado 2 vezes nas páginas 21 e 22.

LEE, K.; CHUNG, W. K. Effective maximum likelihood grid map with conflict evaluation filter using sonar sensors. *IEEE Transactions on Robotics*, IEEE, v. 25, n. 4, p. 887–901, 2009. Citado 2 vezes nas páginas 21 e 22.

MCDONALD, J. H. *Handbook of biological statistics*. [S.l.]: Sparky House Publishing Baltimore, MD, 2009. v. 2. Citado 2 vezes nas páginas 35 e 40.

MUñOZ, M. E. de S. *Projeto de Software RobotDeck versão 0.2*. [S.l.], 2011. Citado 2 vezes nas páginas 17 e 18.

OGISO, S.; KAWAGISHI, T.; MIZUTANI, K.; WAKATSUKI, N.; ZEMPO, K. Self-localization method for mobile robot using acoustic beacons. *ROBOMECH Journal*, Springer, v. 2, n. 1, p. 1–12, 2015. Citado na página 14.

O'KANE, J. M. *A gentle introduction to ROS*. [S.l.]: Jason M. O'Kane, 2014. Citado na página 18.

ROS Concepts. 2014. <http://wiki.ros.org/ROS/Concepts>. Citado 2 vezes nas páginas 18 e 19.

ROS Introduction. 2014. <http://wiki.ros.org/ROS/Introduction>. Citado 2 vezes nas páginas 14 e 18.

ROS Java. 2013. <https://github.com/rosjava/rosjava_core>. Citado na página 20.

SECCHI, H. A. Uma introdução aos robôs móveis. *Instituto de Automática–INAUT. Universidade Nacional de San Juan–UNSJ–Argentina*, 2012. Citado na página 21.

TARDÓS, J. D.; NEIRA, J.; NEWMAN, P. M.; LEONARD, J. J. Robust mapping and localization in indoor environments using sonar data. *The International Journal of Robotics Research*, SAGE Publications, v. 21, n. 4, p. 311–330, 2002. Citado na página 14.

THRUN, S.; BURGARD, W.; FOX, D. *Probabilistic robotics*. [S.l.]: MIT press, 2005. Citado na página 23.

VARVEROPOULOS, V. Robot localization and map construction using sonar data. *The Rossum Project*, v. 10, p. 1–10, 2005. Citado 2 vezes nas páginas 13 e 22.

WOLF, D. F.; SIMÕES, E. do V.; OSÓRIO, F. S.; JUNIOR, O. T. Robótica móvel inteligente: Da simulação às aplicações no mundo real. In: *Mini-Curso: Jornada de Atualização em Informática (JAI), Congresso da SBC*. [S.l.: s.n.], 2009. p. 13. Citado na página 13.

XBOT. Apostila de software do robodeck versão 1.1. [S.l.], 2011. Citado 2 vezes nas páginas 14 e 16.

YENILMEZ, L.; TEMELTAS, H. Map building for mobile robots by ultrasonic and infrared sensor data fusion. In: INTERNATIONAL SOCIETY FOR OPTICS AND PHOTONICS. *Optomechatronic Systems III*. [S.l.], 2002. p. 444–449. Citado na página 13.

ZAKI, J. A.; ARAFA, O. Mobile robot positioning using odometry and ultrasonic sensors. *Journal of Cybernetics and Informatics*, Slovak Society for Cybernetics and Informatics, v. 13, 2012. Citado 3 vezes nas páginas 21, 22 e 23.

ZANOLLA, L.; SOUSA, D. R. de; FURLANETO, R. M.; BOTELHO, W. T.; MARIETTO, M. d. G. B. Implementação com validação real de um controle proporcional, integral e derivativo na plataforma robótica robodeck. *Simpósio de Tecnologia da Informação*, V, 2014. Citado na página 17.