

**UNIVERSIDADE DE CAXIAS DO SUL – UCS
CAMPUS UNIVERSITÁRIO DA REGIÃO DOS VINHEDOS – CARVI
ÁREA DO CONHECIMENTO DE CIÊNCIAS EXATAS E ENGENHARIA
CURSO DE ENGENHARIA ELÉTRICA**

MATEUS KASMIRSKI

**SISTEMA DE NAVEGAÇÃO AUTÔNOMA DE CADEIRA DE RODAS
MOTORIZADA EM AMBIENTE CONHECIDO**

BENTO GONÇALVES

2021

MATEUS KASMIRSKI

**SISTEMA DE NAVEGAÇÃO AUTÔNOMA DE CADEIRA DE RODAS
MOTORIZADA EM AMBIENTE CONHECIDO**

Trabalho de Conclusão de Curso I
apresentado no Campus Universitário da
Região dos Vinhedos, da Universidade de
Caxias do Sul, como requisito parcial para
obtenção do título de Engenheiro
Eletricista.

Orientador: Prof. Me. Patric Janner
Marques

BENTO GONÇALVES

2021

MATEUS KASMIRSKI

**SISTEMA DE NAVEGAÇÃO AUTÔNOMA DE CADEIRA DE RODAS
MOTORIZADA EM AMBIENTE CONHECIDO**

Trabalho de Conclusão de Curso I
apresentado no Campus Universitário da
Região dos Vinhedos, da Universidade de
Caxias do Sul, como requisito parcial para
obtenção do título de Engenheiro
Eletricista.

Orientador: Prof. Me. Patric Janner
Marques

Aprovado em ____/____/____

Banca Examinadora

Prof. Me. Patric Janner Marques
Universidade de Caxias do Sul – UCS

Prof. Dra. Marilda Machado Spindola
Universidade de Caxias do Sul – UCS

Prof. Me. Patricia Giacomelli
Universidade de Caxias do Sul – UCS

RESUMO

Existe um grande número de pessoas que enfrentam dificuldades em seu cotidiano devido às limitações físicas, fato que motiva diversas pesquisas voltadas às tecnologias assistivas, que são recursos utilizados para ampliar as habilidades funcionais de pessoas com deficiência, em prol de uma melhora na qualidade de vida. Diante deste contexto, este trabalho apresenta um sistema de navegação autônoma de uma cadeira de rodas entre corredores e portas em ambiente interno conhecido. O sistema utiliza o algoritmo de Dijkstra para gerar uma trajetória por meio de um mapa de grade do ambiente e do destino selecionado pelo usuário através de uma interface no notebook. A posição relativa da cadeira durante a trajetória é estimada por meio da odometria, fazendo uso de *encoders* nas rodas, acelerômetro e giroscópio. Complementar à odometria, um sistema de localização absoluta por marcos artificiais é empregado, no qual utiliza imagens RGB-D capturadas por uma câmera Kinect v2 para determinar a posição da cadeira por meio da localização de retângulos colocados no ambiente em posições previamente conhecidas. A localização dos retângulos é realizada aplicando o detector de bordas de Canny e a transformada probabilística de Hough para extrair retas nas imagens RGB, e por meio das informações de profundidade da imagem são verificadas quais retas são pertencentes ao marco. As imagens de profundidade capturadas pelo Kinect também são utilizadas para o processo de determinação da posição da cadeira em relação ao centro da porta a ser transpassada. A detecção da porta consiste em localizar suas bordas por meio da variação da profundidade existente entre a parede e seu vão. O controle da trajetória da cadeira no deslocamento entre corredores e na passagem por portas é realizado utilizando controladores baseados em lógica *fuzzy* que atuam na velocidade linear e angular da cadeira. Após a realização dos ensaios, observou-se indicativos para ser um forte potencial de pesquisas na aplicação da odometria juntamente com o Kinect na navegação autônoma de robôs móveis. Entretanto, a localização absoluta por marcos artificiais durante o deslocamento ainda se apresenta pouco confiável, uma vez que há uma falta de sincronismo entre a captura das imagens RGB de profundidade.

Palavras-chave: Cadeiras de Rodas. Navegação Autônoma. Odometria. Kinect. Localização absoluta.

ABSTRACT

There is a large number of people who are facing difficulties in their daily lives, due to physical limitations, this reality motivates several researches focused on assistive technologies, which are resources used to expand functional abilities of people with disability, promoting better quality of life. In this context, this work presents an autonomous navigation system of a wheelchair between corridors and doors in a known indoor environment. This system uses a Dijkstra algorithm to generate a trajectory from a grid map of the place, and the destination selected by the user through a laptop interface. The relative position of the wheelchair during the trajectory is estimated through Odometry using encoders on the wheels, accelerometer and gyroscope. In addition of Odometry, an absolute location system, by artificial landmarks, is applied, which uses RGB-D images captured by a Kinect v2 camera, to determine the position of the wheelchair by locating rectangles placed in the environment in previously known positions. The location of the rectangles is performed by applying the Canny edge detector and the probabilistic Hough transform to extract lines in RGB images, and through the information of image depth, it is verified which lines belong to the landmark. The depth images captured by Kinect are also used for the process of determining the position of the wheelchair in relation to the center of the door to be passed. The depth images captured by Kinect are also used for the process of determining the position of the wheelchair in relation to the center of the door to be passed. The door detection consists of locating its edges through the variation of the depth between the wall and its gap. The control of the wheelchair's trajectory in the dislocation between corridors, and in the passage through doors it is performed using controllers, based on fuzzy logic that act on the linear and angular velocity of the wheelchair. After carrying out the tests, it was observed that there is a strong potential for research in the application of odometry together with Kinect in the autonomous navigation of mobile robots. However, the absolute location by artificial landmarks during the dislocation still performs unreliable, once that there are a lack of synchronism between the capturing the RGB images of depth.

Keywords: Wheelchair. Autonomous Navigation. Odometry. Kinect. Absolute Location.

LISTA DE FIGURAS

Figura 1 - Deslocamento do robô utilizando odometria	17
Figura 2 - Modelo cinemático de um robô com acionamento diferencial.....	18
Figura 3 - Movimento infinitesimal do robô.....	19
Figura 4 - Correção do erro da odometria por meio da localização absoluta	21
Figura 5 - Trajetória característica de um robô com erros tipo A.....	23
Figura 6 - Trajetória característica de um robô com erros tipo B.....	24
Figura 7 - Relações geométricas de deslocamento	25
Figura 8 - Formação de sombras no sensor de profundidade de luz estruturada	28
Figura 9 - Comportamento da luz em superfícies de diferentes matérias	28
Figura 10 - Funcionamento do sensor ToF	29
Figura 11 - Especificações de hardware dos modelos do sensor Kinect	30
Figura 12 - Exemplo de mapa discreto e contínuo	31
Figura 13 - Exemplo de mapa de grade	32
Figura 14 - Resultado do algoritmo Dijkstra	32
Figura 15 - Exemplo de função de pertinência para estatura	34
Figura 16 - Sistema de inferência fuzzy	35
Figura 17 - Representação de uma reta.....	36
Figura 18 - Transformada de Hough para retas	37
Figura 19 - Resultado da aplicação do detector de bordas de Canny.....	39
Figura 20 - Configuração experimental para avaliar o ruído axial e lateral	40
Figura 21 - Ruído axial em função da distância e do ângulo da superfície	41
Figura 22 - Ruído lateral em função da distância e ângulo da superfície.....	42
Figura 23 - Porta com um ponto de descontinuidade.....	44
Figura 24 - Fluxograma do sistema de navegação autônoma implementado	49
Figura 25 - Integração dos componentes	51
Figura 26 - Velocidade linear ao aplicar um degrau	52
Figura 27 - Resposta da velocidade linear a um degrau	54
Figura 28 - Resposta da velocidade angular para esquerda a um degrau.....	55
Figura 29 - Resposta da velocidade angular para direita a um degrau	55
Figura 30 - Interface de seleção do destino	57
Figura 31 - Exemplo de um mapa de grade	57
Figura 32 - Sistema de coordenadas do controle de trajetória.....	58

Figura 33 - Funções de pertinência para a posição y.....	60
Figura 34 - Funções de pertinência para a velocidade linear	61
Figura 35 - Resposta do controlador de velocidade linear	62
Figura 36 - Funções de pertinência para a orientação	62
Figura 37 - Funções de pertinência para a velocidade angular.....	63
Figura 38 - Resposta do controlador de velocidade linear	64
Figura 39 - Medidas obtidas na primeira etapa do ensaio UMBmark.....	67
Figura 40 - Medidas obtidas na segunda etapa do ensaio UMBmark.....	68
Figura 41 - Exemplo de imagem RGB e de profundidade.....	70
Figura 42 - Imagem após limiarização	71
Figura 43 - Imagem após aplicar o detector de borda.....	72
Figura 44 - Retas encontrada pela transformada de Hough	72
Figura 45 - Retas do marco encontradas	74
Figura 46 - Sistema de coordenadas de localização do marco.....	74
Figura 47 - Medidas no centro da imagem de profundidade de uma porta	77
Figura 48 - Retas traçadas entre pares de bordas	78
Figura 49 - Porta com um único ponto de descontinuidade	78
Figura 50 - Reconhecimento da porta com um ponto de descontinuidade	79
Figura 51 - Exemplo de porta localizada.....	80
Figura 52 - Erro da estimativa da orientação em relação à porta.....	80
Figura 53 - Simulação de trajetória	82
Figura 54 - Resultados do controle de velocidade linear.....	84
Figura 55 - Resultados do controle de velocidade angular	84
Figura 56 - Ensaio de desaceleração da cadeira	85
Figura 57 - Erro das medidas de profundidade sem filtro.....	86
Figura 58 - Ruído das medidas de profundidade sem filtro.....	87
Figura 59 - Erro das medidas de profundidade com filtro de Fankhauser	88
Figura 60 - Ruído das medidas de profundidade com filtro de Fankhauser	88
Figura 61 - Erro das medidas de profundidade com filtro gaussiano	89
Figura 62 - Ruído das medidas de profundidade com filtro gaussiano.....	89
Figura 63 - Porta localizada por um ponto de descontinuidade.....	96
Figura 64 - Trajetória posição 9.....	99
Figura 65 - Trajetória posição 4.....	100
Figura 66 - Trajetória posição 8.....	101

Figura 67 - Trajetória posição 6.....	102
Figura 68 - Trajetória posição 12.....	103
Figura 69 - Trajetória posição 2.....	104
Figura 70 - Trajetória posição 7.....	105
Figura 71 - Sequência de imagens RGB-D durante movimento angular.....	107
Figura 72 - Trajetória no ambiente no ensaio 1.....	108
Figura 73 - Trajetória posição 1.....	116
Figura 74 - Trajetória posição 3.....	117
Figura 75 - Trajetória posição 5.....	117
Figura 76 - Trajetória posição 10.....	118
Figura 77 - Trajetória posição 11.....	119
Figura 78 - Trajetória no ambiente no ensaio 2.....	120
Figura 79 - Trajetória no ambiente no ensaio 3.....	121
Figura 80 - Trajetória no ambiente no ensaio 4.....	122
Figura 81 - Trajetória no ambiente no ensaio 5.....	123

SUMÁRIO

1 INTRODUÇÃO	11
1.1 CONTEXTUALIZAÇÃO E JUSTIFICATIVA	11
1.2 OBJETIVO GERAL	12
1.3 OBJETIVOS ESPECÍFICOS	13
1.4 ESCOPO E RESTRIÇÕES	13
1.5 APRESENTAÇÃO DO TRABALHO	14
2 REVISÃO BIBLIOGRÁFICA	15
2.1 NAVEGAÇÃO	15
2.2 LOCALIZAÇÃO	16
2.2.1 Localização relativa: odometria	16
2.2.2 Calibração da odometria: UMBmark	21
2.2.3 Localização absoluta	26
2.2.5 Sensor Kinect	29
2.3 GERAÇÃO DE TRAJETÓRIA	30
2.4 CONTROLADOR FUZZY	33
2.5 PROCESSAMENTO DIGITAL DE IMAGENS	35
2.5.1 Transformada de Hough para retas	36
2.5.2 Detecção de bordas: Canny	38
2.2.4 Modelo de ruído do Kinect v2	40
3 TRABALHOS RELACIONADOS	43
4 DESENVOLVIMENTO DO TRABALHO	48
4.1 HARDWARE	50
4.2 PROJETO DO CONTROLE DE VELOCIDADE	51
4.3 GERAÇÃO DE TRAJETÓRIA	56
4.4 CONTROLE DE TRAJETÓRIA	58
4.5 ODOMETRIA	65

4.6 AQUISIÇÃO DE IMAGENS RGB-D	70
4.7 SISTEMA DE LOCALIZAÇÃO ABSOLUTA.	71
4.8 SISTEMA DE LOCALIZAÇÃO DE PORTAS.....	76
4.9 CONTROLE DA TRAJETÓRIA NA PASSAGEM POR PORTAS.....	81
5 RESULTADOS E DISCUSSÕES	83
5.1 TESTE DOS CONTROLADORES DE VELOCIDADE	83
5.2 APLICAÇÃO DE FILTRO DE RUÍDOS EM IMAGENS DE PROFUNDIDADE .	86
5.3 TESTE DO SISTEMA DE LOCALIZAÇÃO DE MARCOS ARTIFICIAIS	90
5.4 TESTE DO SISTEMA DE LOCALIZAÇÃO DE PORTAS.....	94
5.5 TESTE DE PASSAGEM POR PORTAS	98
5.6 TESTE DE NAVEGAÇÃO.....	106
6 CONCLUSÕES	111

1 INTRODUÇÃO

1.1 CONTEXTUALIZAÇÃO E JUSTIFICATIVA

A deficiência é uma condição que pode impactar a vida de um indivíduo no que diz respeito às estruturas e funções do corpo, à realização de atividades e à participação social (VARELA; OLIVER, 2013). Segundo o Relatório Mundial sobre a Deficiência realizado pela World Health Organization (WHO, 2011), em todo o mundo mais de 1 bilhão de indivíduos possuem alguma forma de deficiência, sendo que aproximadamente 200 milhões apresentam dificuldades funcionais consideráveis. Essas pessoas, por enfrentarem barreiras no acesso a serviços, apresentam piores perspectivas de saúde, baixos níveis de escolaridade e menor participação econômica.

Com o intuito de amenizar o impacto da deficiência no cotidiano do indivíduo, as pesquisas em tecnologias assistivas vem crescendo cada vez mais. Conforme Silva (2007), tecnologia assistiva é qualquer dispositivo que busca melhorar a qualidade de vida de pessoas portadoras de limitações físicas, sensoriais ou psicológicas, aumentando as capacidades funcionais, promovendo uma vida mais independente e inclusão social.

As cadeiras de rodas motorizadas são normalmente utilizadas como solução para pacientes com capacidade de locomoção reduzida. Entretanto, existe uma população de pacientes que apresentam elevado grau de deficiência, e esses indivíduos possuem extrema dificuldade de manobrar as cadeiras de rodas motorizadas atuais disponíveis no mercado, logo, sua mobilidade é altamente limitada, ou até mesmo impossível (PETRY, 2008).

Um estudo realizado por Fehr, Langbein e Skaar (2000), onde indivíduos com dificuldades de locomoção foram submetidos a um treino em cadeira de rodas, verificou que 10% dos pacientes relataram extrema dificuldade de realizar suas tarefas, e que 40% tiveram dificuldade em tarefas específicas. Além disso, conforme relatos, metade dos pacientes seriam beneficiados caso fosse desenvolvido algum sistema autônomo de navegação (PETRY, 2008).

Um sistema autônomo é um sistema que possui recursos que possibilita a realização de determinadas tarefas sem a necessidade de interferência externa durante o processo. Uma das principais características presentes em um sistema

autônomo é a capacidade de se adaptar e contornar situações imprevistas quando interage com um meio externo dinâmico (FABRO, 1996). Em particular, a navegação autônoma é um processo que permite que um dispositivo se mova a partir de uma posição inicial para uma final, obedecendo às restrições cinemáticas e dinâmicas, e desviando de obstáculos quando necessário (BEZERRA, 2004).

Diante do contexto apresentado, foram desenvolvidos no laboratório de biosinais da Universidade de Caxias do Sul dois trabalhos de conclusão do curso de Engenharia Elétrica. Vieira (2019) desenvolveu um sistema de navegação autônoma de cadeiras de rodas motorizada em corredores de ambientes previamente conhecidos, e Boschetti (2019) desenvolveu um sistema de navegação assistiva de cadeiras de rodas motorizada em passagem por portas utilizando um sensor Kinect. A partir dos conceitos desses projetos, este trabalho propõe o desenvolvimento de um sistema de navegação autônoma, entre corredores e portas, de uma cadeira de rodas motorizada em um ambiente conhecido.

O sistema de navegação autônomo proposto utiliza o algoritmo Dijkstra para gerar uma trajetória por meio de um mapa de grade do ambiente. A posição absoluta da cadeira é determinada através da localização de marcos artificiais. Os marcos são formados por retângulos colocados no ambiente em posições previamente conhecidas e são detectados por meio do processamento de imagens RGB-D capturadas com uma câmera Kinect v2. A posição relativa da cadeira durante o deslocamento é estimada por meio da odometria utilizando *encoders* nas rodas, giroscópio e acelerômetro. A porta a ser transpassada é localizada a partir das informações de profundidade obtidas pelo Kinect. O deslocamento da cadeira entre corredores e na passagem por portas é controlado por controladores baseados em lógica *fuzzy* que atuam na velocidade linear e angular.

1.2 OBJETIVO GERAL

O objetivo deste trabalho consiste no desenvolvimento de um sistema de navegação autônoma de uma cadeira de rodas por corredores e portas de um ambiente previamente conhecido, fazendo uso de um sensor Kinect e da técnica de odometria corrigida por marcos artificiais.

1.3 OBJETIVOS ESPECÍFICOS

Os objetivos específicos necessários para atingir o objetivo geral são destacados a seguir:

- a) utilizar o algoritmo Dijkstra para gerar uma trajetória a partir de um mapa discreto do ambiente.
- b) aplicar um sistema de controle capaz de guiar a cadeira de rodas durante a trajetória a ser seguida.
- c) definir uma técnica e algoritmo para detectar uma porta aberta e estimar sua posição em relação à cadeira de rodas.
- d) empregar um sistema de controle para realizar a passagem por portas.
- e) calibrar e avaliar a odometria utilizando a metodologia UMBmark.
- f) avaliar a aplicação de filtros em imagens de profundidade do sensor Kinect v2.
- g) implementar um sistema capaz de localizar marcos artificiais durante a trajetória da cadeira de rodas no ambiente de navegação, utilizando imagens RBG-D do sensor Kinect v2.
- h) comparar o deslocamento registrado pelo sistema na passagem pela porta com o deslocamento do modelo teórico.
- i) avaliar se a odometria juntamente com o sensor Kinect pode ser aplicada em sistemas de navegação autônoma de cadeiras de rodas motorizadas de forma confiável.

1.4 ESCOPO E RESTRIÇÕES

As seguintes restrições são aplicadas ao presente trabalho com a finalidade de restringir-se aos objetivos já discutidos:

- a) o sistema de navegação a ser desenvolvido não avalia possíveis obstáculos que podem ser encontrados em um ambiente durante a trajetória, como lixeiras, cadeiras, armários e demais objetos que não fazem parte da infraestrutura do prédio.
- b) o sistema se limita a realizar a navegação da cadeira de rodas apenas na trajetória entre portas e corredores.
- c) o sistema somente localizará portas que estejam totalmente abertas e livres de obstruções, onde pode-se detectar pelo menos uma das bordas da porta.

- d) o controle de potência dos motores da cadeira de rodas não faz parte do escopo do trabalho. Desta forma, o controle de potência do motor será efetuado pelo próprio *driver* da cadeira de rodas motorizada, fornecido pelo fabricante.

1.5 APRESENTAÇÃO DO TRABALHO

Este trabalho está dividido em 5 capítulos, sendo que o primeiro é a introdução, onde é apresentado a contextualização que justifica a escolha do tema, bem como os objetivos deste trabalho. O segundo capítulo apresenta a revisão bibliográfica sobre os métodos de navegação de robôs móveis que são necessários para o desenvolvimento deste trabalho, onde são apresentadas técnicas de localização de robôs móveis, sensores de profundidade, geração de trajetória e o controlador *fuzzy*, além disso são apresentadas técnicas de processamento de imagens para localização de formas geométricas e o modelo de ruído do sensor Kinect v2. O terceiro capítulo consiste em um resumo dos trabalhos relacionados que serão utilizados como base para este. No quarto capítulo é apresentado as metodologias utilizadas para o desenvolvimento deste trabalho, e por fim, no quinto capítulo são mostrados os ensaios realizados para a validação do sistema desenvolvido e os resultados obtidos.

2 REVISÃO BIBLIOGRÁFICA

Com a finalidade de fundamentar os métodos a serem utilizados no desenvolvimento deste trabalho, neste capítulo serão apresentados conceitos básicos de navegação de robôs móveis, abordando o método de localização relativa por odometria e sua calibração através do método UMBmark, métodos de localização absoluta, sensores de profundidade, geração de trajetória e o sistema de controle baseados em conjuntos *fuzzy*. Também serão apresentadas a transformada de Hough e o detector de bordas de Canny, que são técnicas de processamento de imagens que podem ser utilizadas para localizar formas geométricas em imagens. Além disso, será apresentado o modelo de ruído de imagens de profundidade do sensor Kinect v2 que pode ser aplicado em filtros.

2.1 NAVEGAÇÃO

Segundo Braga (2010), a navegação é o processo de movimentação de um robô, de uma posição e orientação inicial para uma final em seu ambiente de trabalho a partir de um ponto de referência, além de desviar de objetos que podem ser encontrados no caminho. Para que o processo de navegação de robôs móveis seja possível, é necessário a realização de 4 tarefas básicas:

- a) localização: a partir das informações dos sensores e do mapa interno do ambiente, o robô localiza sua própria posição;
- b) mapeamento: é a representação do ambiente em formato de mapa, construído baseado nas informações do sistema sensorial à medida que o robô se desloca no ambiente;
- c) planejamento: a partir de um mapa previamente armazenado é determinada uma trajetória de uma posição inicial para uma final, de forma a evitar colisões com obstáculos;
- d) controle: é o método que controla o deslocamento do robô no ambiente de forma a seguir a trajetória definida na etapa de planejamento.

2.2 LOCALIZAÇÃO

Segundo Castro (2017), localizar um robô consiste em determinar a sua posição e orientação, ou seja, em um sistema de coordenadas global, determinar as coordenadas x e y e direção. A localização é um problema fundamental para o sucesso de um sistema robótico autônomo, uma vez que se o robô não sabe onde ele está localizado em um ambiente é impossível decidir como deve agir. O robô necessita de pelo menos alguma estimativa de sua posição para poder determinar o que fazer. Existem dois métodos principais de localização: localização relativa e localização absoluta.

De acordo com Bezerra (2004), para realizar a localização de um robô, é necessário usar sensores com a finalidade de obter informações do ambiente ao seu redor. Diversos tipos de sensores vêm sendo empregados para executar essa tarefa, por exemplo, *encoders*, lasers, câmeras digitais, sonares, bússolas, etc. A utilização de mais de um sensor de forma combinada é frequente.

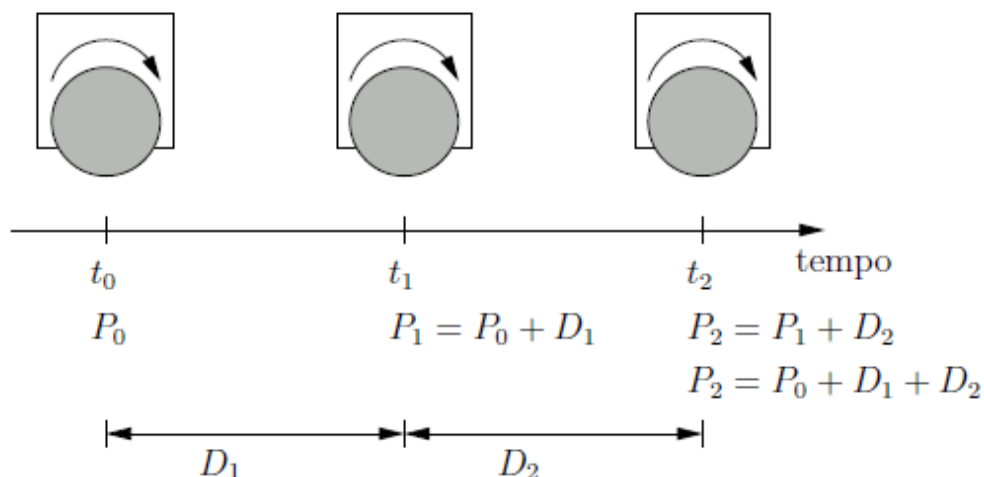
2.2.1 Localização relativa: odometria

O método de localização relativa, também chamado *Dead Reckoning*, foi um dos primeiros métodos utilizados para localização, e ainda vem sendo frequentemente utilizado devido a sua simplicidade. Esse processo consiste em estimar a posição atual baseado na posição registrada nos instantes anteriores. Essa técnica estima a posição apenas com base na velocidade e direção, e no tempo desde a última posição conhecida. Entretanto, verifica-se que os erros de medidas se acumulam com o passar do tempo. Em aplicações robóticas, normalmente é utilizado a odometria para a medição da posição relativa (CASTRO, 2017).

A odometria consiste em estimar a posição e orientação de um robô, medindo o deslocamento das rodas em relação a uma posição referencial fixa. Para entender melhor esse método, considere um robô em movimento como mostra a Figura 1. O robô se localiza na posição P_0 no instante t_0 , em seguida no instante t_1 ele se encontra em P_1 , desta forma, sua posição pode ser determinada como sendo a soma da sua posição anterior e o deslocamento D_1 . Por fim, em t_2 o robô está na posição P_2 , que pode ser obtida somando a posição anterior e o deslocamento D_2 . Com isso,

a posição atual do robô pode ser calculada por meio da integral do deslocamento realizado pelo robô, em relação à posição inicial (BEZERRA, 2004).

Figura 1 - Deslocamento do robô utilizando odometria



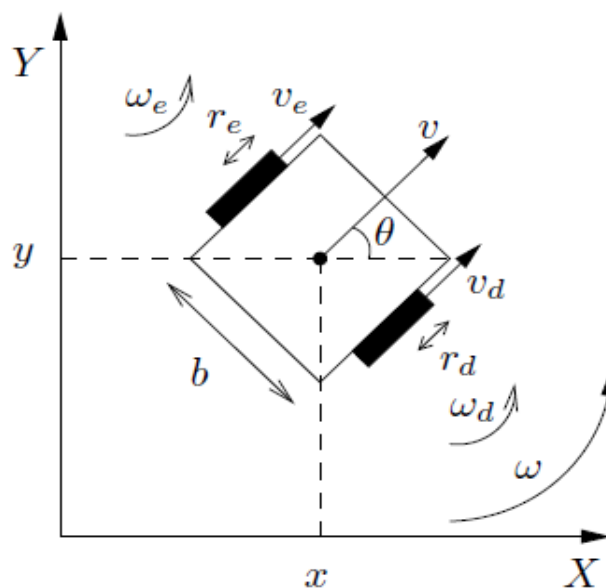
Fonte: Bezerra (2004).

O deslocamento do robô pode ser determinado medindo a rotação das rodas. Um sensor muito utilizado nessa tarefa é o *encoder* óptico, esse dispositivo funciona por meio da transmissão e recepção de luz através de um disco perfurado, que gira acoplado ao eixo do motor do robô (BEZERRA, 2004).

As variáveis cinemáticas de uma plataforma robótica com acionamento diferencial são mostradas na Figura 2, Onde:

- a) b é a distância entre as rodas;
- b) r_e é o raio da roda esquerda;
- c) r_d é o raio da roda direita;
- d) ω_e é a velocidade angular da roda esquerda;
- e) ω_d é a velocidade angular da roda direita;
- f) v_e é a velocidade linear da roda esquerda;
- g) v_d é a velocidade linear da roda direita;
- h) v é a velocidade linear do robô;
- i) ω é a velocidade angular do robô;

Figura 2 - Modelo cinemático de um robô com acionamento diferencial



Fonte: Bezerra (2004).

As velocidades, linear e angular, podem ser calculadas por meio das equações 1 e 2.

$$v = \frac{v_d + v_e}{2} \quad (1)$$

$$\omega = \frac{v_d - v_e}{b} \quad (2)$$

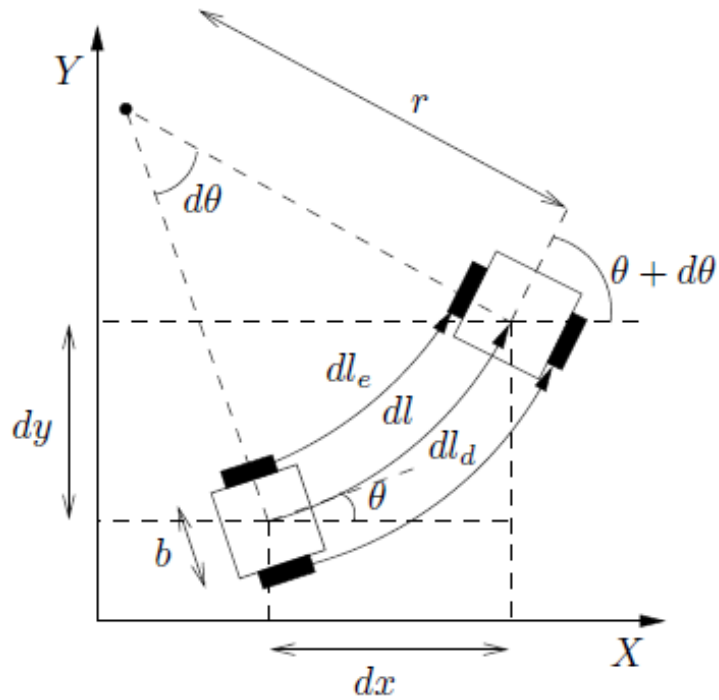
Considerando que o movimento do robô é realizado em um intervalo de tempo infinitesimal, como mostrado na Figura 3, o deslocamento pode ser decomposto em componentes vertical e horizontal. Essas componentes e o deslocamento angular do robô são dados pelas equações 3, 4 e 5:

$$dx = dl \cos \theta = v dt \cos \theta \quad (3)$$

$$dy = dl \sin \theta = v dt \sin \theta \quad (4)$$

$$d\theta = \omega dt \quad (5)$$

Figura 3 - Movimento infinitesimal do robô



Fonte: Bezerra (2004).

Discretizando as equações 3, 4 e 5 obtêm-se as equações 6, 7 e 8:

$$x(t + \Delta t) = x(t) + v \cdot \Delta t \cos \theta(t) \quad (6)$$

$$y(t + \Delta t) = y(t) + v \cdot \Delta t \sin \theta(t) \quad (7)$$

$$\theta(t + \Delta t) = \theta(t) + \omega \cdot \Delta t \quad (8)$$

A velocidade de cada roda pode ser calculada por meio das equações 9 e 10, onde N_{res} é a resolução do encoder, e N_d e N_e são a quantidade de pulsos lidos no encoder direito e esquerdo respectivamente.

$$v_d = \frac{N_d}{N_{res}} \frac{2\pi r_d}{\Delta t} \quad (9)$$

$$v_e = \frac{N_e}{N_{res}} \frac{2\pi r_e}{\Delta t} \quad (10)$$

Substituindo as equações 9 e 10 nas equações 6, 7 e 8, utilizando as equações 1 e 2, a localização de um robô em relação a um referencial inicial é dado pelas equações 11, 12 e 13:

$$x(t + \Delta t) = x(t) + \frac{\pi(N_d.r_d + N_e.r_e)}{N_{res}} \cos \theta(t) \quad (11)$$

$$y(t + \Delta t) = y(t) + \frac{\pi(N_d.r_d + N_e.r_e)}{N_{res}} \sen \theta(t) \quad (12)$$

$$\theta(t + \Delta t) = \theta(t) + \frac{2\pi(N_d.r_d - N_e.r_e)}{N_{res}} \quad (13)$$

De acordo com Bezerra (2017), como a odometria é baseada na soma das distâncias medidas pelos sensores, um erro de medida ocorrido em um determinado instante afeta todas as medições seguintes, desta forma, essa técnica acaba fornecendo apenas uma estimativa da localização do robô. Os erros da odometria são classificados em erros sistemáticos e erros não-sistemáticos.

Os erros sistemáticos são provocados por imperfeições no modelo cinemático do robô, esses erros se acumulam constantemente durante toda a navegação causando grandes distorções na estimação da posição. Já os erros não-sistemáticos são erros que ocorrem de forma inesperada e não estão presentes em todo o processo de navegação (BEZERRA, 2017). As causas dos erros sistemáticos e não-sistemáticos são:

a) erros sistemáticos:

- diferença entre os diâmetros das rodas esquerda e direita;
- média do diâmetro das rodas diferente do diâmetro nominal;
- comprimento do eixo diferente do comprimento nominal;
- desalinhamento das rodas;
- resolução finita do *encoder*;
- taxa de amostragem do *encoder* finita;

b) erros não-sistemáticos:

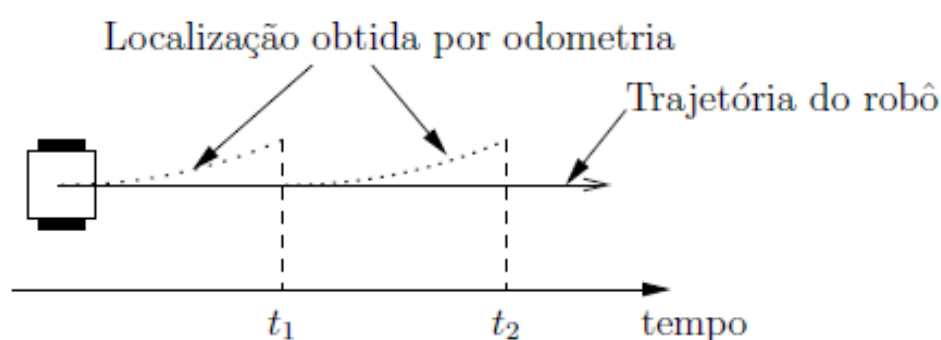
- terrenos irregulares;
- objetos inesperados no chão;

- escorregamento das rodas.

De acordo com Bezerra (2004), os erros sistemáticos em robôs com acionamento diferencial são causados principalmente por erros na determinação dos diâmetros das rodas e do comprimento do eixo do robô. Estes parâmetros podem ser calibrados utilizando um método conhecido como UMBmark (University of Michigan Benchmark). A metodologia desta técnica será discutida na próxima seção.

Como a odometria apresenta erros na estimação da localização do robô, e que aumentam juntamente com distância percorrida, é necessário a utilização de um sistema de localização absoluta com o objetivo de corrigir periodicamente a localização afetada pelo erro acumulado. Como é mostrado na Figura 4, onde a linha tracejada indica a localização do robô registrada pela odometria e que se torna cada vez mais incorreta ao longo da trajetória, nos instantes t_1 e t_2 a localização é corrigida com base em um método de localização absoluta (BEZERRA, 2017).

Figura 4 - Correção do erro da odometria por meio da localização absoluta



Nota: A linha tracejada mostra a localização registrada pela odometria que se torna cada vez mais incorreta ao longo da trajetória. Nos instantes t_1 e t_2 por meio de um método de localização absoluta o erro acumulado pela odometria é corrigido.

Fonte: Bezerra (2004).

2.2.2 Calibração da odometria: UMBmark

Borenstein e Feng (1996) desenvolveram um método chamado UMBmark que busca diminuir os erros sistemáticos em robôs móveis com acionamento diferencial. A técnica consiste em submeter o robô a trajetórias quadrangulares no sentido horário e anti-horário, e a partir das medidas dos erros de localização, são recalculados os parâmetros cinemáticos do robô (diâmetros das rodas e comprimento do eixo) (BEZERRA, 2017).

Conforme Tomasi Junior (2016), no ensaio o robô deve executar uma trajetória quadrada de 4x4m de comprimento, e ao final da execução calcula-se o erro entre a posição final absoluta e a posição final registrada pela odometria. O processo deve ser realizado cinco vezes em cada sentido, horário (*cw*) e anti-horário (*ccw*), e o robô deve partir sempre da mesma posição inicial.

O erro da odometria é calculado pelas equações 14 e 15, onde X_{abs} e Y_{abs} são as coordenadas absolutas da localização do robô, X_{calc} e Y_{calc} são as coordenadas da localização do robô registrada pela odometria.

$$eX = X_{abs} - X_{calc} \quad (14)$$

$$eY = Y_{abs} - Y_{calc} \quad (15)$$

A partir dos dados obtidos, são calculadas as médias dos erros nos eixos x e y pelas equações 16 e 17, com o objetivo de diminuir a influência dos erros não sistemáticos nos cálculos dos parâmetros cinemáticos.

$$mX_{cw/ccw} = \frac{1}{n} \sum_{i=1}^n eX(i) \quad (16)$$

$$mY_{cw/ccw} = \frac{1}{n} \sum_{i=1}^n eY(i) \quad (17)$$

Como os erros são calculados de forma separada para o sentido horário e anti-horário, são calculadas pelas equações 18 e 19 as distâncias euclidianas dos pontos X médio e Y médio, e o sentido que apresentar o maior erro conforme a Equação 20 é utilizado para a avaliação do desempenho da odometria.

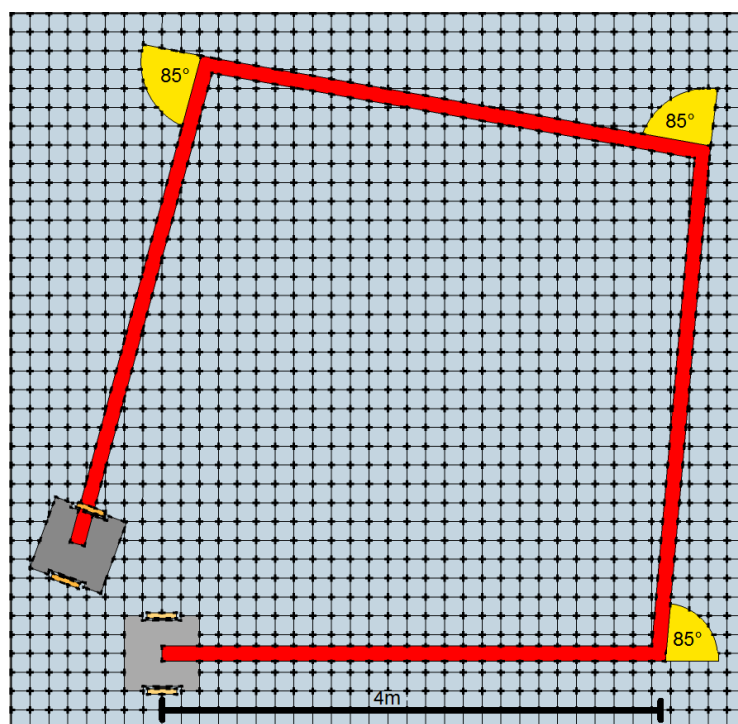
$$euc_{cw} = \sqrt{(mX_{cw})^2 + (mY_{cw})^2} \quad (18)$$

$$euc_{ccw} = \sqrt{(mX_{ccw})^2 + (mY_{ccw})^2} \quad (19)$$

$$eMax = \max (euc_{cw}, euc_{ccw}) \quad (20)$$

Os erros que ocorrem devido a distância entre rodas do modelo cinemático ser diferente da distância real, são chamados de erros tipo A. Quando esse tipo de erro está presente, o robô realiza giros diferentes ao solicitado (TOMASI JUNIOR, 2016). A Figura 5 mostra um exemplo de trajetória realizada por um robô com erros tipo A.

Figura 5 - Trajetória característica de um robô com erros tipo A



Fonte: Tomasi Junior (2016).

Para corrigir os erros tipo A é necessário calcular uma constante de correção E_b , que é determinada por meio da equação 21, onde α é o giro real realizado pelo robô e pode ser calculado pela equação 22.

$$E_b = \frac{90}{90 - \alpha} \quad (21)$$

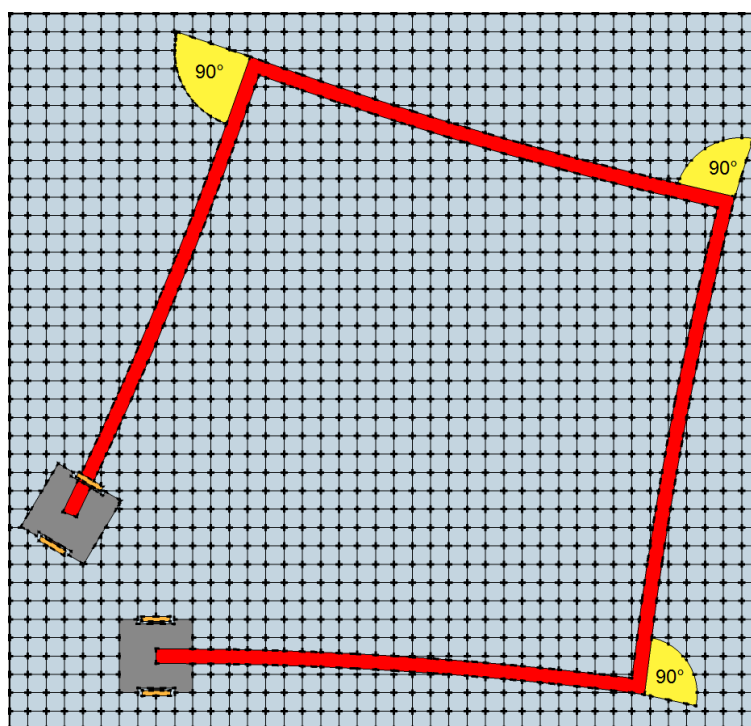
$$\alpha = \frac{mX_{cw} + mX_{ccw}}{-4L} \quad (22)$$

Com a constante E_b calculada, é possível calcular a distância entre as rodas através da equação 23.

$$b_{atual} = E_b \times b_{nominal} \quad (23)$$

Os erros que ocorrem devido a diferença entre o diâmetro das rodas do robô e do modelo cinemático, são chamados de erros tipo B. Quando esse tipo de erro está presente, o robô realiza um movimento não linear (TOMASI JUNIOR, 2016). A Figura 6 mostra a trajetória característica de um robô que apresenta erros tipo B.

Figura 6 - Trajetória característica de um robô com erros tipo B

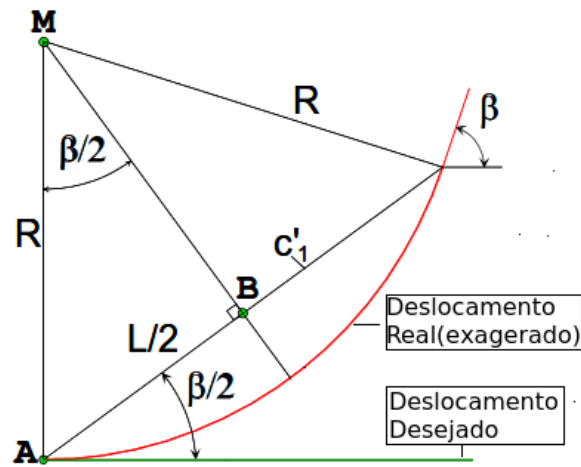


Fonte: Tomasi Junior (2016).

Para corrigir os erros tipo B é necessário calcular uma constante de correção E_d , que é determinada através da equação 24. Onde b é a distância entre as rodas, r_e e r_d são respectivamente o raio da roda esquerda e direita, e R é o raio de curvatura da trajetória do robô. A Figura 7 exemplifica a dedução do cálculo da constante R .

$$E_d = \left(\frac{r_e}{r_d}\right) \frac{\left(R - \frac{b}{2}\right)}{\left(R + \frac{b}{2}\right)} \quad (24)$$

Figura 7 - Relações geométricas de deslocamento



Fonte: Tomasi Junior (2016).

A constante R pode ser determinada pela equação 25, onde β representa o ângulo da trajetória não circular que pode ser calculado pela equação 26.

$$R = \frac{\frac{L}{2}}{\text{sen } \frac{\beta}{2}} \quad (25)$$

$$\beta = \frac{mX_{cw} - mX_{ccw}}{-4L} \quad (26)$$

Com a constante E_d calculada, é possível calcular os raios das rodas por meio da equação 27 e 28, onde rm é a média dos raios das rodas do modelo cinemático.

$$re_{\text{corrigido}} = \frac{2}{E_d + 1} rm \quad (27)$$

$$rd_{\text{corrigido}} = \frac{2}{\frac{1}{E_d} + 1} rm \quad (28)$$

Como o método de calibração UMBmark considera que os erros tipo A e B atuam de forma separada, ou seja, um erro não influencia no outro, é necessário

executar o processo de calibração mais de uma vez para corrigir os parâmetros com maior eficiência.

Mesmo com a calibração pelo método UMBmark a odometria ainda acumulará erros durante a trajetória. Logo, como já citado anteriormente, é necessário utilizar alguma técnica de localização absoluta de forma complementar à odometria para corrigir os erros acumulados.

2.2.3 Localização absoluta

Diferentemente da localização relativa, a localização absoluta não depende da posição registrada anteriormente, ou seja, a localização é feita por uma única medida absoluta, sem a necessidade de somar uma sequência de medidas. A principal vantagem dessa técnica é que não há acúmulo de erro com o passar do tempo. A localização absoluta pode ser efetuada baseada em marcos de referência ou em mapas (CASTRO, 2017).

Conforme Castro (2017), na localização absoluta baseada em marcos de referência, o robô se localiza no ambiente através da combinação de informações extraídas de características detectáveis junto com informações previamente conhecidas. Os marcos são divididos em passivos e ativos.

Na técnica de marcos ativos, o robô se localiza baseado em informações capturadas de ondas de rádio ou sinais de satélite. Já no método de marcos passivos, o robô precisa enxergar os marcos para se localizar, nesse tipo de sistema os sensores empregados dependem diretamente do tipo de marco a ser detectado. Os marcos passivos são divididos em artificiais e naturais (CASTRO, 2017).

Os marcos artificiais são colocados em lugares conhecidos no ambiente de navegação, de forma a serem facilmente detectados pelos sensores do robô. Alguns exemplos de marcos artificiais são códigos de barras e figuras geométricas, como quadrados e círculos. Uma das desvantagens dos artificiais é que quanto mais longe o robô se encontra do marco, menor será a precisão da estimação da posição, além disso, em diferentes condições de iluminação, ou visibilidade parcialmente bloqueada, as medidas podem ser incorretas ou ausentes. Desta forma, os marcos artificiais podem ser empregados principalmente em ambientes internos. (CASTRO, 2017).

Os marcos naturais são nativos do ambiente de navegação. Em ambientes externos, alguns exemplos de marcos naturais são ruas, árvores e placas, enquanto

em ambientes internos são portas, janelas e lâmpadas. O sistema de localização baseado em marcos naturais se comparado ao de marcos artificiais, apresenta menor confiabilidade de reconhecimento e maior complexidade computacional (CASTRO, 2017).

Outro método de localização absoluta é a baseada em mapas. Nessa técnica são utilizadas características do ambiente que podem ser extraídas por meio de sensores sonares, câmeras RGB-D e Lasers Scanners. A localização do robô é realizada combinando estas características com informações de um mapa do ambiente previamente armazenado (CASTRO, 2017).

As Câmeras RGB-D, como o Microsoft Kinect, fornecem dados de cores e profundidade em cada pixel em imagens. Esses equipamentos reduzem a complexidade da computação visual, com isso, eles vêm sendo utilizados em pesquisas voltadas à localização e mapeamento de ambientes internos, onde as condições de iluminação são favoráveis (SCHNEIDER; STEMMER, 2019).

2.2.4 Sensor de profundidade - RGB-D

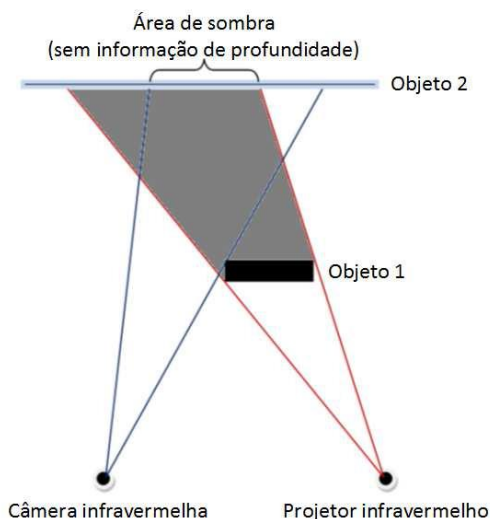
Segundo Da Motta (2016), os sensores de profundidade são equipamentos que fornecem o mapa de profundidade de uma imagem. O mapa de profundidade armazena a distância entre o objeto e a câmera em uma matriz bidimensional que corresponde a cada pixel da imagem capturada. Em alguns sensores o mapa é integrado na imagem RGB, originando o sensor de profundidade RGB-D. As principais técnicas utilizadas para capturar o mapa de profundidade são a de luz estruturada e o *time of flight* (ToF).

No sensor de profundidade que usa a tecnologia de luz estruturada, um projetor de luz infravermelha emite uma luz em forma estruturada, e uma câmera captura a luz refletida pelos objetos do ambiente. A distância entre o sensor e cada ponto de luz refletido é calculado a partir da análise das distorções no padrão de luz ocorridas ao refletir nos objetos. Nesse tipo de sensor, a iluminação proveniente de outras fontes, causam interferência na estimação da profundidade, desta forma, esses sensores são indicados em aplicações internas com iluminação controlada (DA MOTTA, 2016).

Um problema dessa tecnologia é que ela pode não conseguir mapear todas as regiões. Devido à distância existente entre o projetor e a câmera, são originadas regiões de sombra, isso ocorre porque a luz emitida pelo projetor não atinge algumas

áreas visíveis à câmera devido a existência de objetos que obstruem a passagem da luz (DA MOTTA, 2016). A Figura 8 exemplifica como são geradas as regiões de sombra.

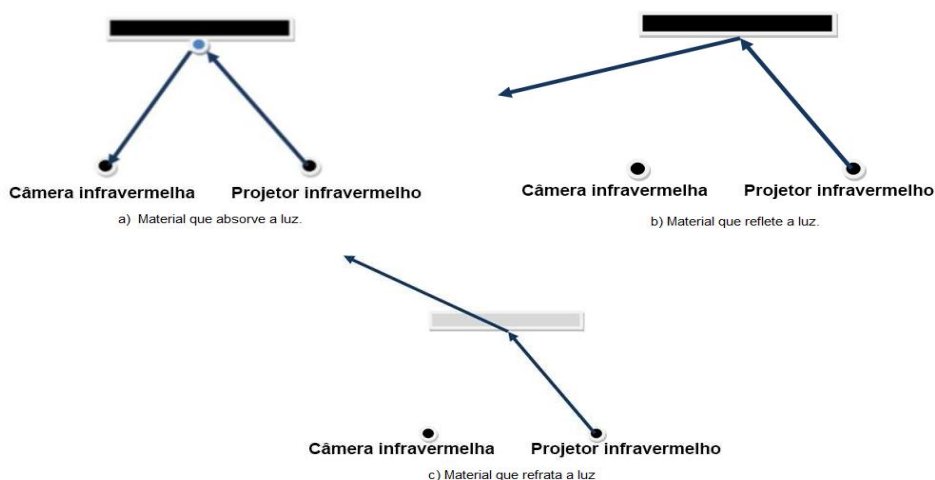
Figura 8 - Formação de sombras no sensor de profundidade de luz estruturada



Fonte: Da Motta (2016)

Além das regiões de sombras, outras áreas que não podem ser mapeadas são as superfícies formadas por materiais capazes de promover a reflexão regular da luz, como espelhos e louças, e materiais que refratam a luz, como vidros e plásticos transparentes (DA MOTTA, 2016). A Figura 9 mostra fenômenos que podem ocorrer com a luz emitida pelo projetor.

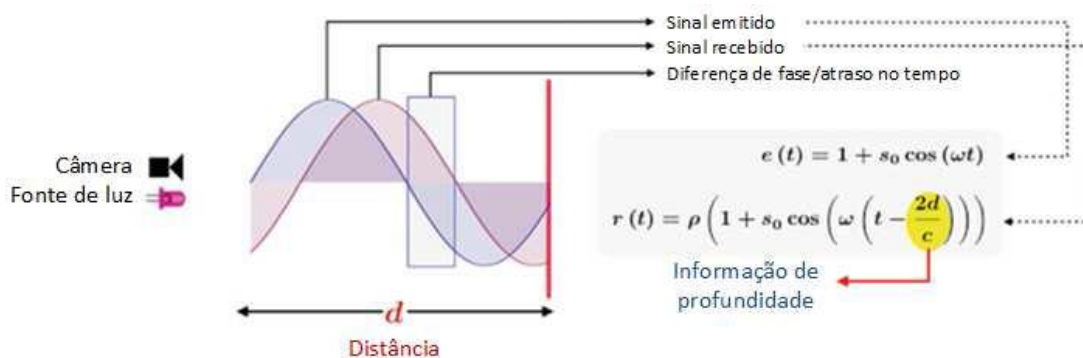
Figura 9 - Comportamento da luz em superfícies de diferentes matérias



Fonte: Da Motta (2016).

O sensor de profundidade que usa a tecnologia *time of flight* consiste em um emissor de luz infravermelha acoplado com uma câmera que captura a luz refletida. O cálculo da distância de cada pixel é baseado no “tempo de voo” da luz, ou seja, a distância é determinada pela diferença de fase entre a luz emitida e a luz capturada pelo sensor (DA MOTTA, 2016). A Figura 10 mostra o funcionamento básico do método *time of flight*.

Figura 10 - Funcionamento do sensor ToF



Fonte: Da Motta (2016).


A vantagem desse tipo de sistema é que como a câmera e o emissor de luz estão próximos, os erros provocados por oclusões e regiões de sombras são reduzidos. Entretanto, devido ao processamento exigido pela tecnologia, as câmeras ToF normalmente apresentam uma resolução mais baixa. Além disso, o sensor pode saturar devido a radiação de luz por fontes externas (DA MOTTA, 2016).

2.2.5 Sensor Kinect

Em 2010, a Microsoft lançou a câmera RGB-D Kinect de primeira geração para os consoles Xbox, o que permitiu aos jogadores interagirem com os jogos eletrônicos através de movimentos em tempo real. Em 2013, a primeira geração foi substituída pela segunda com funcionalidades semelhantes. A principal diferença é que a primeira geração utiliza a luz estruturada para calcular a distância, já a segunda usa a tecnologia ToF. (DA MOTTA, 2016). A Figura 11, mostra a diferença das especificações entre as duas gerações do Kinect.

Figura 11 - Especificações de hardware dos modelos do sensor Kinect

Comparing the Different Kinect Generations



	1 st Generation Kinect	2 nd Generation Kinect
Color resolution/rate	1280x960 @ 12 Hz <i>or</i> 640x480 @ 30 Hz	1920x1080 @ 30 Hz
Infrared resolution/rate	640x480 @ 30 Hz	512x424 @ 30 Hz
Depth resolution/rate	320x240 @ 30 Hz	512x424 @ 30 Hz
Depth range*	0.4 m – 3.0 m <i>or</i> 0.8 m – 4.0 m	0.5 m – 4.5 m
Depth sensing technology	Structured light	Time-of-flight
Field of view (horizontal)	58°	71°
Mic array	4 elements	4 elements
Tilt motor	±27°	none

Fonte: Da Motta (2016)

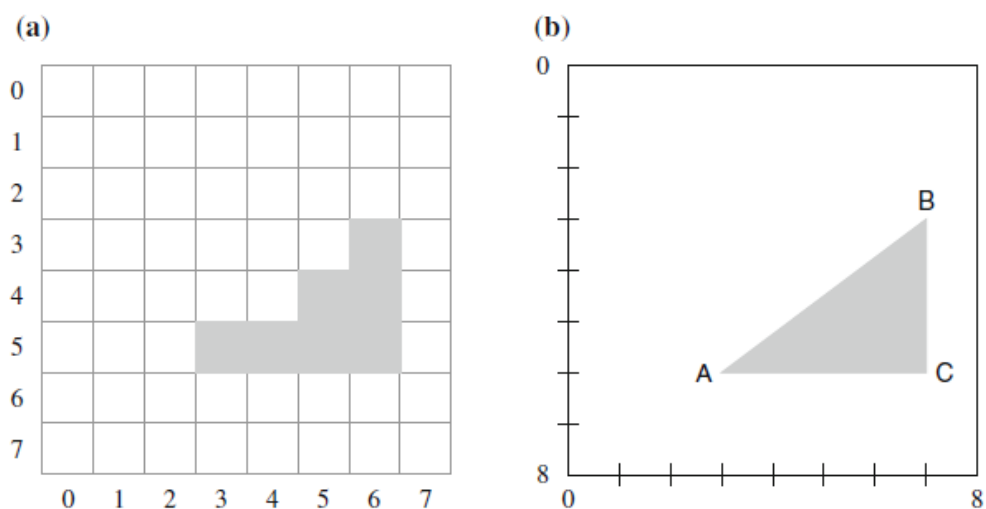
Através da Figura 11, observa-se outras diferenças entre as gerações além da tecnologia utilizada para medir a distância. O Kinect de primeira geração consegue adquirir imagens de profundidade em distâncias entre 0,3 e 3 metros ou 0,8 e 4 metros com uma resolução de 320x240, já o Kinect de segunda geração captura imagens de profundidade entre 0,5 e 4,5 metros com uma resolução de 512x424. Além disso, o Kinect v2 possui um campo de visão de 71° que é maior do que o campo de visão do Kinect de primeira geração que é de 58°.

2.3 GERAÇÃO DE TRAJETÓRIA

Segundo Rodrigues (2017), a geração de trajetória consiste em determinar o caminho mais curto a ser percorrido pelo robô a partir da localização atual e um ponto de destino. Durante o processo de navegação, o sistema robótico deve ser capaz de reagir a obstáculos, redefinindo a trajetória com o objetivo de evitar colisões.

Conforme Ben-Ari e Mondada (2018), para um robô gerar uma trajetória é necessário armazenar um mapa virtual em sua memória. Basicamente existem dois tipos de mapas digitais, os mapas discretos e os mapas contínuos. A Figura 12 exibe um exemplo de objeto triangular, sendo representado em um mapa discreto e contínuo.

Figura 12 - Exemplo de mapa discreto e contínuo



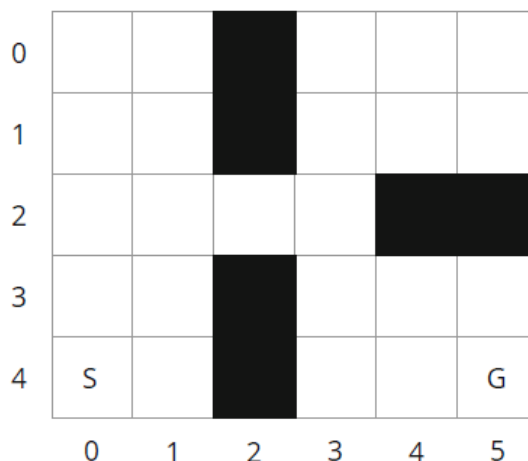
Nota: (a) mapa discreto de um triângulo, (b) mapa contínuo de um triângulo.
 Fonte: Ben-Ari e Mondada (2018).

Os mapas contínuos são mais eficientes, além de precisos, em ambientes que possuem poucos objetos e de formatos simples. Já os mapas discretos são mais eficientes em termos de memória e quantidade de computação necessária, principalmente em ambientes que apresentam muitos objetos e de formatos complexos. Entretanto os mapas discretos não são precisos como pode ser visto na Figura 12a, onde é impossível reconhecer que o objeto representado é um triângulo. Para melhorar a precisão é necessário utilizar uma grade mais fina (256x256, por exemplo), mas isso aumenta o custo de memória e processamento computacional (BEN-ARI; MONDADA, 2018).

Um mapa discreto utiliza notação convencional para descrever o ambiente. A notação mais simples é alocar um bit para cada célula, o valor 1 representa um objeto na célula e o valor 0 indica que a célula está vazia (BEN-ARI; MONDADA, 2018). Por exemplo, na Figura 12a, a cor cinza equivale ao valor 1 e a cor branca equivale ao valor 0.

Conforme Ben-Ari e Mondada (2018), a partir de um mapa é possível planejar o caminho utilizando um algoritmo de nível superior. Edsger W. Dijkstra propôs um algoritmo para encontrar o caminho mais curto em um mapa de grade. A Figura 13 mostra um exemplo de mapa de grade, onde S é a célula inicial, G é a célula final e as células em preto são os obstáculos.

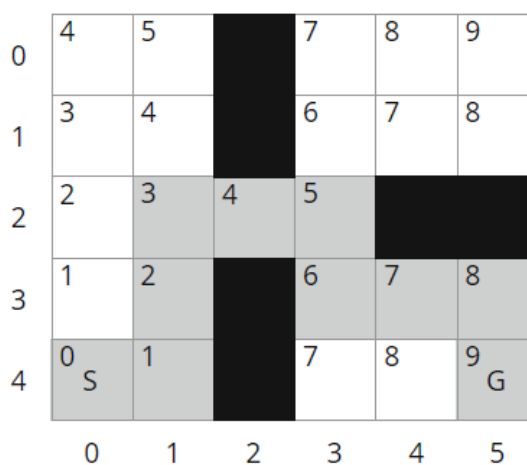
Figura 13 - Exemplo de mapa de grade



Fonte: Ben-Ari e Mondada (2018).

O algoritmo marca cada célula com o número de etapas necessárias para chegar na célula inicial. Inicialmente, a célula inicial por não apresentar nenhuma etapa é a marca com 0, em seguida todos seus vizinhos são marcados com 1, pois estão a 1 etapa da célula inicial. O processo continua iterativamente, se uma célula está marcada com um valor n , suas células vizinhas que não estiverem marcadas serão marcadas com o valor $n+1$. Quando a célula final for marcada, para encontrar o caminho mais curto, basta voltar seguindo as células marcadas com o menor número de etapas (BEN-ARI; MONDADA, 2018). O resultado do algoritmo é mostrado na Figura 14.

Figura 14 - Resultado do algoritmo Dijkstra



Fonte: Ben-Ari e Mondada (2018).

2.4 CONTROLADOR FUZZY

De acordo com Sandeep e Supriya (2016), muitos dos problemas do mundo real não possuem relações de entradas e saídas bem definidas. Desta forma, não é possível utilizar a teoria clássica dos conjuntos para resolver esses problemas, logo, nessas situações a lógica *fuzzy* pode ser útil. A lógica *fuzzy* assume vários valores e produz um grau de verdade dentro de um conjunto de associação, com isso, essa ferramenta é conveniente para trabalhar com dados imprecisos e incertos, em sistemas de tomada de decisão complexos e não lineares, como na navegação de robôs móveis.

Na teoria clássica, os elementos de um universo X , pertencem ou não pertencem a um conjunto A . Já no sistema *fuzzy*, um conjunto A em um universo X é definido por uma função de pertinência μ_A , e representado por um conjunto de pares ordenados $(x, \mu_A(x))$, onde $\mu_A(x)$ é um valor no intervalo $[0,1]$ que indica o quanto x é compatível com o conjunto A . Um elemento pode pertencer a mais de um conjunto *fuzzy*, com diferentes graus de pertinência (TANSCHEIT, [2004?]).

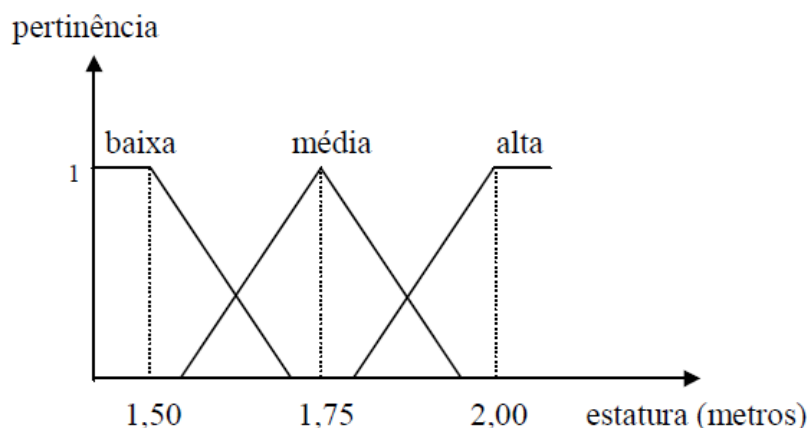
$$\mu_A(x): X \rightarrow [0,1] \quad (29)$$

$$A = \{(\mu_A(x), x) \mid x \in X\} \quad (30)$$

Uma variável linguística é uma variável cujos valores são nomes de conjuntos *fuzzy*. A função dessas variáveis é caracterizar fenômenos complexos e imprecisos de forma sistemática, semelhante às descrições linguísticas usadas pelos seres humanos, e não de variáveis quantificadas. Os valores de uma variável linguística podem ser construídos a partir de termos primários (alto, baixo, pequeno, médio, grande, zero), de conectivos lógicos (e, ou, não), modificadores (muito, pouco, levemente, extremamente) e de delimitadores (como parênteses) (TANSCHEIT, [2004?]).

As funções de pertinência podem ter diferentes formas, normalmente são utilizadas funções de pertinência padrão, como triangular, trapezoidal e gaussiana, mas também podem ser definidas a partir do conhecimento do usuário (TANSCHEIT, [2004?]). Na Figura 15 é mostrado um exemplo de funções de pertinência para a variável estatura (de pessoa).

Figura 15 - Exemplo de função de pertinência para estatura



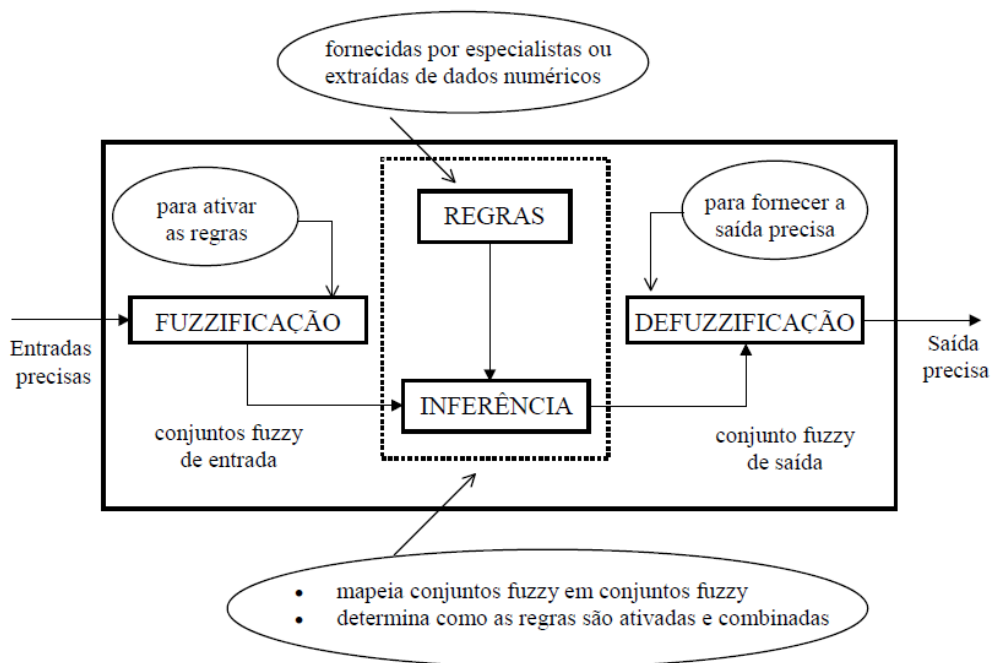
Fonte: Tanscheit (2004).

O controlador *fuzzy* baseado em regras, fundamenta-se estritamente na linguagem natural para descrever como o sistema se comporta. Esses controladores mapeiam por meio de conjuntos linguísticos tanto os antecedentes quanto os consequentes. Caso exista mais de uma variável para cada regra, deve-se aplicar uma técnica de agregação dos conjuntos antecedentes, para gerar um conjunto consequente (DE BARROS FILHO, 2011).

As regras *fuzzy* são combinações de diferentes operadores, como conectivos lógicos (*e*, *ou* e *não*) e operador de implicação (*se...então*). Normalmente o conectivo *ou* conecta valores linguísticos de uma mesma variável, enquanto o conectivo *e* conecta variáveis de diferentes universos. O operador *se...então* descreve a dependência do valor de uma variável linguística em relação ao valor de outra. As regras podem ser fornecidas por um especialista, mas é necessário que elas sejam consistentes para que o desempenho do sistema de inferência *fuzzy* seja adequado (TANSCHHEIT, [2004?]).

Em um sistema de inferência *fuzzy*, os valores de entrada do controlador, que são precisos, são fuzzyficados, ou seja, os valores são transformados em variáveis linguísticas. Em seguida, os dados fuzzyficados são processados juntamente com as regras *fuzzy*, inferindo as ações de controle a serem tomadas. Na última etapa é realizada o processo de defuzzyficação, que transforma as variáveis linguísticas de controle *fuzzy* em variáveis quantitativas (BECKER, 2000). A Figura 16 mostra um sistema de inferência *fuzzy*.

Figura 16 - Sistema de inferência fuzzy



Fonte: Becker (2000).

2.5 PROCESSAMENTO DIGITAL DE IMAGENS

A visão computacional é definida como um conjunto de técnicas utilizadas em sistemas computacionais para interpretar imagens. Na robótica móvel, os sistemas de visão computacional são frequentemente empregados para extrair características do ambiente, auxiliando na localização do robô e na detecção de obstáculos. O sensor mais comumente empregado em robótica móvel para capturar informações do ambiente são as câmeras digitais. (BEZERRA, 2004).

A seguir será apresentado a transformada Hough para retas, que pode ser aplicada em imagens digitais para a detecção de formas geométricas. Antes de aplicar este método é necessário realizar um pré-processamento nas imagens com o objetivo de reduzir a quantidade de informação a serem processadas, com isso, também será apresentado o detector de bordas de Canny. Além disso, será apresentado o modelo de ruído da câmera Kinect v2 que pode ser aplicado em filtros de imagens de profundidade.

2.5.1 Transformada de Hough para retas

A transformada de Hough é uma técnica usada para localizar em imagens formas geométricas que podem ser descritas de forma matemática por uma curva paramétrica, por exemplo, retas, círculos e elipses. Esse método é bastante empregado em sistemas de navegação de robôs em ambientes internos que utilizam imagens digitais para detectar características do ambiente (BEZERRA, 2004). A principal vantagem dessa técnica é sua invariância a imperfeições das formas geométricas a serem detectadas, ou seja, a forma geométrica pode ser identificada mesmo que partes de suas bordas não estejam bem definidas (EISENKRAEMER, 2016).

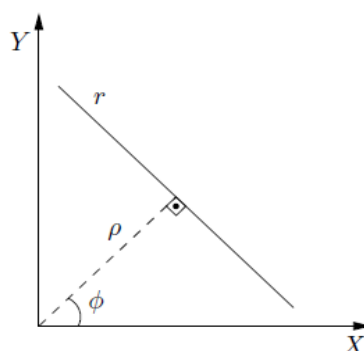
Essa transformada utiliza a curva paramétrica que foi definida para mapear cada ponto da imagem em um espaço de parâmetros. Dessa forma, os parâmetros calculados para cada ponto da imagem são acumulados no plano de parâmetros formando curvas. A forma geométrica é identificada a partir da determinação do ponto do espaço de parâmetros onde ocorrem a maior quantidade de intersecções entre as curvas (BEZERRA, 2004).

No processo de detecção de retas pela transformada Hough uma reta é descrita pela equação 31.

$$\rho = x \cos \phi + y \sin \phi \quad (31)$$

Onde ρ é a distância perpendicular da linha em relação a origem e ϕ é o ângulo entre o segmento e o eixo horizontal, como mostra a Figura 17.

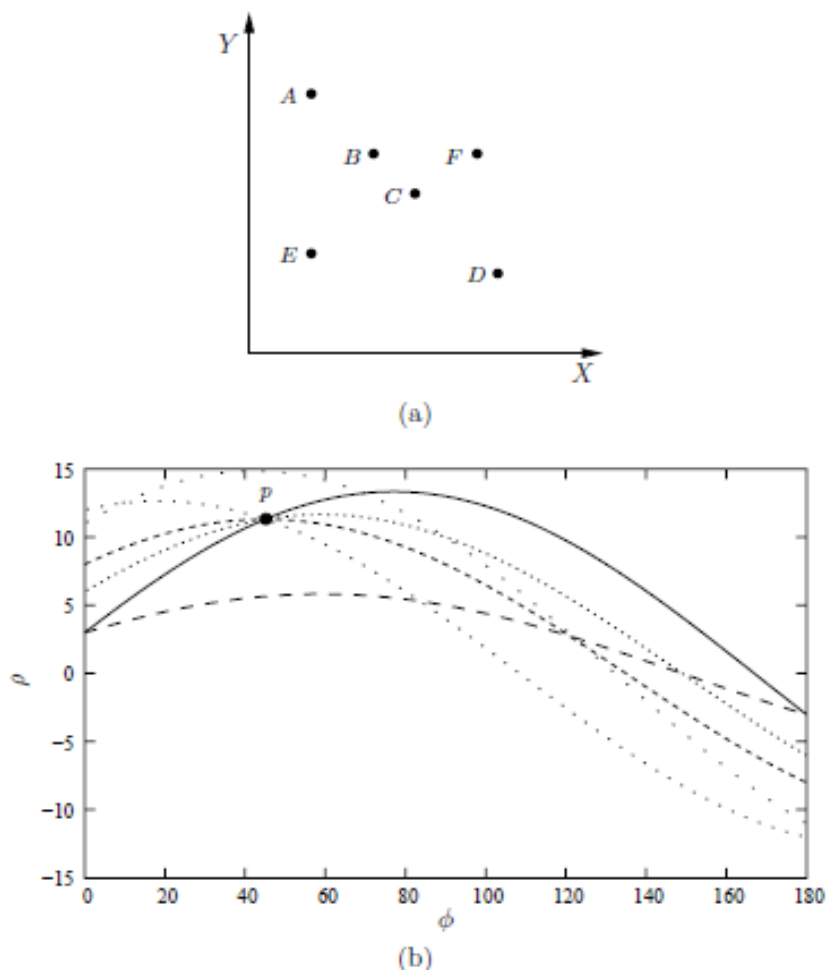
Figura 17 - Representação de uma reta



Fonte: Bezerra (2004).

Com essa curva paramétrica, cada ponto da imagem irá gerar um solenoide no espaço de parâmetros. Por exemplo, considerando o conjunto de pontos mostrados na Figura 18a, para o ponto A, a partir da equação 31 são determinados os parâmetros (ρ, ϕ) calculando os valores de ρ variando ϕ de 0 a 180 graus, esses parâmetros são então acumulados no espaço de parâmetros formando um solenoide. Esse processo é repetido para os pontos B, C, D e F, formando assim um conjunto de solenoides no espaço de parâmetros, como mostrado na Figura 18b. O ponto onde ocorre o maior número de interações entre os solenoides identifica a presença de uma reta na imagem (BEZERRA, 2004).

Figura 18 - Transformada de Hough para retas



Nota: a) Imagem com compostos que formam uma reta. b) Espaço de parâmetros obtido pela transformada de Hough.
Fonte: Bezerra (2004).

Como a transformada de Hough calcula os parâmetros para todos os pontos existentes em uma imagem, o processamento necessário para executar essa tarefa é bastante elevado. Para amenizar esse problema pode-se utilizar a transformada probabilística de Hough para retas, onde a curva é detectada aplicando a transformada em apenas uma parte dos pixels da imagem que são escolhidos baseados em uma função de densidade de probabilidade uniforme estabelecida em cima da imagem (BEZERRA, 2004).

A aplicação da transformada de Hough necessita, primeiramente, da realização de um pré-processamento na imagem digital, como a limiarização ou detector de bordas, a fim de reduzir o número de pixels a serem processados pela transformada (BEZERRA, 2004)

2.5.2 Detecção de bordas: Canny

Conforme Da Silva et al. (2004), um algoritmo de detecção de bordas consiste em um operador que localiza variações significativas na intensidade de tons de cinza dos pixels que compõem a imagem. Canny (1986) criou um detector de bordas que atendem a três condições que ele estabeleceu:

- a) baixa taxa de erro: possui baixa probabilidade de não detectar uma borda e de detectar uma borda falsa;
- b) localização: o ponto localizado deve estar próximo do centro da borda verdadeira;
- c) resposta: para uma borda apenas pode haver um máximo como resposta.

O filtro desenvolvido é aproximado pela primeira derivada da função gaussiana dada pela equação 32. Quanto maior for o desvio padrão, maior é o grau de suavização da imagem. Com isso, a imagem resultante fica mais borrada e suas bordas se tornam mais espessas, o que dificulta a sua localização exata. Essa é uma função separável, desta forma, pode ser aplicada em uma imagem primeiramente na horizontal e em seguida na vertical (DA SILVA et al., 2004).

$$G'(x) = -\frac{x}{\sigma^3\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}} \quad (32)$$

A posição exata de uma borda é o ponto cuja derivada é um máximo local, com isso, Canny também propôs a supressão não máxima, que é um processo que anula os pixels que na direção perpendicular à borda não são máximos locais. Essa técnica resulta no afinamento das bordas e conseqüentemente em uma melhor localização, além disso, faz com que permaneçam na imagem apenas os pontos que apresentam magnitude de grande intensidade (DA SILVA et al., 2004).

Para completar o processo de segmentação da imagem ainda é necessário realizar uma limiarização na imagem. Como ao longo de uma borda a magnitude dos pixels não se mantém constantes, é aplicada uma dupla limiarização chamada de histerese. Nesse processo é usado um limiar alto T_a e um limiar baixo T_b , todos os pontos com intensidade superior a T_a são classificados como bordas, formando um conjunto L_1 , os pontos com intensidade abaixo de T_b são descartados, e os pontos com intensidade entre T_a e T_b , formam um conjunto L_2 que é utilizado para completar as bordas do conjunto L_1 quando são verificadas descontinuidades (DA SILVA et al., 2004). O resultado da aplicação do detector de bordas de Canny pode ser verificado na Figura 19.

Figura 19 - Resultado da aplicação do detector de bordas de Canny



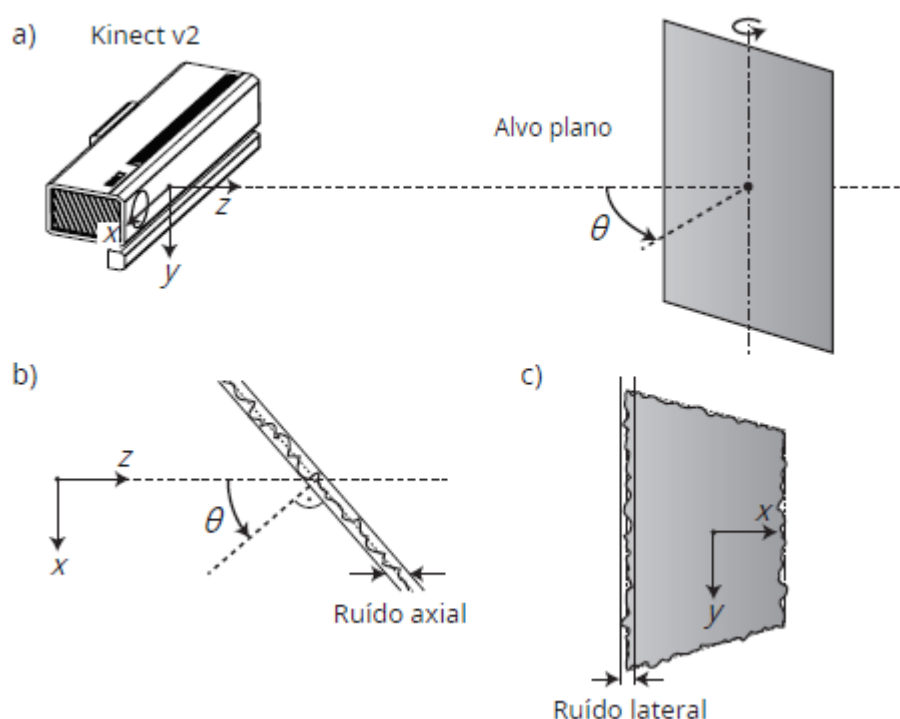
Fonte: Da Silva et al. (2004).

2.2.4 Modelo de ruído do Kinect v2

Fankhauser et al. (2015) modelaram empiricamente os ruídos do sensor Kinect v2 nas direções axial e lateral. O modelo de ruído proposto leva em consideração a distância de medição e o ângulo da superfície observada. Segundo Fankhauser et al. (2015) o modelo pode ser aplicado no pós-processamento para filtrar imagens de profundidade do sensor Kinect para diversas aplicações.

Para extrair os dados necessários para modelar o ruído axial e lateral foi colocado um plano rotativo na frente da câmera de forma que ele apareça no centro da imagem, e a partir disso foram realizadas medições em diferentes distâncias e ângulos de rotação. Para cada posição do alvo, o ruído foi determinado para cada pixel a partir de uma série de 100 imagens de profundidade. A Figura 20 exemplifica a configuração experimental.

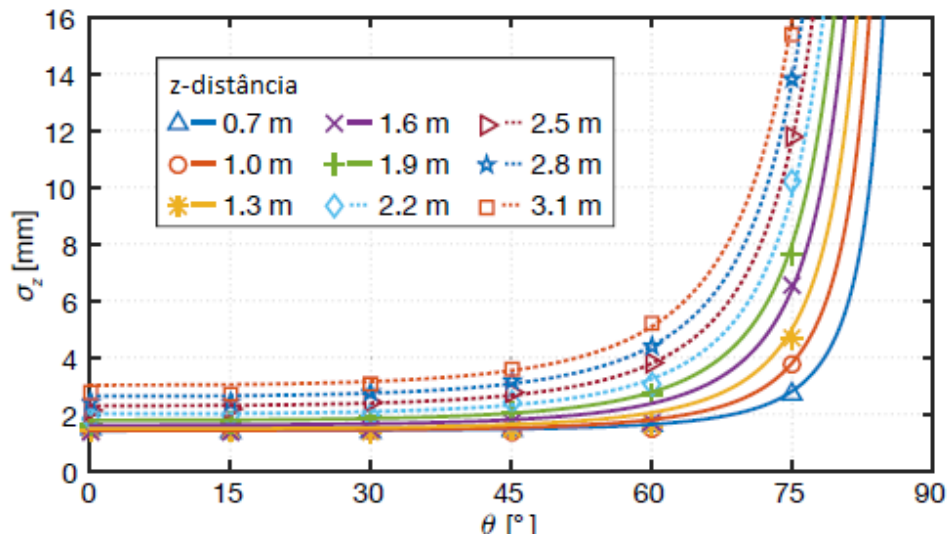
Figura 20 - Configuração experimental para avaliar o ruído axial e lateral



Nota: a) O sensor captura imagens de um plano posicionado em diferentes distâncias e ângulos. b) Vista superior do plano. O ruído axial é a variação das medidas de distância dentro do plano. c) O ruído lateral é a variação das distâncias medidas ao longo das bordas do plano. Fonte: Fankhauser et al. (2015).

O parâmetro de ruído axial foi definido como o desvio padrão para as medidas de tempo voo de pixels em uma região dentro do plano. A Figura 21 mostra o ruído axial em função da distância e do ângulo do plano obtido experimentalmente.

Figura 21 - Ruído axial em função da distância e do ângulo da superfície



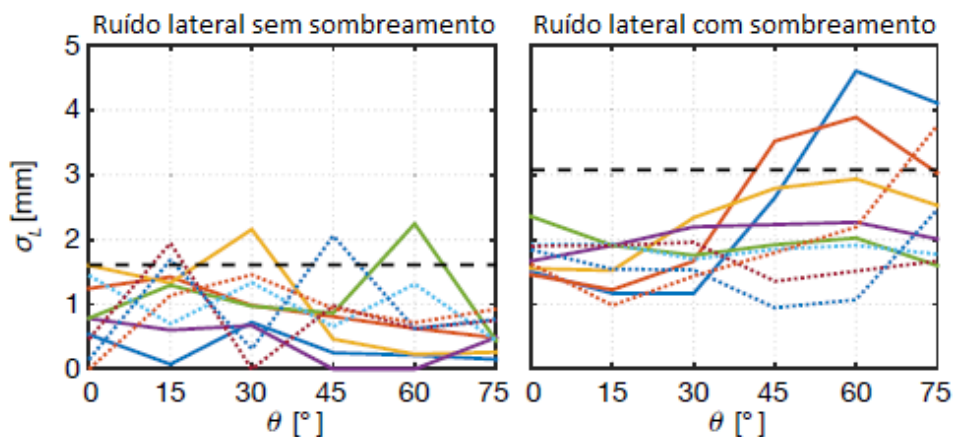
Fonte: Fankhauser et al. (2015)

Através dos resultados, os autores observaram que o desvio padrão aumenta com o quadrado da distância, e que para ângulos menores que 45 graus é menor que 4 mm, mas aumenta rapidamente para ângulos maiores que 45 graus. Com isso, o desvio padrão do ruído axial foi aproximado pela função quadrática 33.

$$\sigma_z(z, \theta)[mm] = 1,5 - 0,5z + 0,3z^2 + 0,1z^{\frac{3}{2}} \frac{\theta^2}{\left(\frac{\pi}{2} - \theta\right)^2} \quad (33)$$

O parâmetro de ruído lateral foi obtido por meio de medidas de tempo de voo em pixels ao longo das bordas verticais do plano em diferentes distâncias e ângulos. A partir dos resultados obtidos, os autores constataram que as medidas realizadas em bordas que são afetadas pelo sombreamento são mais ruidosas, desta forma o ruído lateral foi analisado separadamente no que diz a respeito à presença de sombreamento. A Figura 22 mostra o ruído lateral com sombreamento e sem sombreamento obtido para diferentes distâncias e ângulos.

Figura 22 - Ruído lateral em função da distância e ângulo da superfície



Fonte: Fankhauser et al. (2015)

Através dos resultados os autores concluíram que ruído lateral não possui nenhuma tendência evidente. Desta forma, sugeriram uma estimativa de ruído lateral constante, a equação 34 é o modelo de ruído lateral para bordas sem sombreamento e a equação 35 para bordas com sombreamento.

$$\sigma_L = 1,6 \text{ mm} \quad (34)$$

$$\sigma_L = 3,1 \text{ mm} \quad (35)$$

3 TRABALHOS RELACIONADOS

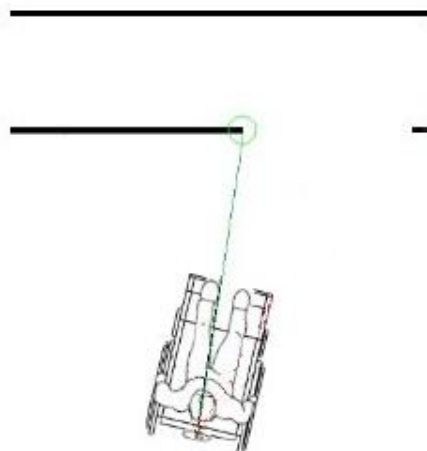
Tendo em vista os avanços da tecnologia, diversas técnicas de navegação autônoma de robôs móveis são abordadas na literatura. Neste capítulo serão apresentados diferentes trabalhos relacionados à navegação autônoma de cadeiras de rodas motorizadas e à localização de robôs móveis em ambientes internos.

Boschetti (2019) desenvolveu um sistema capaz de realizar a passagem de uma cadeira de rodas motorizada através de portas de forma segura. O sistema consiste na aquisição de imagens de profundidade de um sensor Kinect v2 para localizar a existência de uma porta e determinar a posição da cadeira de rodas em relação a essa porta. Durante a passagem, a posição da cadeira é obtida por meio da odometria por meio de *encoders* instalados no eixo das rodas, e o controle da trajetória é realizado utilizando um controlador *fuzzy*.

O processo de localização da cadeira é realizado por meio de um notebook que extrai imagens de profundidade do sensor Kinect e executa um algoritmo que realiza uma varredura buscando detectar as bordas da porta. O método utilizado para localizar bordas, consiste em realizar no centro da imagem de profundidade uma varredura da direita para esquerda, calculando a diferença de profundidade entre um pixel e seu subsequente. Foi estabelecida uma diferença de profundidade de no mínimo 0,5 metros entre pixels para se considerar o ponto uma possível borda, e quando a variação de profundidade é positiva a borda é candidata ao lado direito e quando a variação é negativa a borda é candidata ao lado esquerdo. Com os pontos de possíveis bordas extraídos, em seguida, é realizado um processo que seleciona os pares de bordas que satisfaçam uma série de condições para serem consideradas bordas de uma porta.

Também foi desenvolvido um método para identificar portas onde apenas um único ponto de descontinuidade pode ser detectado, como no caso ilustrado na figura 23. Quando não é encontrado um par de pontos que satisfaçam as condições de uma porta, é executado um algoritmo que busca a partir de uma descontinuidade o seu outro extremo, isso é feito encontrando o ponto que possui a menor distância em relação à descontinuidade. Após o reconhecimento da porta, a partir das coordenadas das bordas esquerda e direita, por meio de equações trigonométricas é determinado a posição da cadeira de rodas em relação ao centro da porta em coordenadas X, Y e θ .

Figura 23 - Porta com um ponto de descontinuidade



Fonte: Boschetti (2019).

Após a determinação da localização da cadeira em relação à porta, as coordenadas são enviadas para uma plataforma ESP-32, onde foi implementado o controlador *fuzzy* que controla a trajetória da cadeira durante a passagem pela porta. Para o funcionamento do controle é necessário conhecer a posição atual da cadeira de rodas durante a execução da trajetória, para isso, foi utilizado a odometria, que permite determinar a posição por meio da medição do deslocamento das rodas, que é obtido através de *encoders* acoplados nos eixos dos motores.

Para o controle foi utilizado o controlador Fuzzy de Mamdani, que é construído com base em um conjunto de regras ou implicações do tipo *if-then*. As funções de pertinência e as regras de inferência foram desenvolvidas com o uso da ferramenta *Fuzzy Logic Designer* do software Matlab. A saída do controlador é a velocidade angular que a cadeira deve atingir, desta forma, o controlador atua apenas na rotação enquanto a velocidade linear é mantida constante.

A informação da velocidade angular da saída do controlador é convertida em valores de 0 a 255, que correspondem proporcionalmente às saídas analógicas de 0 a 3,3 V da plataforma ESP-32, e essa informação analógica é enviada para o *driver* de acionamento dos motores da cadeira de rodas. O controle da velocidade angular é realizado em malha aberta, desta forma, foi determinado experimentalmente a velocidade angular atingida pela cadeira para valores de 0 a 255. O autor constatou que a velocidade da cadeira possui uma resposta diferente para quando ela está

acelerada do que quando está desacelerando, com isso, foi utilizado o valor médio entre as velocidades.

Com base nos resultados obtidos, o autor observou uma diferença na trajetória quando a cadeira precisa realizar uma rotação para esquerda do que quando precisa realizar para a direita, conforme o próprio autor isso pode ter ocorrido devido às rodas da cadeira não se deslocarem com a mesma velocidade linear. Outro fator que pode ter prejudicado é a tensão de saída que representa a velocidade angular, pois a tensão necessária para se atingir determinada velocidade é diferente para quando está acelerando do que quando está desacelerando. Mas mesmo assim, o controlador consegue corrigir a trajetória e atingir o ponto de passagem no centro da porta. Em relação a detecção e localização das portas, o autor conclui que os resultados obtidos foram satisfatórios, condizentes ao esperado por um protótipo.

No trabalho de Vieira (2019), é relatado o desenvolvimento de um sistema de navegação autônomo para uma cadeira de rodas motorizada, no qual o usuário informa o destino através de um display *touch screen*. Em seu trabalho todo o processamento do sistema é realizado por meio da plataforma embarcada Raspberry Pi 3 B+, e um microcontrolador ESP32 é utilizado para envio dos comandos de acionamento para o drive dos motores da cadeira.

A partir do destino informado e um mapa discreto do ambiente previamente carregado no sistema, é traçada uma rota utilizando o algoritmo de Dijkstra. Após, o deslocamento da cadeira pelo ambiente de navegação é controlado por meio de um controlador *fuzzy*.

A localização e os dados necessários para executar o controle da cadeira são obtidos através da odometria, que é realizada utilizando dois *encoders* ópticos acoplados ao eixo do motor de cada roda. Os dados adquiridos por meio da técnica empregada são linearizados aplicando filtro de Kalman, com o intuito de garantir que valores não lineares sejam devidamente considerados.

Como a odometria é um método que acumula erros durante o deslocamento, se fez necessário a implementação de um sistema de localização absoluta, que utiliza um sensor de espectro RGB para detectar marcos artificiais. Esses marcos são constituídos por placas de piso tátil com cores específicas, que são distribuídas pelo ambiente de navegação em pontos críticos, como na entrada do ambiente e no início de cada corredor.

Após a coleta de dados, a correção da posição e da trajetória é efetuado pelo controlador *fuzzy* que atua no acionamento dos motores permitindo um correto deslocamento da cadeira. Para o controlador *fuzzy*, foi definido um conjunto de regras que possibilitam controlar a velocidade linear a partir do monitoramento da distância percorrida e o restante da distância até o destino final, também foi definido um conjunto de regras para realizar apenas movimentos angulares sob o eixo, e um conjunto de regras que executam simultaneamente o controle da velocidade angular e linear, para realizar curvas de 90 graus.

Durante o movimento linear, o autor verificou que existe uma diferença entre as velocidades lineares das rodas, o que causa um movimento angular quando o movimento deveria ser apenas retilíneo. Para solucionar o problema, foi definido um conjunto de regras *fuzzy* para realizar a compensação da diferença de velocidades entre as rodas.

As informações fornecidas pela saída do controlador são enviadas via comunicação serial para a ESP32, onde o valor é convertido para um sinal analógico, que é então enviado para o *driver* nativo da cadeira de rodas. O controlador da cadeira possui dois canais, um serve para a velocidade linear e o outro para velocidade angular, desta forma foi necessário medir as tensões que precisam ser aplicadas nos canais para simular o funcionamento do joystick da cadeira.

Ao fim dos testes, o autor conclui que o sistema de navegação utilizando a odometria a partir de sensores do tipo *encoder* óptico, juntamente com uma técnica de correção da posição baseada na detecção de marcos artificiais, é válida em aplicações de navegação autônomo de cadeira de rodas motorizados em ambientes internos conhecidos.

Ganganath e Leung (2012) desenvolveram um método preciso e de baixo custo para a localização de robôs móveis usando odometria e um sensor Kinect. A odometria é utilizada para rastrear qualquer movimento arbitrário do robô, e é realizada a partir da medição da rotação das rodas por meio de *encoders*. O sensor Kinect é usado para estimar a posição do robô em relação à marcos de referência, que são compostos por diferentes círculos coloridos espalhados pelo ambiente. Os pontos de referência são detectados em quadros de imagem RGB aplicando a transformada de Hough, e são diferenciados uns dos outros usando o modelo de cores HSI. A odometria e as medidas obtidas pelo sensor Kinect são fundidas usando o filtro

de Kalman estendido (EKF) e o filtro de partículas (PF), a fim de minimizar a incerteza na localização do robô.

Conforme os autores, o sistema proposto utilizando a odometria em conjunto com o sensor Kinect, apesar de seu baixo custo, a precisão da localização é comparável com a maioria dos métodos baseados em odometria de última geração.

Li e Huang (2018) exploraram o uso de um sensor Kinect para detectar pontos de referência baseados em código QR e determinar a posição de um robô em um ambiente interno. Os códigos QR são espalhados pelo ambiente, e cada um possui um identificador que contém as informações de localização de referência. A detecção dos códigos é realizada a partir do processamento de imagens RGB capturadas pelo Kinect usando a biblioteca *Open Source Computer Vision Library (OpenCV)*.

A imagem é primeiramente convertida para escala de cinza, e em seguida, é aplicado o algoritmo de Canny para extrair as bordas e reduzir a quantidade de dados a serem processados. Após é utilizado um algoritmo de localização de hierarquia de contornos, disponibilizado pela biblioteca *Opencv*, para destacar o padrão de detecção de posição dos códigos QR na imagem.

Depois de localizados o código QR na imagem, é empregado o algoritmo de código aberto *Zbar*, para decodificar e extrair as informações de localização de referência. Em seguida, por meio de imagens de profundidade, obtidas pelo Kinect, é determinada a distância do sensor em relação ao código QR.

Os autores concluíram a partir dos experimentos realizados, que o ângulo de medição impacta na precisão dos dados de profundidade, quanto mais próximo da região central da câmera mais precisos são os dados coletados. Além disso, perceberam que existem erros nas medidas que ocorrem devido ao desalinhamento do sensor de profundidade e a câmera RGB do Kinect.

4 DESENVOLVIMENTO DO TRABALHO

Este trabalho visa o desenvolvimento de um sistema de navegação autônoma de uma cadeira de rodas motorizada entre corredores e portas em ambientes previamente conhecidos, utilizando a odometria por meio de *encoders* nas rodas, acelerômetro e giroscópio, juntamente com um sensor Kinect para determinar a localização da cadeira no ambiente.

O processo de navegação do sistema desenvolvido inicia-se com o usuário informando o destino por meio de uma interface em um Notebook, ainda fora do ambiente. Em seguida, a trajetória que a cadeira deve realizar até a porta da sala selecionada é gerada aplicando o algoritmo de Dijkstra em um mapa discreto do ambiente.

Após, a posição inicial da cadeira no ambiente é determinada através da localização de um marco artificial de referência, que é composto por um retângulo colocado na entrada do ambiente em uma posição conhecida. A localização do marco é realizada por meio do processamento de imagens RGB-D capturadas pelo Kinect.

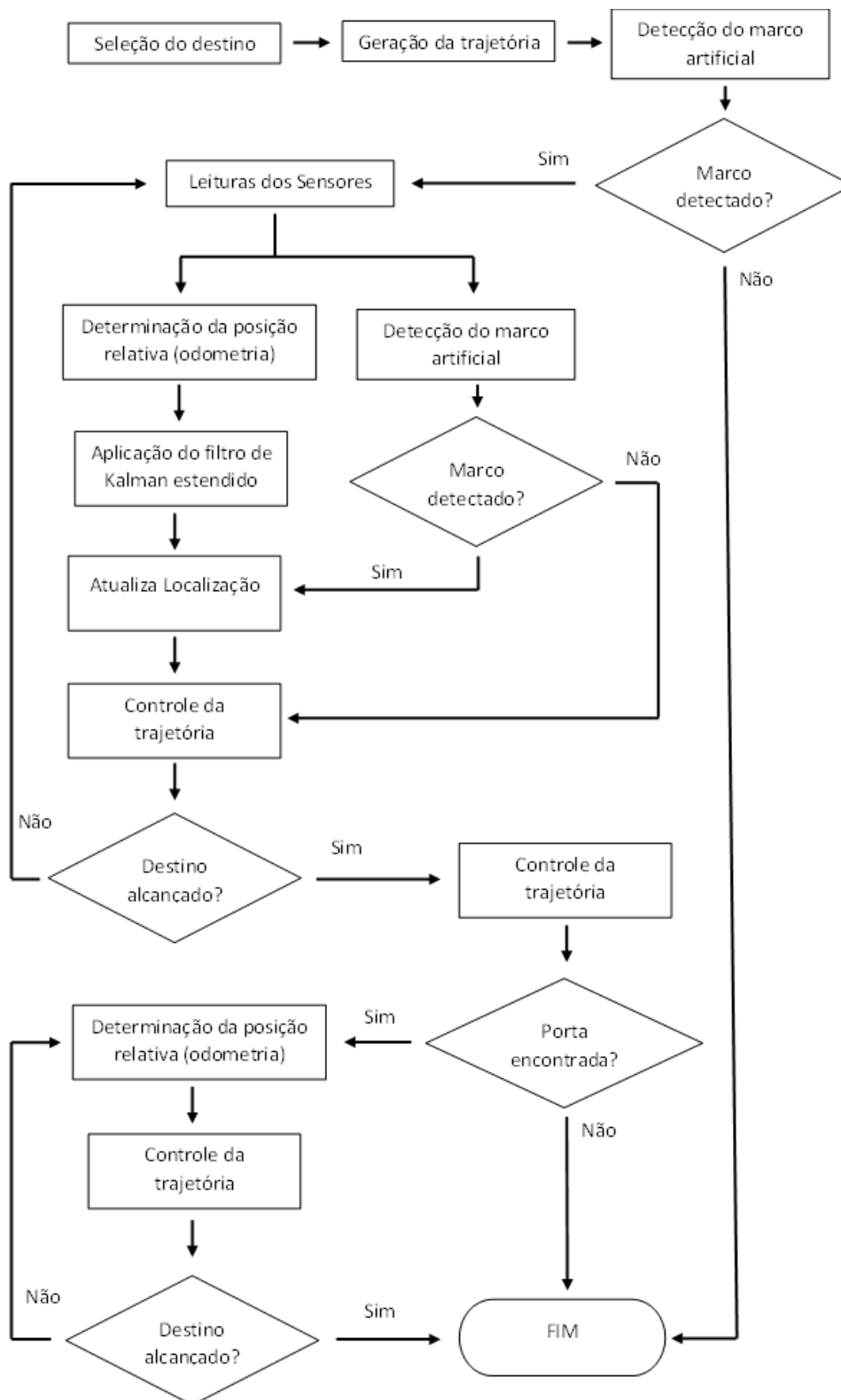
O deslocamento da cadeira se dá por meio de dois controladores baseados em regras *fuzzy que* controlam as velocidades linear e angular, fazendo com que a cadeira siga a trajetória definida. Durante a trajetória, a posição relativa da cadeira é estimada por meio da odometria, que é realizada utilizando dois *encoders* acoplados nos eixos dos motores das rodas, um giroscópio e um acelerômetro. Os dados da posição obtidos pela odometria são filtrados aplicando o filtro de Kalman estendido.

A fim de corrigir o erro acumulado pela odometria, o sistema de navegação desenvolvido conta com um sistema de localização absoluta que busca encontrar marcos artificiais por meio de imagens RGB-D. Esses marcos são formados por retângulos vermelhos espalhados pelo ambiente de navegação em posições conhecidas.

Quando a cadeira chega próximo da porta da sala informada pelo usuário, o sistema de passagem por portas é inicializado. O processo de passagem por portas faz uso das imagens de profundidade obtidas através do Kinect para determinar a posição da cadeira em relação à porta. Em seguida, por intermédio das informações obtidas pela odometria, se realiza a passagem da cadeira pela porta. Caso nenhuma porta seja encontrada, o sistema de navegação é interrompido.

Com intuito de esclarecer o funcionamento do sistema desenvolvido, a Figura 24 apresenta o fluxograma completo.

Figura 24 - Fluxograma do sistema de navegação autônoma implementado



Fonte: autor (2021).

A seguir são apresentados o *hardware* utilizado e as metodologias para a geração da trajetória, localização absoluta e relativa da cadeira, e controle da trajetória da mesma no ambiente.

4.1 HARDWARE

O equipamento eletrônico utilizado para esse projeto contém uma câmera Microsoft Kinect v2 instalada na horizontal e sobre a cadeira de rodas a uma altura de 1,5 metros do chão, para capturar imagens RGB-D que são utilizadas para detectar portas e marcos artificiais no ambiente de navegação. O Kinect requer uma alimentação de 12V e a cadeira utilizada opera com duas baterias ligadas em série que totalizam 24V, com isso foi utilizado um conversor de tensão DC-DC *buck-boost* com corrente máxima de saída de 3A para reduzir os 24V para 12V.

O processamento de dados é realizado utilizando um Notebook, por possuir uma maior capacidade de processamento e também porque o Kinect v2 necessita de uma interface USB 3.0 para comunicação. O notebook utilizado possui um processador i5 7200U e 4GB de memória RAM com sistema operacional Linux Ubuntu 20.04.3 LTS.

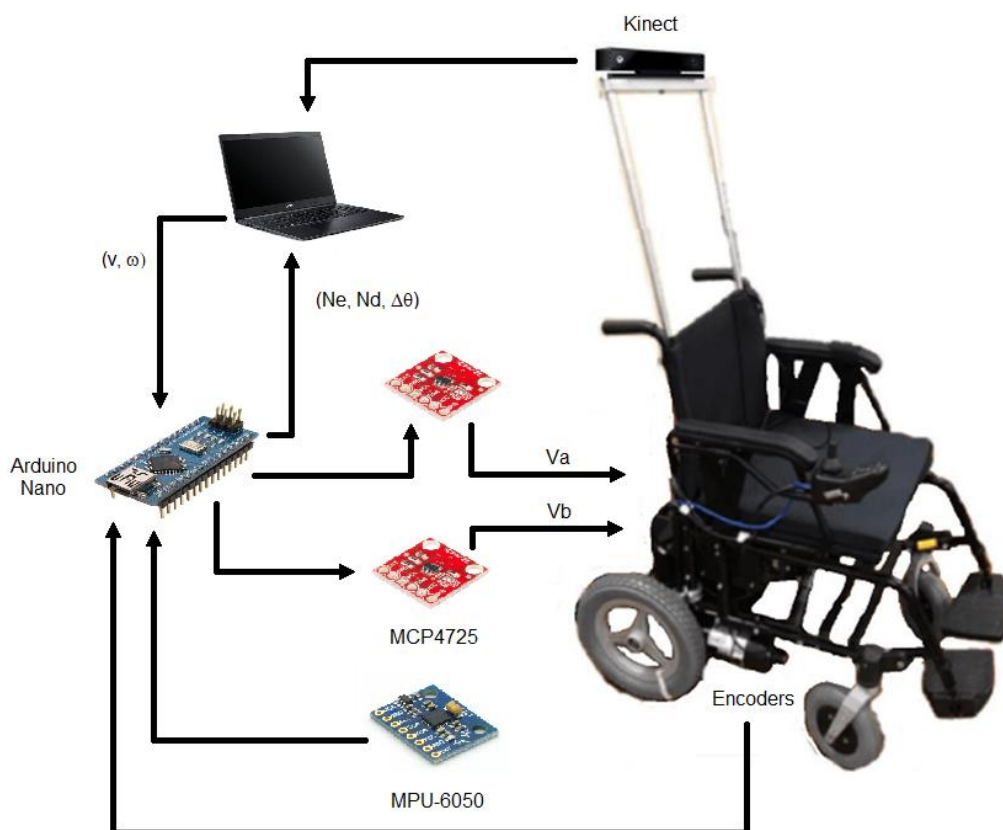
Inicialmente, a odometria seria realizada apenas através das leituras de *encoders* instalados nas rodas. Posteriormente, verificou-se a necessidade da utilização de um giroscópio e de um acelerômetro para medir o deslocamento angular, a fim de diminuir o erro da estimativa da posição angular causados pelas imperfeições do chão e escorregamento das rodas. Com isso, foi utilizado um módulo MPU-6050, que possui um acelerômetro e um giroscópio de 3 eixos.

Os *encoders* foram instalados no eixo dos motores de cada roda, os *encoders* utilizados possuem dois canais, A e B, o que permite determinar o sentido de rotação das rodas, e possuem uma resolução de 360 pulsos por volta. Entretanto a rotação da roda e do eixo do motor possui uma relação 1:32, desta forma a cada volta realizada pela roda são contabilizados 11520 pulsos.

A leitura das informações dos sensores instalados na cadeira e o envio dessas informações para o notebook é efetuado por um Arduino Nano. O Arduino também é responsável por receber do notebook a velocidade linear e angular que a cadeira deve atingir e realizar o controle das velocidades com realimentação. Como o Arduino não possui conversores digitais analógicos que são necessários para enviar os comandos

de acionamento ao *driver* da cadeira de rodas, também foi utilizado dois DACs MCP4725 que se comunicam com o Arduino via I2C. A Figura 25 mostra a integração dos componentes e o fluxo dos dados.

Figura 25 - Integração dos componentes



Fonte: autor (2021).

4.2 PROJETO DO CONTROLE DE VELOCIDADE

O acionamento dos motores é realizado utilizando o *driver* nativo da cadeira de rodas, que possui dois canais de controle de movimento, um para movimento linear e outro para angular. Para projetar o controlador de velocidade com realimentação, primeiramente foi necessário conhecer os níveis de tensões que devem ser aplicados nos canais de controle para simular o *joystick*. A Tabela 1 mostra os níveis de tensão dos canais de controle obtidos. Vale ressaltar que estes valores foram obtidos utilizando o conversor digital analógico MCP4725.

Tabela 1 - Níveis de tensão dos canais de controle

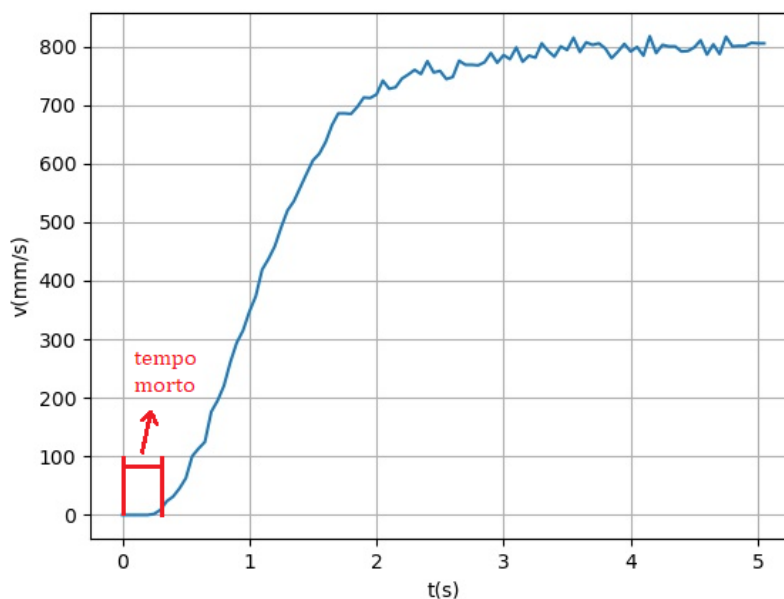
CANAL A		CANAL B	
Faixa de tensão (mV)	Movimento	Faixa de tensão (mV)	Movimento
2350 - 2850	parado	2050 - 2500	parado
2850 - 4000	ré	2500 - 4000	direita
1100 - 2350	frente	1200 - 2050	esquerda

Fonte: autor (2021).

Foi utilizado um compensador PI para efetuar o controle da velocidade. Para projetar os ganhos, K_p e K_i , primeiramente foi necessário obter as funções de transferência do sistema a ser controlado. A obtenção da função de transferência é feita adquirindo a curva da velocidade em função do tempo ao aplicar um degrau de tensão nos canais de controle. A velocidade é adquirida medindo a rotação das rodas por meio dos *encoders* acoplados nos eixos dos motores da cadeira. Por meio da leitura dos *encoders* a velocidade linear individual de cada roda é calculada através das equações 9 e 10 apresentada na seção 2.2.1. A partir da velocidade de cada roda a velocidade linear e angular são determinadas por meio das equações 1 e 2.

A figura 26 mostra a curva da velocidade linear ao aplicar um degrau de 700mV no canal A.

Figura 26 - Velocidade linear ao aplicar um degrau



Fonte: autor (2021).

A partir da curva obtida, constata-se que o sistema pode ser aproximado por uma função de transferência de segunda ordem, e também que possui uma resposta superamortecida uma vez que não apresenta sobre-sinal.

A função de transferência característica de um sistema de segunda ordem é dada pela equação 36. Os pólos do sistema podem ser determinados pelas equações 37 e 38, desta forma a função de transferência pode ser escrita conforme a equação 39. A resposta do sistema superamortecido no domínio do tempo é descrita pela função mostrada na equação 40, onde a constante A é a amplitude do degrau aplicado.

$$\frac{X(s)}{F(s)} = \frac{K \cdot \omega_n^2}{s^2 + 2\xi \cdot \omega_n \cdot s + \omega_n^2} \quad (36)$$

$$p_1 = \xi \cdot \omega_n + \omega_n \sqrt{\xi^2 - 1} \quad (37)$$

$$p_2 = \xi \cdot \omega_n - \omega_n \sqrt{\xi^2 - 1} \quad (38)$$

$$\frac{X(s)}{F(s)} = \frac{K \cdot p_1 \cdot p_2}{(s + p_1)(s + p_2)} \quad (39)$$

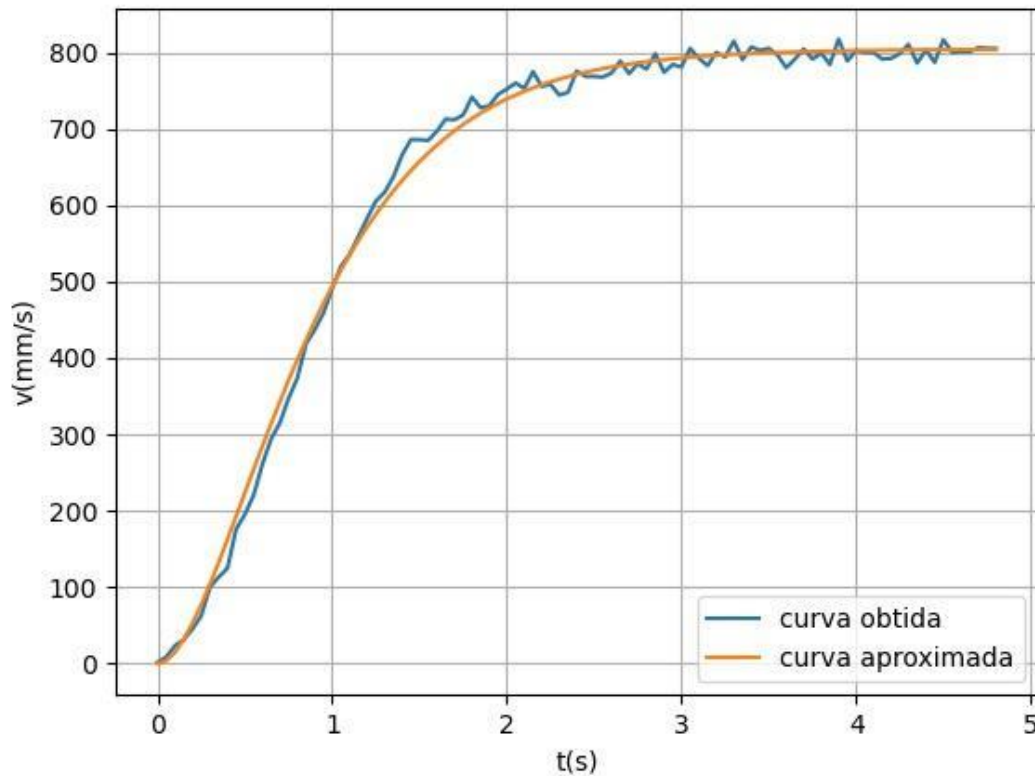
$$X(t) = A \cdot K \left(1 + \frac{p_1}{p_2 - p_1} e^{-p_2 \cdot t} - \frac{p_2}{p_2 - p_1} e^{-p_1 \cdot t} \right) \quad (40)$$

A partir da função da equação 40 e da curva da velocidade obtida experimentalmente, foi escrito um *script* em python para calcular os polos dos sistemas e o ganho K. A rotina utiliza o método *curve_fit* da biblioteca *scipy*, esse método ajusta as constantes da função da equação 40 de forma a se obter uma curva aproximada dos dados obtidos experimentalmente. Através dos parâmetros retornados é possível obter a função de transferência do sistema.

A Figura 27 mostra a curva da velocidade linear ao aplicar um degrau de 700mV e a curva aproximada pelo algoritmo de otimização. É importante salientar que a resposta do sistema apresenta um tempo morto, que é o atraso entre o instante em que o degrau é aplicado e o momento em que o sistema começa a responder. O tempo morto pode ser observado na Figura 26. Para a aquisição da função de transferência

o tempo morto foi desconsiderado. A equação 41 mostra a função de transferência obtida para a velocidade linear.

Figura 27 - Resposta da velocidade linear a um degrau



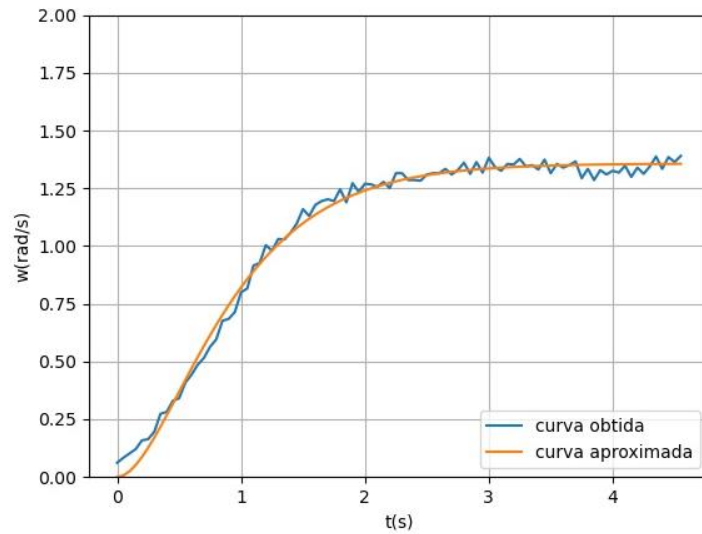
Fonte: autor (2021).

$$\frac{V(s)}{F(s)} = \frac{5,0197}{s^2 + 4,1512s + 4,308} \quad (41)$$

Para a velocidade angular observou-se que ao aplicar um degrau com a mesma amplitude para os dois sentidos, a velocidade angular para a esquerda era aproximadamente o dobro maior do que para a direita. A fim de se obter uma dinâmica de controle de velocidade angular semelhante para ambos os sentidos, foi projetado um controlador PI para cada direção. Desta forma, foi necessário determinar a função de transferência para ambos os sentidos.

A Figura 28 mostra a curva da velocidade angular para esquerda ao aplicar um degrau de 250mV e a curva aproximada. A função de transferência obtida é dada pela equação 42.

Figura 28 - Resposta da velocidade angular para esquerda a um degrau

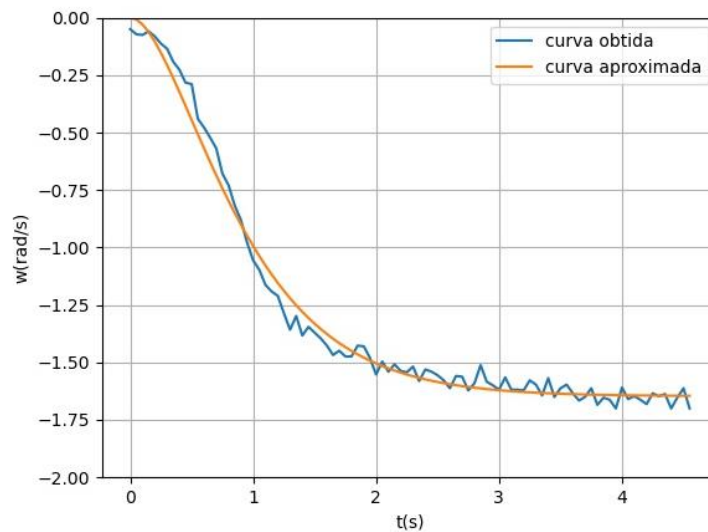


Fonte: autor (2021).

$$\frac{V(s)}{F(s)} = \frac{0,02455}{s^2 + 4,071s + 4,1616} \quad (42)$$

A Figura 29 mostra a curva da velocidade angular para direita ao aplicar um degrau de -500mV e a curva aproximada. O módulo da amplitude do degrau aplicado no ensaio para direita é maior do que o para a esquerda a fim de diminuir a diferença do módulo da velocidade entre os dois ensaios. A função transferência obtida para a velocidade angular para a direita é dada pela equação 43.

Figura 29 - Resposta da velocidade angular para direita a um degrau



Fonte: autor (2021).

$$\frac{V(s)}{F(s)} = \frac{0,01366}{s^2 + 4,069s + 4,1392} \quad (43)$$

Como pode-se observar as funções de transferência da velocidade angular para direita e para esquerda são semelhantes, a diferença está principalmente no ganho K. O que explica porque é necessário aplicar uma tensão maior para atingir uma determinada velocidade angular para o lado direito do que para o lado esquerdo.

A partir das funções de transferência obtidas, os ganhos, Kp e Ki, dos controladores PI foram projetados utilizando a ferramenta *Control System Designer* do MatLab. A Tabela 2 mostra os ganhos dos controladores PI projetados.

Tabela 2 - Ganhos dos controladores PI projetados

	Linear	Angular Esquerda	Angular Direita
Kp	0,583	118	207
Ki	0,882	174	314

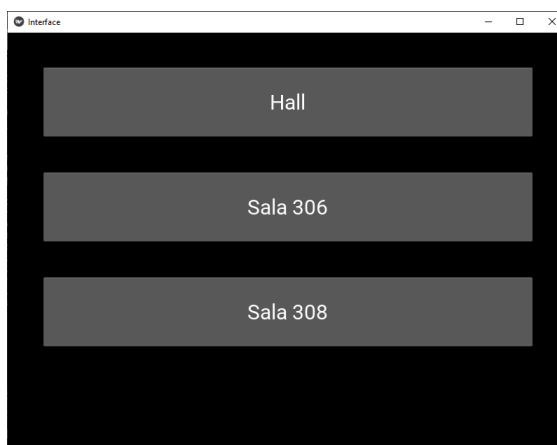
Fonte: autor (2021).

4.3 GERAÇÃO DE TRAJETÓRIA

Para executar a navegação da cadeira é necessário definir a trajetória que deve ser seguida. A geração de trajetória consiste em determinar o caminho mais curto a ser percorrido pelo robô a partir da localização atual e um ponto de destino. A definição da trajetória inicia-se com o usuário selecionando o destino em uma interface no notebook desenvolvida em linguagem python e Kivy. A Figura 30 exibe a interface de seleção do destino.

Os botões da interface são construídos dinamicamente a partir dos destinos armazenados nas informações do mapa, desta forma, para adicionar novos destinos no sistema basta apenas acrescentá-los no mapa.

Figura 30 - Interface de seleção do destino

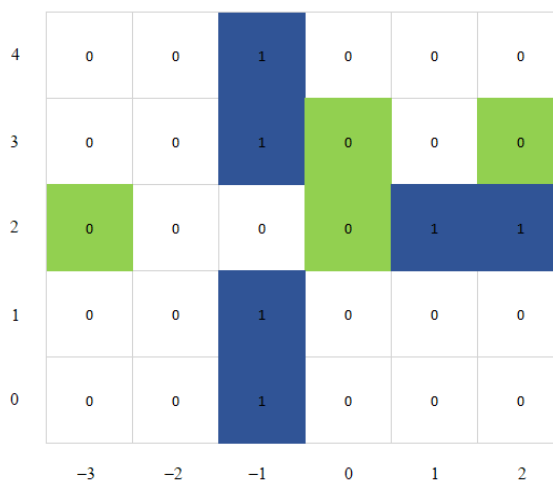


Fonte: autor (2021)

Após o destino ser selecionado, é utilizado o algoritmo de Dijkstra apresentado na seção 2.3 para gerar a trajetória a partir de um mapa de grade armazenado no sistema em um arquivo XLSX. O algoritmo de geração de trajetória retorna a sequência de coordenadas que a cadeira deve seguir.

Por exemplo, supondo o mapa de grade da Figura 31, onde as células marcadas com 1 representam um objeto e as marcadas com 0 representam que o espaço está vazio. Considerando que a coordenada (-3, 2) seja a posição de partida e a coordenada (2, 3) o destino, o algoritmo retorna as coordenadas [(-3, 2), (0, 2), (0, 3), (2, 3)] que estão marcadas em verde na figura. Esses pontos representam a sequência de coordenadas em que a cadeira deve se deslocar.

Figura 31 - Exemplo de um mapa de grade



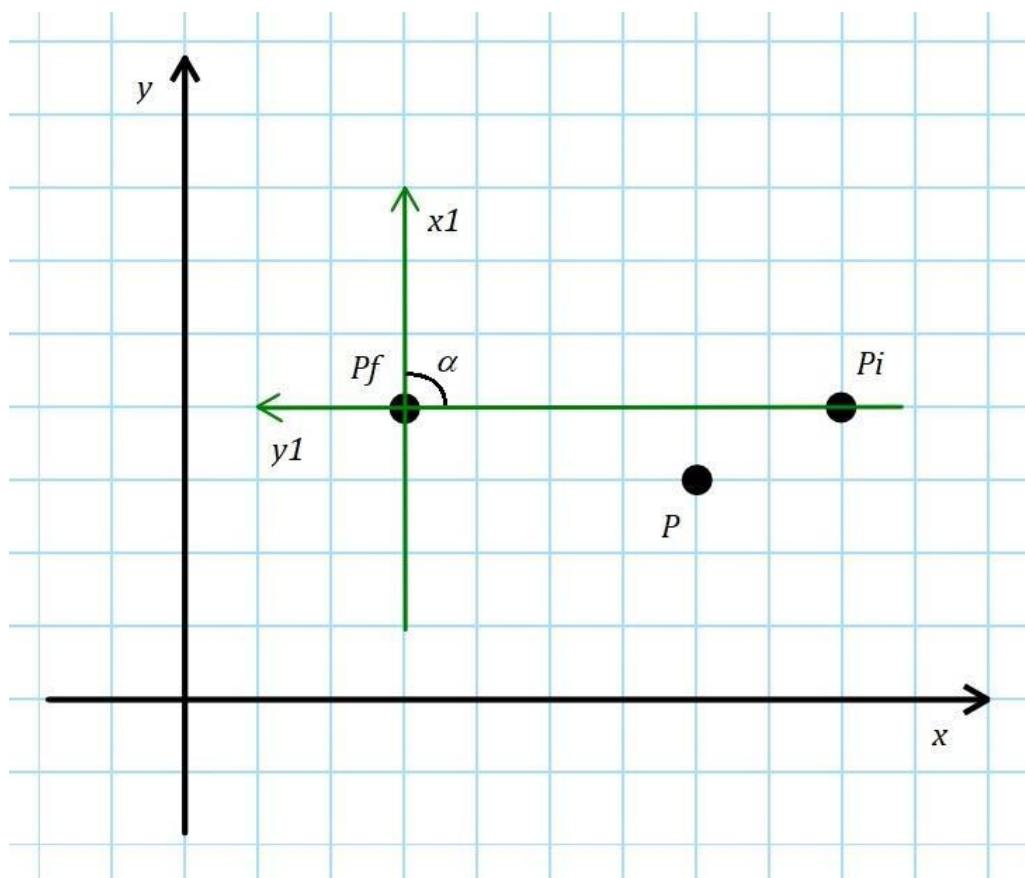
Fonte: autor (2021).

4.4 CONTROLE DE TRAJETÓRIA

O controle de trajetória consiste em fazer com que a cadeira se desloque de forma a seguir os segmentos de retas formados pelas coordenadas retornadas pelo algoritmo de geração de trajetória. Para efetuar o controle, para cada segmento de reta é calculada a posição da cadeira em um novo sistema de coordenadas, onde a posição que a cadeira deve chegar é a origem, e o eixo y negativo passa pela posição inicial da trajetória.

A Figura 32 exemplifica o cálculo do novo sistema de coordenadas utilizado para realizar o controle de trajetória. Na figura, os eixos x e y correspondem ao sistema de coordenadas do mapa do ambiente, e os eixos x_1 e y_1 correspondem ao novo sistema de coordenadas, P é a posição da cadeira no ambiente, P_i e P_f correspondem respectivamente a coordenada inicial e final do segmento de reta que a cadeira deve seguir.

Figura 32 - Sistema de coordenadas do controle de trajetória



Fonte: autor (2021).

A posição da cadeira no novo sistema de coordenadas pode ser calculada através das equações 44 e 45, onde α é o ângulo entre o eixo x e x1 que pode ser obtido por meio da equação 46.

$$X1 = (X_p - X_{pf}) \cos(\alpha) + (Y_p - Y_{pf}) \sin(\alpha) \quad (44)$$

$$Y1 = -(X_p - X_{pf}) \sin(\alpha) + (Y_p - Y_{pf}) \cos(\alpha) \quad (45)$$

$$\alpha = \arctan2(Y_{pi} - Y_{pf}, X_{pi} - X_{pf}) + \frac{\pi}{2} \quad (46)$$

A localização da cadeira no ambiente também é composta por uma componente que representa a orientação em relação ao eixo x positivo. A orientação em relação ao eixo x1 do novo sistema de coordenadas é calculada por meio da equação 47.

$$\theta1 = \theta_p - \alpha \quad (47)$$

As componentes do novo sistema de coordenadas são utilizadas para alimentar os controladores que são responsáveis por realizar a trajetória. Foram utilizados dois controladores para executar esta tarefa, um que controla a velocidade linear em função da distância até o destino e um que controla a velocidade angular a partir da orientação da cadeira. O controlador utilizado foi o controlador Fuzzy de Mamdani, que funciona a partir de um conjunto de regras do tipo *if-then*.

O primeiro passo para o projeto do controlador foi a definição das variáveis linguísticas das entradas e saídas dos controladores. A Tabela 3 mostra as variáveis linguísticas definidas para o controle. No controlador de velocidade linear a entrada é posição y em mm e a saída é a velocidade em mm/s, já no controlador de velocidade angular a entrada é a orientação da cadeira em graus e a saída é a velocidade angular em rad/s. Vale observar que caso o valor de entrada do controlador esteja fora do intervalo da variável, o valor é considerado igual ao valor do limite do intervalo da variável.

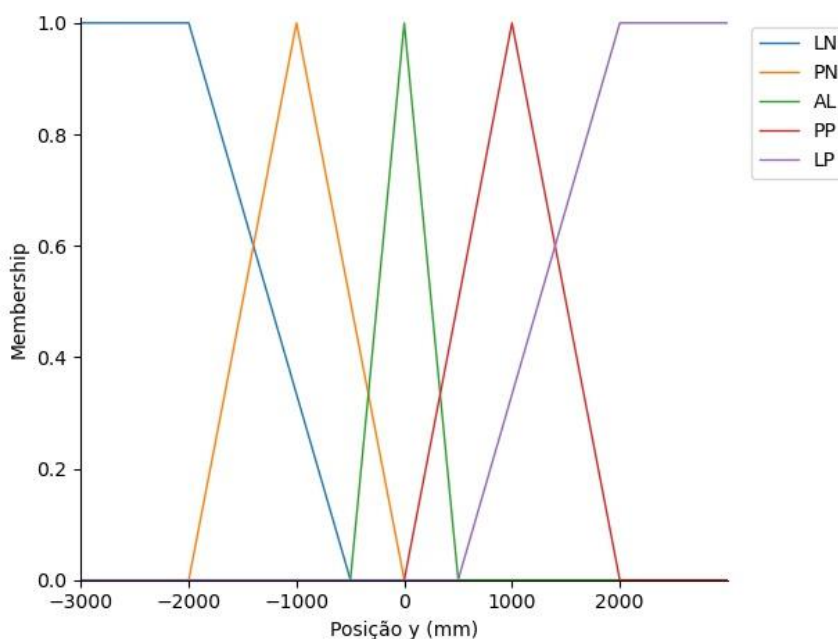
Tabela 3 - Variáveis linguísticas

Controlador	Entrada/Saída	Variável	Intervalo
Velocidade Linear	Entrada	Posição y	[-3000 a 3000] mm
	Saída	Velocidade linear	[-700 a 700] mm/s
Velocidade Angular	Entrada	Orientação	[45 a 135] graus
	Saída	Velocidade Angular	[-0,8 a 0,8] rad/s

Fonte: autor (2021).

A variável linguística da posição y que representa a distância que a cadeira está do destino é definida de -3000 a 3000 metros, essa variável é constituída por cinco grupos sendo eles: (LN) Longe Negativo, (PN) Perto Negativo, (AL) Alcançado, (PP) Perto Positivo e (LP) Longe Positivo. A Figura 33 mostra as funções de pertinência para a posição y.

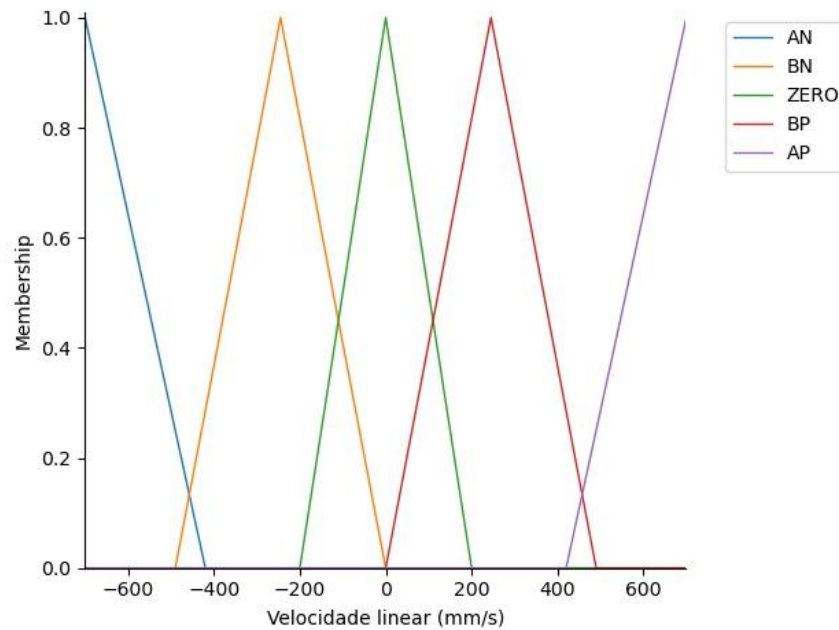
Figura 33 - Funções de pertinência para a posição y



Fonte: autor (2021).

A variável linguística da velocidade linear é definida no intervalo de -700 a 700 mm/s, ela é constituída por cinco grupos: (AN) Alto Negativo, (BN) Baixo Negativo, (ZERO) Zero, (BP) Baixo Positivo e (AP) Alto Positivo. A Figura 34 exibe as funções de pertinência para a velocidade linear.

Figura 34 - Funções de pertinência para a velocidade linear



Fonte: autor (2021).

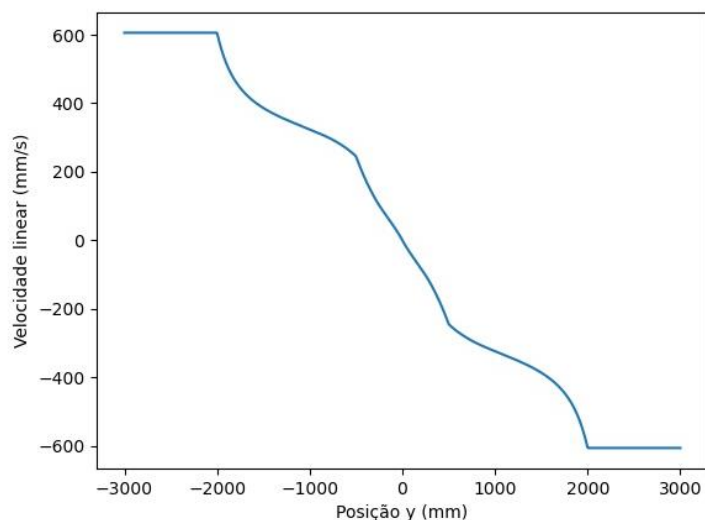
O conjunto de regras *fuzzy* que regem o controle de velocidade linear são mostradas pela Tabela 4. O objetivo do controlador projetado é manter uma velocidade linear constante quando a cadeira estiver longe do destino e reduzir a velocidade a partir de uma distância de 2 metros até a cadeira atingir o destino. Além disso, caso a cadeira ultrapasse a posição do destino, o controlador também é capaz de movimentar a cadeira para trás até que o objetivo seja alcançado. A resposta do controlador projetado para a velocidade linear pode ser vista na Figura 35.

Tabela 4 - Conjunto de regras para a velocidade linear

Se posição y	Então velocidade linear
LN	AP
PN	BP
AL	ZERO
PP	BN
LP	AN

Fonte: autor (2021).

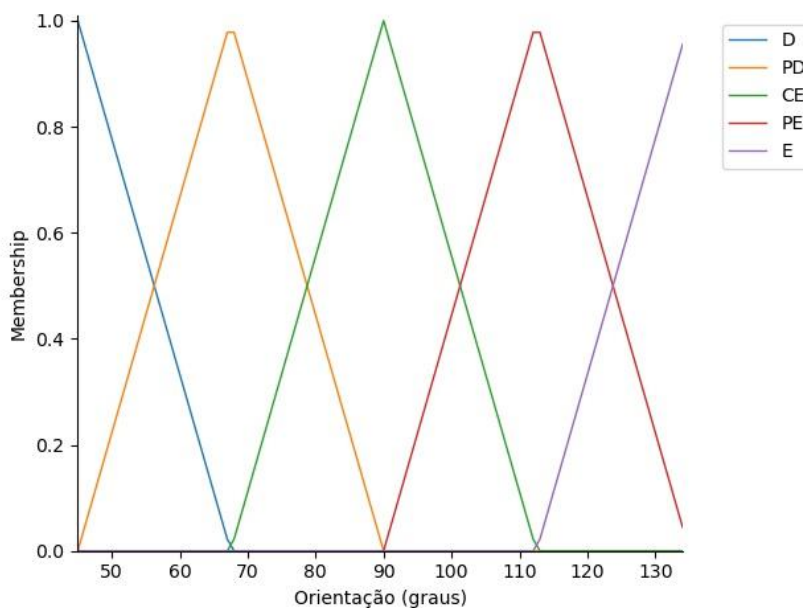
Figura 35 - Resposta do controlador de velocidade linear



Fonte: autor (2021).

A variável linguística da orientação que representa o ângulo da cadeira em relação ao eixo x é definida de 45 a 135 graus. Essa variável possui cinco grupos: (D) Direita, (PD) Pouco Direita, (CE) Centro, (PE) Pouco Esquerda, (E) Esquerda. As funções de pertinência para a orientação são mostradas na Figura 36.

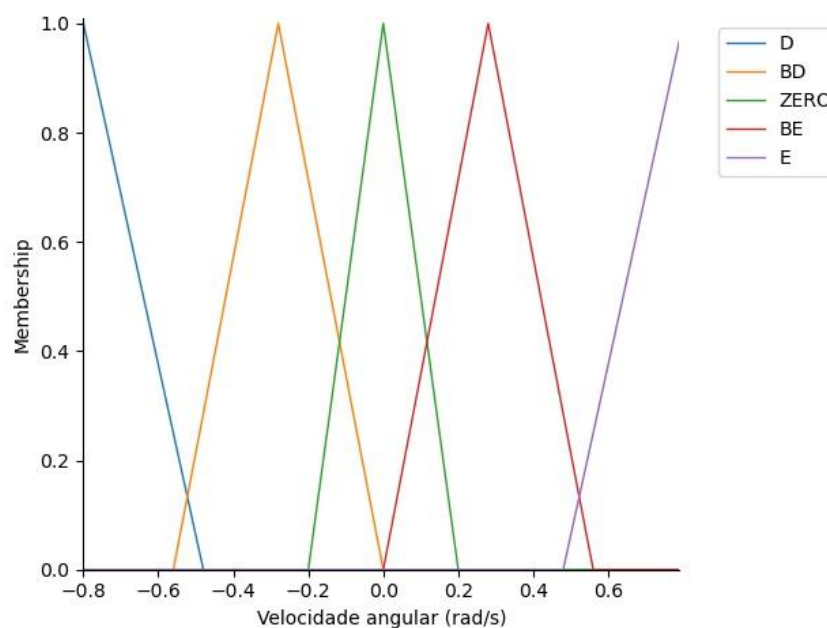
Figura 36 - Funções de pertinência para a orientação



Fonte: autor (2021).

A variável linguística da velocidade angular é definida no intervalo de -0,8 a 0,8 rad/s e é constituído por cinco grupos: (D) Direita, (BD) Baixo Direita, (ZERO) Zero, (BE) Bastante Esquerda, (E) Esquerda. A Figura 37 mostra as funções de pertinência para a velocidade angular.

Figura 37 - Funções de pertinência para a velocidade angular



Fonte: autor (2021).

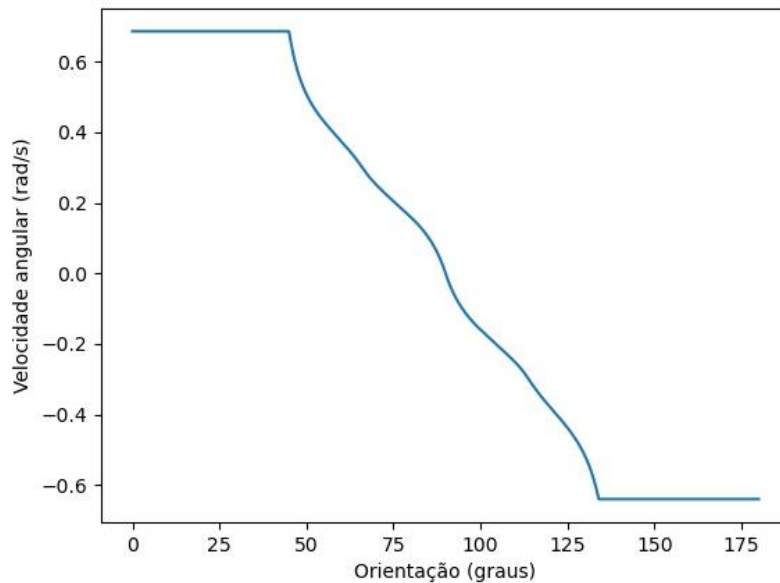
O conjunto de regras *fuzzy* que compõem o controlador de velocidade angular são exibidas na Tabela 5. A finalidade do controlador é ajustar a orientação da cadeira de forma a mantê-la em 90° em relação ao eixo x positivo, desta forma fazendo com que a cadeira se movimente paralelamente ao eixo y. A resposta do controlador de velocidade angular projetado pode ser vista na Figura 38.

Tabela 5 - Conjunto de regras para a velocidade angular

Se orientação	Então velocidade angular
D	E
PD	BE
CE	ZERO
PE	BD
E	D

Fonte: autor (2021).

Figura 38 - Resposta do controlador de velocidade linear



Fonte: autor (2021).

Com a utilização apenas desses controladores a cadeira se desloca de forma paralela com o eixo y, mas não corrige a posição no eixo x onde ela deve ficar o mais próximo de zero. Para corrigir a trajetória no eixo x é acrescentado um ângulo de compensação no valor do ângulo que alimenta a entrada do controlador de velocidade angular. Esse ângulo de compensação é proporcional à posição da cadeira no eixo x, e é permitido um ângulo de compensação entre -90° a 90° , com isso, caso o valor esteja fora do intervalo é atribuído o valor limite para o ângulo de compensação.

O ângulo de compensação é calculado pela equação 48 e o ângulo que alimenta a entrada do controlador de velocidade angular é dado pela equação 49. A constante $-0,045$ indica que para uma distância x_1 maior que 2000mm o ângulo de compensação é -90° o que faz com que a cadeira se posicione em 180° em relação ao eixo x positivo e se desloque paralelamente ao eixo x indo no sentido do eixo y, e para uma distância menor de 2000mm a cadeira se desloca realizando uma curva com um raio de 2000mm até se alinhar paralelamente com o eixo y.

$$\theta_c = -0,045 x_1 \quad (48)$$

$$\theta_{controle} = \theta_1 + \theta_c \quad (49)$$

A partir desses controlados a cadeira desloca-se seguindo os segmentos de retas que formam a trajetória. Na transição de um segmento de reta para outro, a cadeira antes de seguir o segmento ela realiza uma curva de 90°. A curva é realizada utilizando apenas o controlador de velocidade angular enquanto que a velocidade linear é mantida em zero. Quando a cadeira se alinha em 90° com o eixo x é inicializada a trajetória pelo segmento de reta.

4.5 ODOMETRIA

Inicialmente, a posição relativa da cadeira no ambiente seria estimada a partir da técnica de odometria descrita na seção 2.2.1, onde o deslocamento da cadeira é medido apenas por meio de *enconders* acoplados nos eixos dos motores das rodas. Como forma de reduzir os erros sistemáticos presentes nessa técnica, buscou-se calibrar os parâmetros da odometria aplicando o método UMBmark, descrito da seção 2.2.2. Mas infelizmente devido às características deste trabalho a calibração não gerou resultados positivos.

Um dos motivos que invalidam a calibração pelo método UMBmark, é que a cadeira não possui um controlador de velocidade individual em cada roda, com isso a cadeira não realiza curvas de 90 graus sem deslocar o centro do eixo. Outro fator, são as rodas dianteiras da cadeira que são livres. Após realizar uma curva de 90 graus, o posicionamento das rodas dianteiras juntamente com o fato de o sistema não possuir controle de velocidade individual em cada roda, faz com que a cadeira se desloque para o lado quando deveria realizar um movimento retilíneo. A cadeira possui controle de trajetória que corrige os deslocamentos indesejados para os lados, mas os movimentos para ajustar posição durante a trajetória retilínea prejudicam a metodologia de calibração. A seguir é descrito o processo de calibração realizado e os resultados obtidos.

Antes de realizar a calibração, foi necessário medir a distância entre as rodas da cadeira e o raio das rodas. Utilizando uma fita métrica estimou-se uma distância entre as rodas de 565mm e um perímetro das rodas aproximadamente 1000mm, o que resulta em um raio de 159,15mm.

Com a técnica UMBmark é possível corrigir a distância entre as rodas e a diferença de tamanho entre as rodas, mas não é possível corrigir o raio médio das rodas. Com isso, primeiramente realizou-se um ensaio para obter uma estimativa

melhor do raio médio das rodas. O ensaio consiste em a cadeira executar cinco deslocamentos de 10 metros e medir a distância real percorrida, e em seguida a partir da média das medidas é calculada um fator de correção que é aplicado no raio médio estimado. A Tabela 6 mostra as medidas obtidas com uma fita métrica com resolução de 2mm.

Tabela 6 - Medidas para estimativa do raio médio das rodas

Amostra	Distância medida (m)
1	9,885
2	9,894
3	9,884
4	9,900
5	9,905
Média	9,894

Fonte: autor (2021).

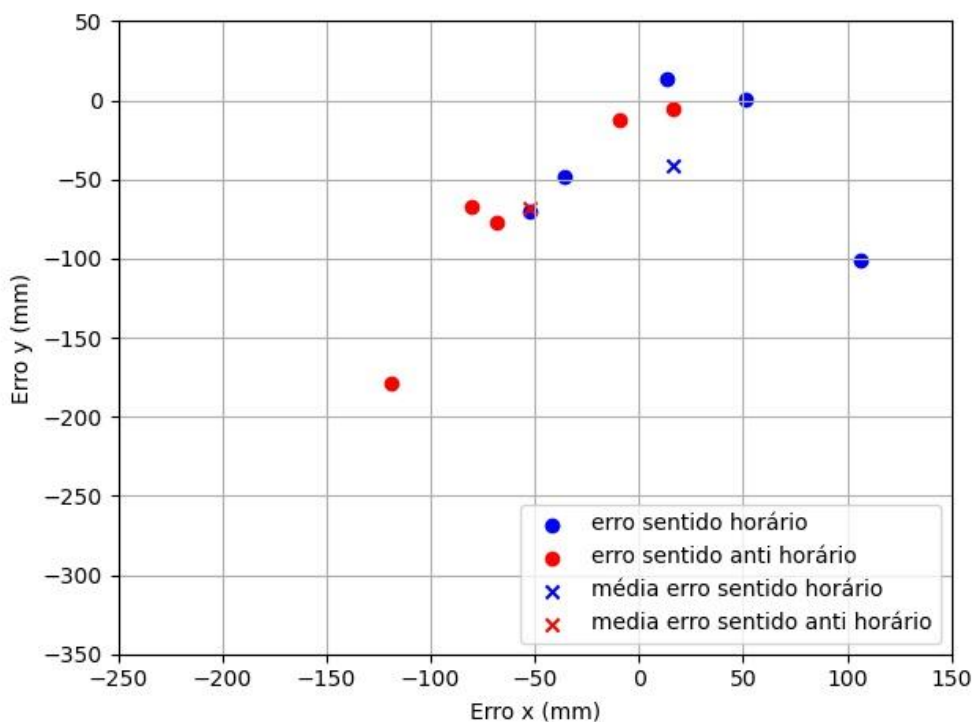
O fator de correção do raio médio é calculado a partir da equação 50, onde obteve-se um fator de 0,9894. Já o novo raio médio é calculado por meio da equação 51 na qual resulta em um novo raio igual a 157,46mm.

$$FC = \frac{\textit{distância média medida}}{10} \quad (50)$$

$$Rm_{\textit{novo}} = FC . Rm \quad (51)$$

Após a determinação dos parâmetros iniciais da odometria, foi realizado o processo de calibração por meio da metodologia UMBmark, onde a cadeira é colocada para executar trajetórias em forma de um quadrado de 4x4 metros, sendo cinco trajetórias no sentido horário e cinco no sentido anti-horário, e em cada trajetória realizada é medida a diferença entre a posição registrada pela odometria e a posição real. As medidas e o erro médio obtido em cada sentido obtidos na primeira etapa são mostradas através do gráfico de dispersão da Figura 39. As medidas foram obtidas utilizando uma trena com resolução de 1mm.

Figura 39 - Medidas obtidas na primeira etapa do ensaio UMBmark



Fonte: autor (2021).

Na primeira etapa obteve-se um erro euclidiano de 44,27mm no sentido horário e 85,93mm no sentido anti-horário. A partir dos erros médios obtidos em ambos os sentidos, são calculados os novos parâmetros da odometria utilizando as equações mostradas na seção 2.2.2. Os novos parâmetros calculados são mostrados na Tabela 7.

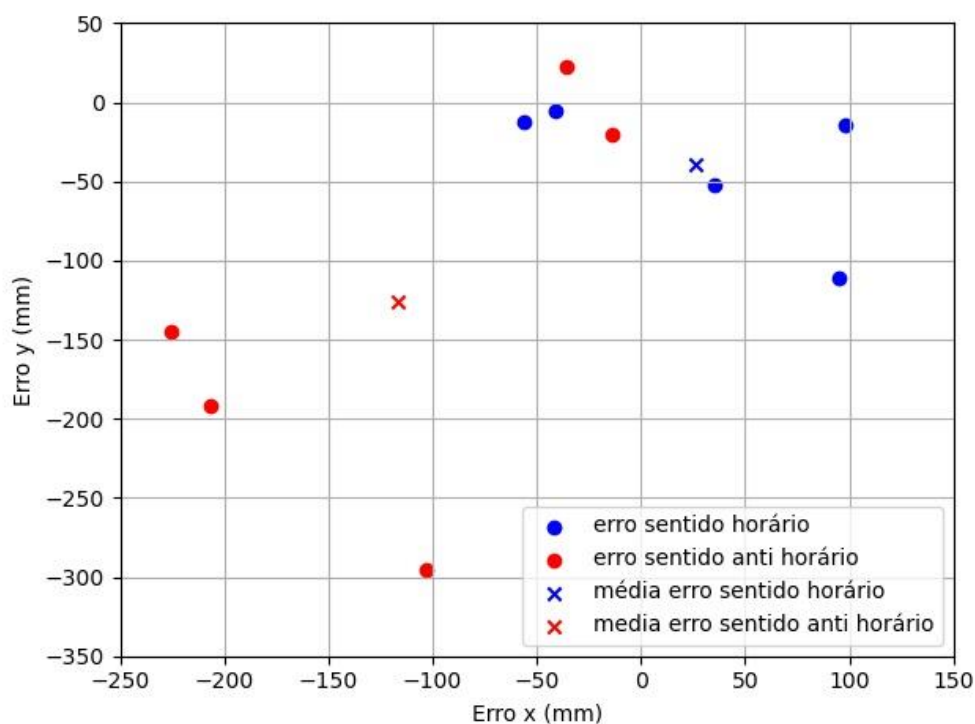
Tabela 7 - Novos parâmetros após primeira etapa de calibração

PARÂMETRO	VALOR (mm)
Distância entre as rodas	565,01
Raio da roda esquerda	157,41
Raio da rida direita	157,51

Fonte: autor (2021).

Utilizando os novos parâmetros calculados o ensaio de obtenção do erro é realizado novamente. A Figura 40 mostra os resultados obtidos na segunda etapa da calibração.

Figura 40 - Medidas obtidas na segunda etapa do ensaio UMBmark



Fonte: autor (2021).

Na segunda etapa obteve-se um erro euclidiano de 47,21mm no sentido horário e 171,69mm no sentido anti-horário, ou seja, o erro obtido é maior que o erro encontrado na etapa anterior, o que indica que a metodologia UMBmark não é válida para as características desse projeto.

Desta forma, para a odometria foram utilizados os parâmetros estimados inicialmente uma vez que os erros encontrados com esses parâmetros são aceitáveis. Entretanto constatou-se que com a técnica de odometria utilizada, não era possível executar uma trajetória retilínea nos corredores por muito tempo sem colidir com a parede. A causa principal do problema é o erro acumulado na estimativa da orientação causado pelo escorregamento das rodas e pelas imperfeições do chão. Como a posição da cadeira em coordenadas x e y são calculadas em função do ângulo de orientação, os erros na estimativa da orientação acabam gerando grandes erros na estimativa da posição da cadeira ao se deslocar por grandes distâncias.

Como forma de melhorar a estimativa da orientação acrescentou-se ao sistema de odometria um acelerômetro e um giroscópio de três eixos. Com isso, a variação da orientação é estimada combinando as medidas dos dois sensores empregando filtro complementar. Com o uso desses sensores, os escorregamentos das rodas e as

imperfeições do chão não causam erros na estimativa da orientação da cadeira. Além disso, os erros na estimativa da posição causados pela diferença do tamanho das rodas e pelo erro na determinação do comprimento do eixo são eliminados, uma vez que na nova técnica o cálculo da posição depende apenas do raio médio das rodas.

Com o novo método de odometria a orientação da cadeira é calculada através da equação 52, onde $\Delta\theta$ é variação da orientação entre amostras medida com o acelerômetro e giroscópio, sendo que o período de amostragem é de 50 ms.

$$\theta(t + \Delta t) = \theta(t) + \Delta\theta \quad (52)$$

Para comparar as duas técnicas de odometria foi realizado um ensaio onde a cadeira foi colocada para realizar uma trajetória retilínea de 10 metros partindo da posição (0, 0) com orientação de 90°, com isso a posição final esperada era (0, 10000), em mm. Ao final do deslocamento a posição real em que a cadeira parou foi medida utilizando uma trena com resolução de 2mm. O ensaio foi realizado 5 vezes para cada técnica de odometria, a Tabela 8 mostra posição final medida em cada ensaio.

Tabela 8 - Comparação entre técnicas de odometria

ENSAIO	TÉCNICA DE ODOMETRIA ANTERIOR		TÉCNICA DE ODOMETRIA NOVA	
	x (mm)	y (mm)	x (mm)	y (mm)
1	72	10050	-30	10044
2	-25	10040	6	10040
3	-138	10046	-2	10042
4	-110	10042	-24	10060
5	-318	10038	-26	10040

Fonte: autor (2021).

Como pode-se observar a odometria utilizando apenas encoders, apresentou grandes erros na estimativa da posição da cadeira no eixo x, sendo que no pior caso o erro foi de 318mm. Já na odometria utilizando *encoders* juntamente com um acelerômetro e um giroscópio a posição final real da cadeira se aproximou mais da posição esperada, apresentando um erro máximo no eixo x de 30mm. Com isso conclui-se que com a nova técnica de odometria é possível realizar trajetórias retilíneas entre corredores a distâncias maiores sem que a cadeira colida.

4.6 AQUISIÇÃO DE IMAGENS RGB-D

As imagens RGB-D foram capturadas utilizando a versão em python da biblioteca de código aberto *freenect2* da *OpenKinect*. A biblioteca foi instalada em um sistema operacional baseado em Linux seguindo o tutorial disponível no *GitHub* na página da *OpenKinect*.

Por meio da biblioteca são adquiridas as imagens RGB e de profundidade. A imagem RGB possui uma resolução de 1920x1080 enquanto que a imagem de profundidade possui uma resolução de 512x424, ou seja, as imagens não são alinhadas. O alinhamento é feito utilizando o método *apply(rgb,depth)* que retorna a imagem RGB alinhada com a imagem de profundidade. A Figura 41 mostra um exemplo de imagem RGB e de profundidade adquirida utilizando a biblioteca *freenect2*.

A imagem de profundidade é representada por uma imagem em escala de cinza, onde a tonalidade do pixel é proporcional à distância, desta forma os objetos mais próximos aparecem na imagem em tom mais escuro enquanto que os objetos mais distantes aparecem em tom mais claro. Tanto na imagem RGB quanto na imagem de profundidade os pixels totalmente em preto são pixels em que a câmera Kinect não conseguiu registrar a medida de profundidade. Vale observar que as imagens RGB-D capturas são espelhadas horizontalmente com a imagem real do ambiente.

Figura 41 - Exemplo de imagem RGB e de profundidade



Para obter as informações de profundidade de cada pixel, a biblioteca *frenect2* disponibiliza o método *get_points_xyz_array(undistorted)*. Este método retorna a distância de cada pixel em relação ao Kinect em coordenadas cartesianas x, y e z com uma resolução de 1mm.

4.7 SISTEMA DE LOCALIZAÇÃO ABSOLUTA.

A odometria é uma técnica que acumula erros ao longo do tempo. Para corrigir a localização da cadeira foi desenvolvido um sistema de localização absoluta através de marcos artificiais que são detectados por meio de imagens RGB-D capturadas com o Kinect. Os marcos são compostos por retângulos vermelhos com dimensões de 297 mm de largura e 210mm de altura, que são colocados na parede a uma altura de aproximadamente 800mm em relação ao chão.

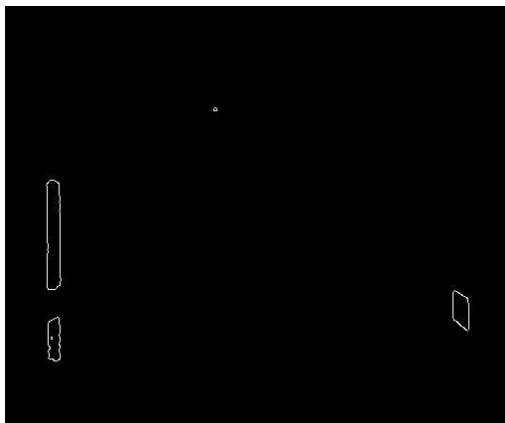
Para realizar o processamento das imagens RGB foi utilizado a biblioteca de código aberto *OpenCV* disponível também em linguagem python. A detecção do retângulo iniciasse convertendo a imagem RGB para a escala de cor HSV. Em seguida é aplicado um processo de limiarização por cor, onde os pixels que estão dentro de uma faixa de tom de vermelho são convertidos para a cor branca e os demais para a cor preta. O intervalo de cor que deve ser limiarizado foi determinado através de uma amostra de uma imagem do quadrado, onde obteve-se o intervalo de (0, 84, 100) a (186, 255, 255) no modelo de cor HSV. A Figura 42 mostra um exemplo de imagem obtida após aplicar a limiarização.

Figura 42 - Imagem após limiarização



Após é aplicado o detector de bordas de Canny para realçar as bordas do retângulo e reduzir a quantidade de informação a ser processada. A Figura 43 exibe a imagem após a aplicação do detector de bordas de Canny.

Figura 43 - Imagem após aplicar o detector de borda



Fonte: autor (2021).

Em seguida é utilizado a transformada probabilística de Hough para retas para localizar as retas que podem pertencer ao retângulo. A transformada retorna as coordenadas do pixel inicial e final da reta encontrada. Em algumas situações a transformada pode retornar várias retas onde deveria localizar apenas uma, o que prejudica o algoritmo de localização do marco. Para diminuir esse problema foi desenvolvido um algoritmo que mescla as retas que estão próximas. O resultado do algoritmo de detecção de retas pode ser visto na figura 44, onde as retas encontradas são mostradas em verde.

Figura 44 - Retas encontrada pela transformada de Hough



Fonte: autor (2021).

A partir das retas encontradas pela transformada de Hough e das informações de profundidade dos pixels em coordenadas cartesianas são agrupadas as retas horizontais e verticais que podem pertencer ao marco. O agrupamento é realizado verificando o comprimento, o ângulo em relação ao plano xz, e a posição no eixo y das retas encontradas.

Sendo p_1 e p_2 a posição em coordenadas cartesianas dos pixels extremos de uma reta em relação a câmera, é calculado um vetor \vec{v} por meio da equação 53 que representa a distância entre os pixels em coordenadas cartesianas. O comprimento da reta pode ser calculado através da equação 54 e o ângulo da reta em relação ao plano xz é obtido por meio da equação 55.

$$\vec{v} = p_2 - p_1 \quad (53)$$

$$|\vec{v}| = \sqrt{X_v^2 + Y_v^2 + Z_v^2} \quad (54)$$

$$\theta = \arctan2(Y_v, \sqrt{X_v^2 + Z_v^2}) \quad (55)$$

São consideradas como possíveis retas horizontais pertencentes ao marco as retas com comprimento entre 150 e 307mm e com um ângulo em relação ao plano xz entre -10 e 10 graus. E são consideradas como possíveis retas verticais as retas com comprimento entre 105 e 220mm e com um ângulo em relação ao plano xz entre 80 e 100 graus. Como os marcos foram colocados a uma altura de 800mm do chão e o Kinect foi posicionado a uma altura de 1500mm, as retas, tanto horizontais quanto verticais, devem estar em uma posição em y entre -500 e -290mm.

Em seguida é realizada uma varredura verificando a distância entre uma reta vertical e sua subsequente mais próxima. Os pares de retas que apresentam uma distância entre 287 e 307mm são candidatos a pares de bordas verticais do marco. Após é realizado uma varredura procurando pares de retas horizontais que estão localizados entre os pares de retas verticais encontrados anteriormente. Para que duas retas horizontais sejam consideradas como candidatas a bordas do marco elas devem possuir uma distância entre 200 e 220mm. A Figura 45 mostra um exemplo de retas encontradas pelo algoritmo, onde os pontos em azul mostram o centro das retas encontradas.

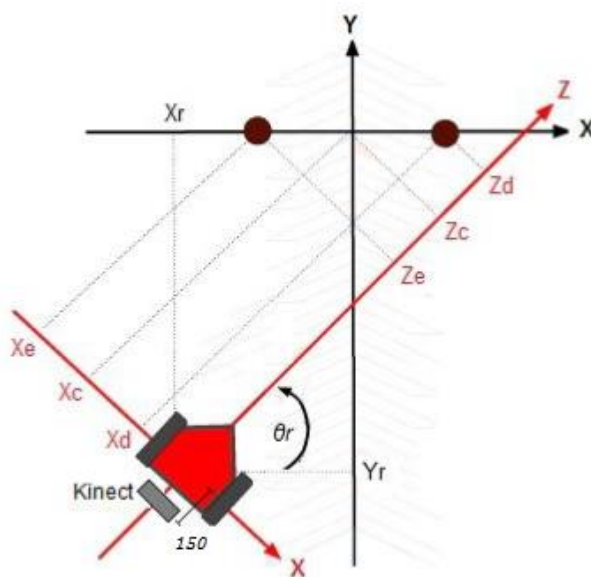
Figura 45 - Retas do marco encontradas



Fonte: autor (2021).

Caso o algoritmo encontre mais do que um retângulo é considerado que o marco não foi localizado. Caso o marco seja localizado com sucesso, é realizado o cálculo da posição da cadeira de rodas em relação ao marco encontrado. A localização da cadeira é determinada a partir da distância obtida no centro das retas verticais encontradas. A Figura 46 mostra o sistema de coordenadas da localização de marcos.

Figura 46 - Sistema de coordenadas de localização do marco



Fonte: Boschetti (2019).

As coordenadas do ponto médio do marco em relação a cadeira são determinadas através das equações 56 e 57. O ponto focal do Kinect ficou deslocado 150mm do centro do eixo da cadeira, sendo desta forma necessário a aplicação de uma correção.

$$X_c = \frac{X_d + X_e}{2} \quad (56)$$

$$Z_c = \frac{Z_d + Z_e}{2} - 150 \quad (57)$$

Para obter a posição da cadeira em relação ao centro do marco são utilizadas as equações 58, 59 e 60.

$$\theta_r = \frac{\pi}{2} - \arctan2(Z_d - Z_e, X_d - X_e) \quad (58)$$

$$X_r = -X_c \sin(\theta_r) - Z_c \cos(\theta_r) \quad (59)$$

$$Y_r = X_c \cos(\theta_r) - Z_c \sin(\theta_r) \quad (60)$$

A localização real de cada marco é armazenada nas informações do mapa do ambiente. Para determinar a posição da cadeira no ambiente durante a navegação, primeiramente a partir da localização da cadeira em relação ao marco obtida e da posição da cadeira no ambiente registrada pela odometria é estimada a posição do marco encontrado. Em seguida é buscado nas informações do mapa a posição real do marco mais próximo à posição estimada.

A partir da posição real do marco no ambiente e da localização da cadeira em relação ao marco é determinada a localização da cadeira no ambiente. As coordenadas da localização cadeira no ambiente são calculadas através das equações 61, 62 e 63, onde α é o ângulo formado entre os eixos do sistema de coordenadas da localização da cadeira no ambiente e do sistema de coordenadas da localização da cadeira em relação ao marco, a informação desse ângulo para cada marco está armazenado nas informações do mapa. Nas equações a abreviação m

significa a posição do marco no ambiente e cm significa a posição da cadeira em relação ao marco no sistema de coordenadas de localização de marcos.

$$X_c = X_m + X_{cm} \cos(-\alpha) + Y_{cm} \sin(-\alpha) \quad (61)$$

$$Y_c = Y_m - X_{cm} \sin(-\alpha) + Y_{cm} \cos(-\alpha) \quad (62)$$

$$\theta_c = \theta_{cm} + \alpha \quad (63)$$

4.8 SISTEMA DE LOCALIZAÇÃO DE PORTAS.

A metodologia para a detecção e localização de portas seguem os conceitos apresentados por Boschetti (2019). Com o Kinect posicionado na horizontal as bordas de uma porta podem ser vistas na imagem de profundidade através da variação de profundidade entre a parede e o vão da porta. Baseado neste princípio, o sistema de localização de portas busca encontrar essas discontinuidades na imagem de profundidade e determina quais são pertencentes a uma porta.

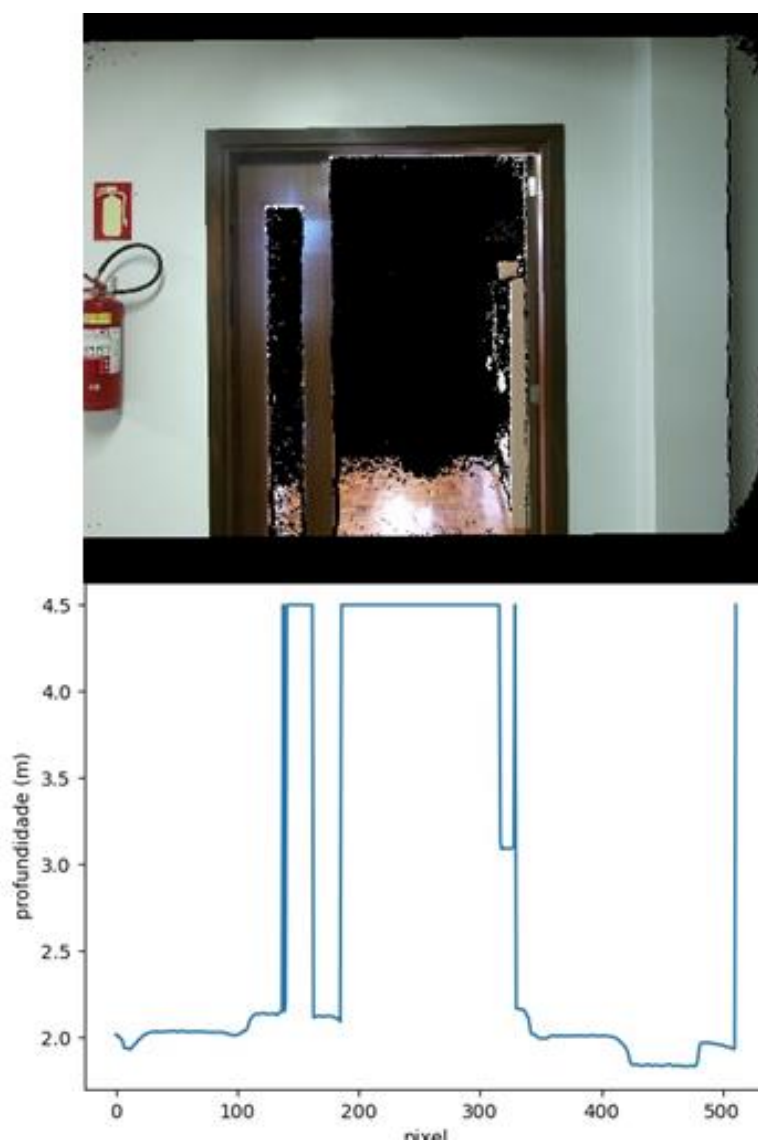
A localização das discontinuidades é feita através da profundidade extraída de pixels localizados em uma linha horizontal no centro da imagem. A Figura 47 mostra um exemplo de medidas de profundidade obtidas no centro da imagem de uma porta. Os pixels que apresentam informação de profundidade nula são considerados que estão a uma profundidade de 4,5 metros. Verificou-se que o sensor pode registrar medidas errôneas em pixels que estão mais distantes que o limite do sensor, com isso, é aplicado um algoritmo que anula medidas registradas entre pixels nulos.

O método para extrair as discontinuidades que podem pertencer a uma porta consiste em verificar a variação da profundidade de um pixel e seu subsequente à direita. Caso a variação seja maior que 0,5 metros a discontinuidade é candidata a borda de uma porta, se a taxa de variação for positiva a borda é candidata ao lado esquerdo, se a taxa de variação for negativa a borda é candidata ao lado direito. Para garantir que a variação não seja proveniente de nenhum ruído é verificado se a variação ainda existe em um range de 10 pixels, sendo 5 para direita e 5 para a esquerda.

A próxima etapa é avaliar os pontos de discontinuidades encontrados. A finalidade desta etapa é buscar os pares de bordas que satisfazem condições para

serem considerados como uma porta. Primeiramente é verificado a distância entre cada borda esquerda com todas as bordas que estão à sua direita, caso a distância seja entre 0,7 e 1,25 metros o par de bordas é candidato a ser uma porta. São descartados os pares de bordas que estão a uma distância de 20 pixels.

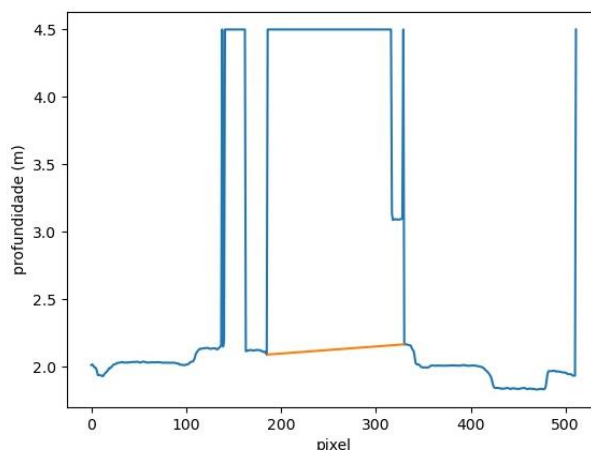
Figura 47 - Medidas no centro da imagem de profundidade de uma porta



Fonte: autor (2021).

Em seguida é traçado uma reta entre os pares de bordas encontrados, como pode ser vista na Figura 48, onde a reta traçada para cada par de borda está representada em alaranjado. Para um par ser considerado como pertencente a uma porta todos os pixels entre o par devem estar a uma distância mínima de 0,5 metros da reta traçada.

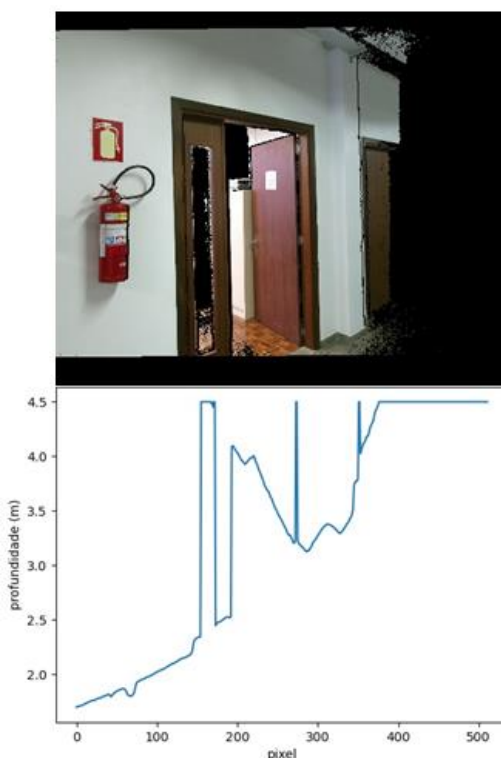
Figura 48 - Retas traçadas entre pares de bordas



Fonte: autor (2021).

Caso o algoritmo encontre mais do que um par de bordas é considerado que não foi possível localizar a porta. Caso nenhum par de bordas seja encontrado, é utilizado um algoritmo que busca encontrar uma porta apenas a partir de um ponto de descontinuidade, uma vez que existem situações que não podem ser verificadas dois pontos de descontinuidades, como no caso mostrado na Figura 49.

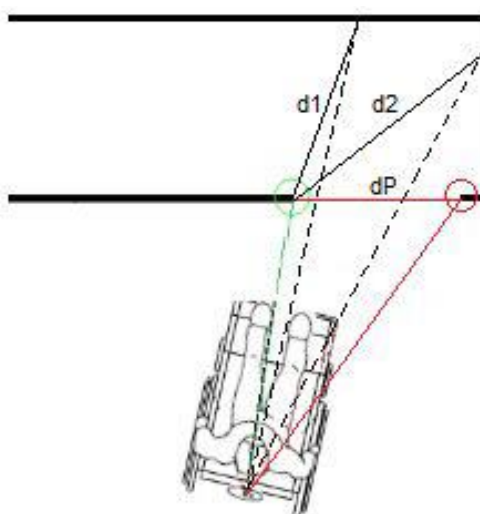
Figura 49 - Porta com um único ponto de descontinuidade



Fonte: autor (2021).

O algoritmo consiste em realizar a partir de um ponto de descontinuidade uma varredura buscando o pixel com a distância mais próxima da borda encontrada. A varredura iniciasse a uma distância de 20 pixels, caso o ponto de descontinuidade seja candidato a borda esquerda a varredura é realizada para a direita, caso o ponto de descontinuidade seja candidato a borda direita a varredura é realizada para a esquerda. A Figura 50 ilustra o processo de localização da porta por um único ponto de descontinuidade.

Figura 50 - Reconhecimento da porta com um ponto de descontinuidade



Fonte: Boschetti (2019).

O processo de localização por borda única é realizado para todos os pontos de descontinuidades encontrados. Caso o algoritmo encontre mais que uma possível porta é considerado que não foi possível localizar a porta. Como os ruídos presentes podem prejudicar o reconhecimento de portas, o sistema de localização de portas executa 10 tentativas até encontrar uma porta, caso contrário o sistema é encerrado. A Figura 51 mostra um exemplo de porta encontrada pelo algoritmo, onde os pontos em verde destacam as bordas da porta encontrada pelo algoritmo.

A partir da informação de profundidade em coordenadas cartesianas das bordas da porta encontrada é determinada a posição da cadeira em relação à porta. O sistema de coordenadas e as equações utilizadas são as mesmas utilizadas para determinar a posição da cadeira em relação aos marcos artificiais apresentadas na seção 4.6.

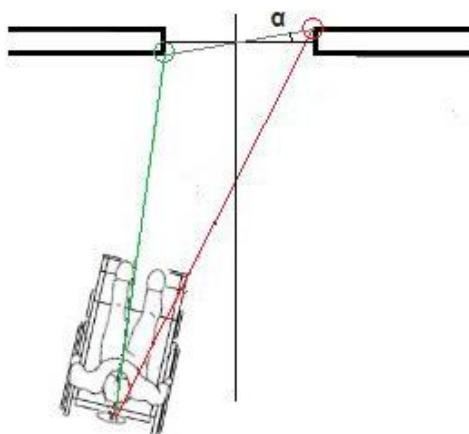
Figura 51 - Exemplo de porta localizada



Fonte: autor (2021).

Entretanto, observou-se que esse sistema de localização apresenta um erro na estimativa da orientação da cadeira em relação à porta. Esse erro também foi verificado por Boschetti (2019) em seu trabalho. Esse problema se deve ao fato de o algoritmo poder detectar como ponto de descontinuidade a parte interna ou externa do marco da porta. Quanto maior a largura do marco maior é o erro da estimativa da orientação, como pode ser observado pela Figura 52, onde α é erro da estimativa da orientação.

Figura 52 - Erro da estimativa da orientação em relação à porta



Fonte: adaptado de Boschetti (2019).

Na tentativa de amenizar esse erro, quanto a porta é localizada por dois pontos de descontinuidade é aplicado em cada ponto a rotina que localiza o outro extremo da porta pela distância mais curta. Os dois pontos que apresentam a menor distância entre eles são utilizados para calcular a posição da cadeira em relação à porta.

4.9 CONTROLE DA TRAJETÓRIA NA PASSAGEM POR PORTAS.

O controle da trajetória na passagem por portas é realizado a partir da informação da posição da cadeira em relação à porta obtida na etapa de localização da porta. Durante a trajetória a posição da cadeira é atualizada utilizando o sistema de odometria.

Os controladores de trajetória utilizados para realizar a passagem pelas portas são os mesmos empregados para executar a trajetória em corredores apresentados na seção 4.3. A diferença é que nesse caso a velocidade linear da cadeira é limitada em 250 mm/s. Além disso, como na passagem por portas a cadeira tem menos espaço para realizar manobras, a equação do ângulo de compensação também é alterada. O cálculo do ângulo de compensação utilizado nessa etapa é mostrado pela equação 64.

$$\theta_c = -0,09 x1 \quad (64)$$

O objetivo do controlador de trajetória é fazer com que a cadeira se desloque paralelamente à porta enquanto estiver a uma distância maior de 1000mm do eixo y, e se desloque realizando uma curva com raio de 1000mm quando estiver próxima ao eixo y até que a cadeira fique alinhada perpendicularmente com a porta.

A fim de verificar o funcionamento do controle de passagem por portas desenvolvido, foram realizadas simulações através do modelo cinemático discretizado de um robô diferencial caracterizado pelas equações 65, 66 e 67. Vale observar que o modelo utilizado na simulação não leva em consideração a dinâmica de aceleração da cadeira, desta forma a simulação não representa o comportamento da cadeira de forma fidedigna.

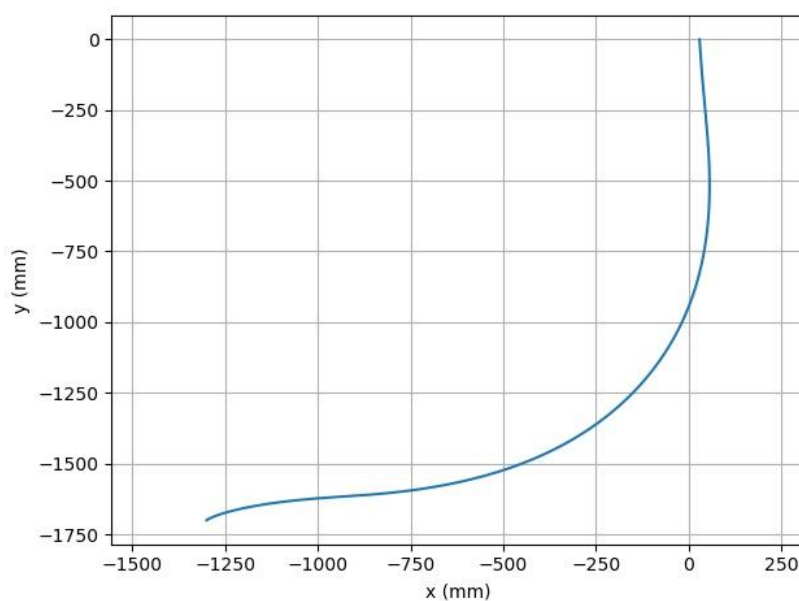
$$x_{k+1} = x_k + T \cdot v \cos(\theta_k) \quad (65)$$

$$y_{k+1} = y_k + T \cdot v \sin(\theta_k) \quad (66)$$

$$\theta_{k+1} = \theta_k + T \cdot \omega \quad (67)$$

O algoritmo de simulação foi escrito em linguagem python. A Figura 53 mostra um exemplo de trajetória simulada da cadeira de rodas em uma posição inicial (-1300,-1700), em mm, e um ângulo de orientação de 45 graus.

Figura 53 - Simulação de trajetória



Fonte: autor (2021).

5 RESULTADOS E DISCUSSÕES

Neste capítulo são relatados os ensaios realizados a fim de analisar o desempenho do sistema de navegação autônoma de cadeira de rodas implementado, além disso, são apresentados os resultados e suas respectivas análises. Os ensaios realizados consistem em verificar o controle de velocidade linear e angular com realimentação, a aplicação de filtros de ruídos em imagem de profundidade do Kinect, o sistema de detecção de marcos artificiais, o algoritmo de localização de portas, o controle de passagem por portas e o sistema de navegação completo.

5.1 TESTE DOS CONTROLADORES DE VELOCIDADE

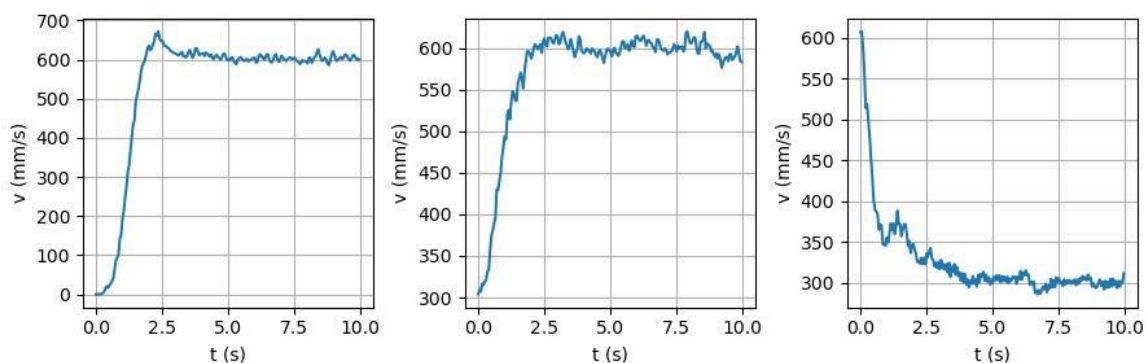
Como forma de verificar o desempenho dos controladores de velocidade com realimentação, foram realizados três ensaios para analisar o comportamento do controle de velocidade linear e angular da cadeira. O primeiro ensaio consiste em analisar a resposta do controlador partindo com a cadeira em repouso, o segundo em observar a aceleração da cadeira quando ela já estiver em movimento, e o terceiro em verificar o comportamento da desaceleração. A velocidade linear e angular da cadeira foram obtidas medindo a rotação das rodas através dos *encoders* acoplados no eixo dos motores.

No primeiro ensaio de controle de velocidade linear, foi analisado o comportamento da aceleração da cadeira ao atingir uma velocidade de 600 mm/s partindo do repouso, no segundo ao aumentar a velocidade de 300 mm/s para 600 mm/s, e no terceiro ao reduzir a velocidade de 600 mm/s para 300 mm/s.

Para a velocidade angular, no primeiro ensaio foi verificado o comportamento da aceleração angular da cadeira ao atingir uma velocidade angular de 1,2 rad/s partindo do repouso, no segundo ao aumentar a velocidade angular de 0,6 rad/s para 1,2 rad/s, e no terceiro ao reduzir a velocidade angular de 1,2 para 0,6 rad/s. Esse ensaio foi realizado tanto para movimentos para o lado esquerdo quanto para o direito.

A Figura 54 mostra os resultados obtidos no ensaio de controle de velocidade linear com realimentação.

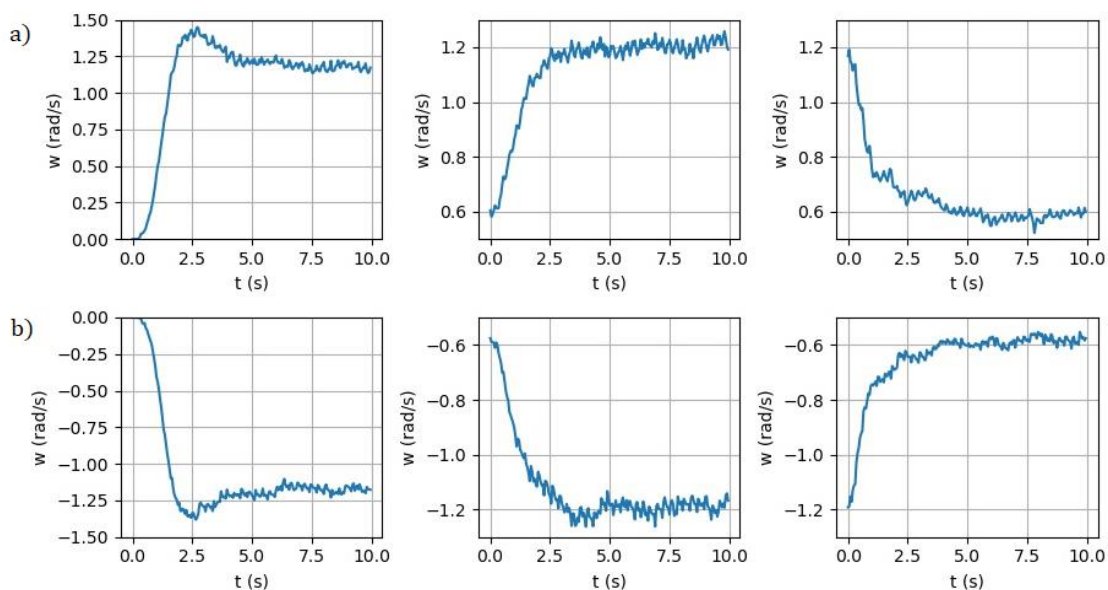
Figura 54 - Resultados do controle de velocidade linear



Fonte: autor (2021).

A Figura 55 mostra os resultados obtidos para o controle de velocidade angular com realimentação.

Figura 55 - Resultados do controle de velocidade angular



Nota: a) Resultados do controle de velocidade angular para a esquerda. b) Resultados do controle de velocidade angular para a direita.

Fonte: autor (2021).

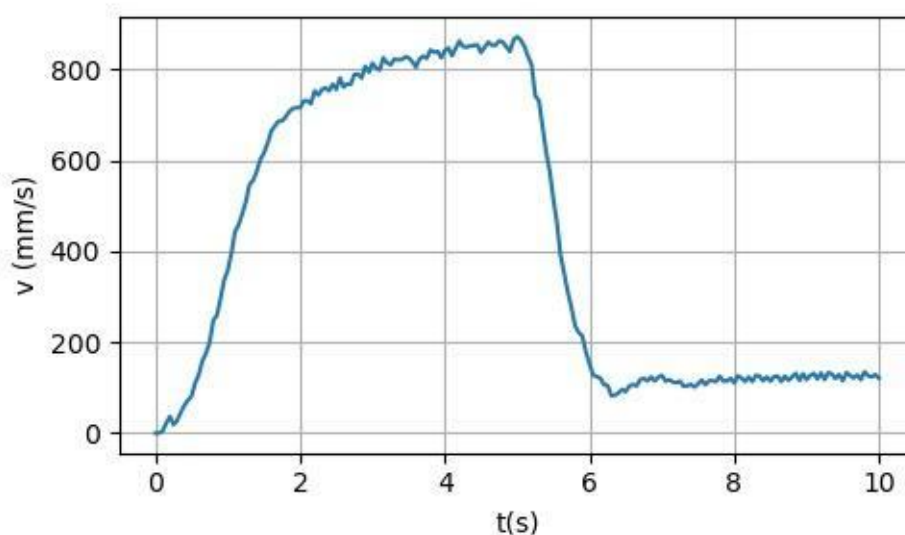
A partir dos resultados obtidos é possível observar que quando a cadeira acelera partindo do repouso ocorre um *overshoot* na velocidade. Isso acontece pois o sistema de controle nativo da cadeira possui um suavizador de partida o que causa um atraso na resposta da cadeira. Esse atraso faz com que a integral do erro aumente

rapidamente causando posteriormente o *overshoot*. Para retirar o *overshoot* seria necessário incluir esse atraso nas funções de transferência do sistema.

Entretanto, quando a cadeira aumenta a velocidade a partir de uma velocidade não nula, o *overshoot* não ocorre. Isso porque nessa situação o tempo de atraso na resposta da cadeira é muito pequeno, e como o tempo morto foi desconsiderado nas funções de transferência a resposta do controlador se aproxima ao projetado.

Já no caso em que a cadeira desacelera, observa-se que a resposta do controle é mais lenta. Esse fato indica que a dinâmica de aceleração da cadeira é diferente da dinâmica de desaceleração. Para testar essa hipótese foi realizado um ensaio onde é verificada a dinâmica de aceleração e a de desaceleração da cadeira em malha aberta. O ensaio consiste em aplicar um degrau de tensão no canal de velocidade linear e após a cadeira atingir a velocidade de regime permanente a amplitude da tensão é reduzida. É importante observar que para verificar a desaceleração não foi aplicado a tensão de velocidade igual a zero, pois nessa situação o sistema de freios da cadeira é acionado, travando as rodas. A Figura 56 exibe o resultado do experimento.

Figura 56 - Ensaio de desaceleração da cadeira



Fonte: autor (2021).

Como pode-se observar, a resposta de desaceleração da cadeira é mais rápida do que a de aceleração. Essa característica presente no controlador nativo da cadeira faz com que o controlador de velocidade com realimentação projetado apresente respostas diferentes na aceleração e na desaceleração.

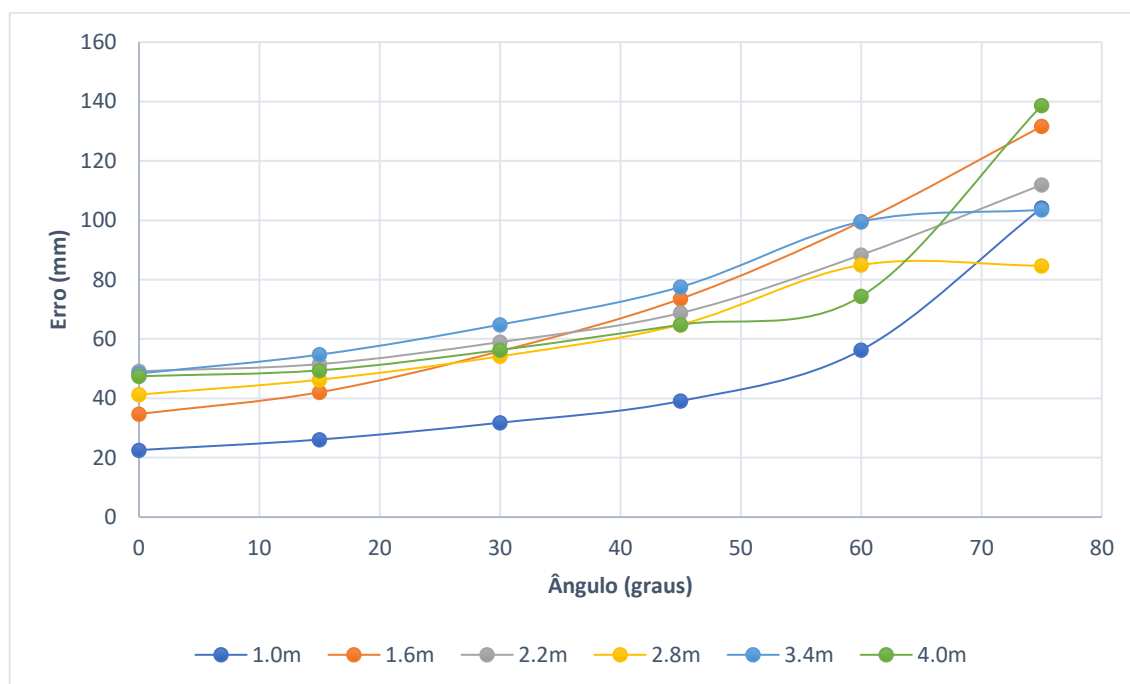
5.2 APLICAÇÃO DE FILTRO DE RUÍDOS EM IMAGENS DE PROFUNDIDADE

Neste trabalho também foi avaliado a aplicação de filtro de ruídos nas imagens de profundidade. Os ensaios realizados para analisar os ruídos presentes nas imagens de profundidade segue a metodologia utilizada por Fankhauser et al. (2015), onde o ruído axial é definido como a variação das medidas obtidas pela câmera Kinect no centro de uma superfície plana.

A metodologia consiste em obter medidas no centro do plano em diferentes distâncias e ângulos de orientação. Para cada posição do plano é calculado o desvio padrão das medidas e o erro médio a partir de 100 amostras. O plano foi posicionado em uma distância entre 1 e 4 metros em intervalos de 0,4 metros e ângulos entre 0 e 75 graus em intervalos de 15 graus.

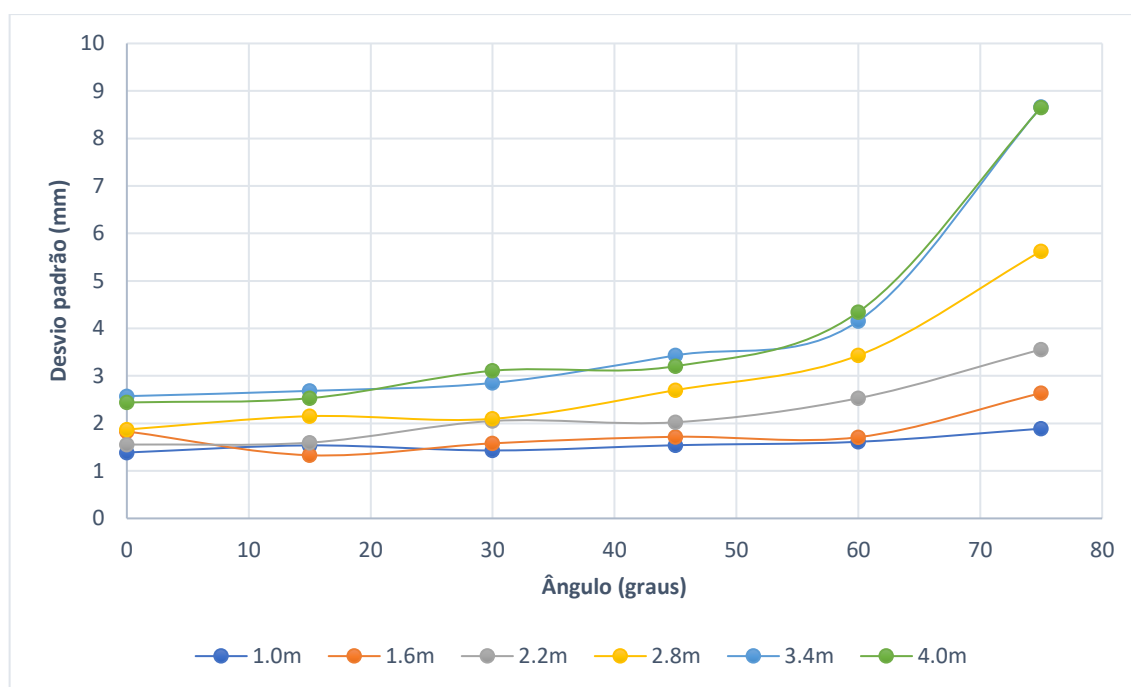
Primeiramente as medidas foram obtidas sem aplicar nenhum tipo filtro de ruído. A Figura 57 mostra o gráfico do erro e a Figura 58 mostra o gráfico do ruído, ambas as figuras mostram as medidas em função do ângulo para diferentes distâncias.

Figura 57 - Erro das medidas de profundidade sem filtro



Fonte: autor (2021).

Figura 58 - Ruído das medidas de profundidade sem filtro

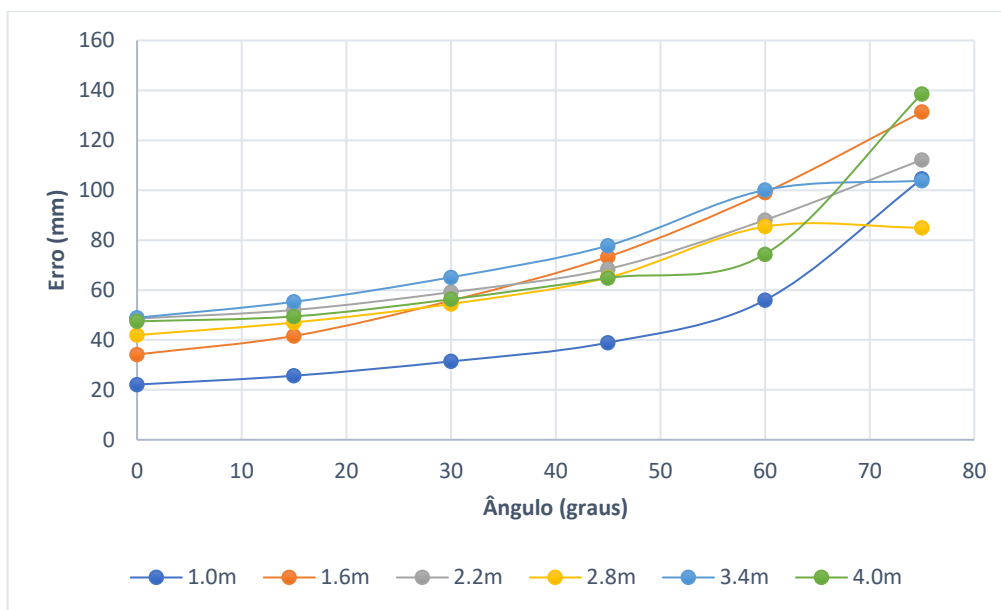


Fonte: autor (2021).

Por meio dos dados adquiridos observa-se que o erro das medidas de distância aumenta com a inclinação do plano e também é possível constatar que o ruído aumenta lentamente com a inclinação para ângulos menores que 45 graus e aumenta rapidamente para ângulos maiores que 45 graus. Com relação ao erro da medida e o ruído em função da distância não é possível verificar nenhum tipo de relação.

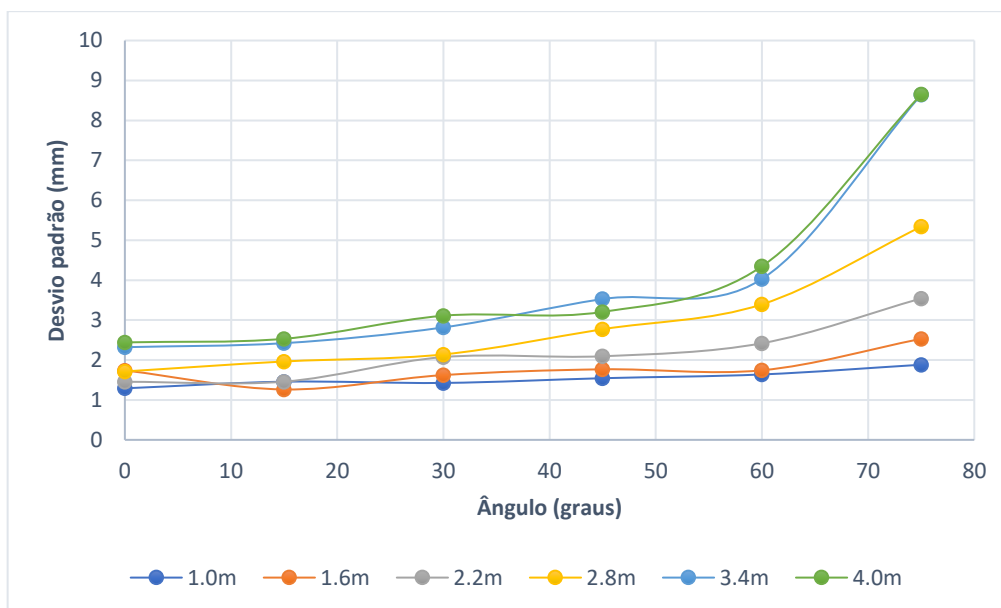
Em seguida é aplicado um filtro de ruído nas imagens de profundidade capturadas. O filtro utilizado é baseado no modelo de ruído do sensor Kinect v2 proposto por Fankhauser et al. (2015), e que está implementado na biblioteca *Kinect_Smoothing* escrita em linguagem *python* e disponível no site *github*. A Figura 59 mostra os resultados obtidos para o erro das medidas e a Figura 60 para o ruído após a aplicação do filtro.

Figura 59 - Erro das medidas de profundidade com filtro de Fankhauser



Fonte: autor (2021).

Figura 60 - Ruído das medidas de profundidade com filtro de Fankhauser

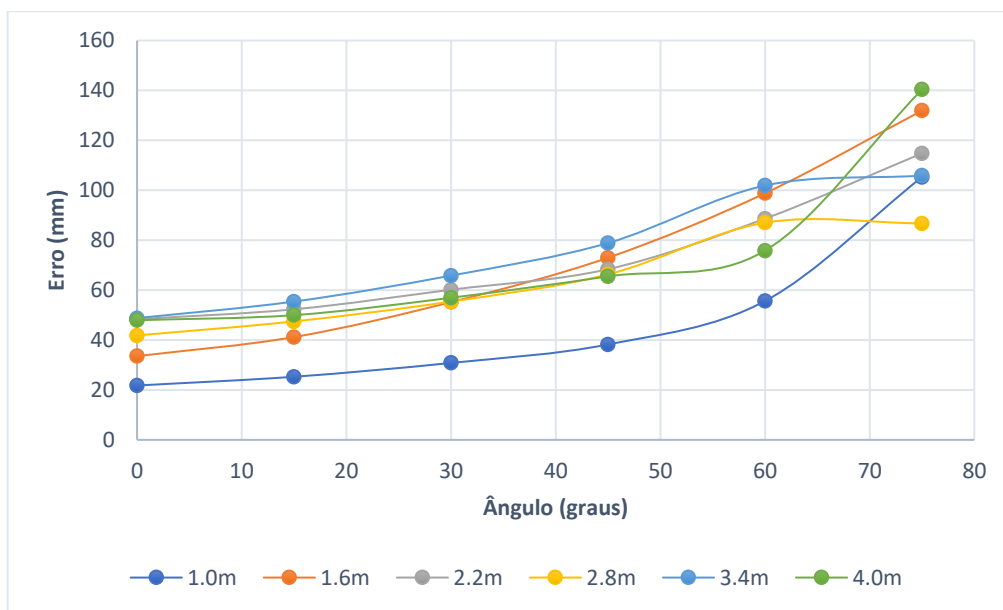


Fonte: autor (2021).

Através dos resultados obtidos verificou-se que a redução do ruído das medidas foi muito pequena, a média da diminuição do ruído foi de apenas 0,06mm. Com relação ao erro das medidas observou-se que na média houve um aumento do erro em 0,05mm. A partir dos resultados concluiu-se que a aplicação desse modelo de filtro de ruído não se justifica.

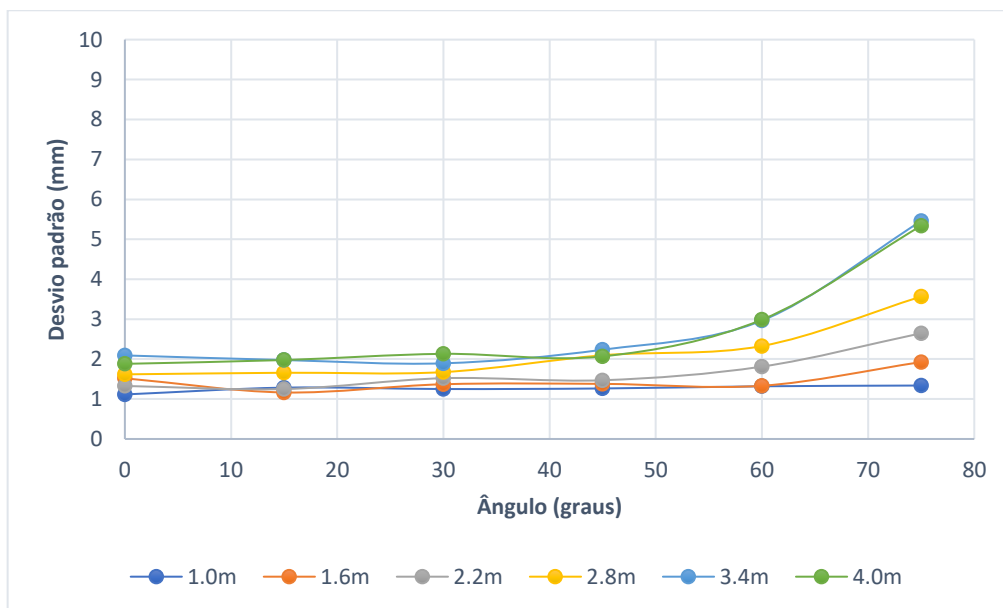
Também foi verificada a aplicação de um filtro gaussiano nas imagens de profundidade que também está implementado na biblioteca *Kinect_Smoothing*. A Figura 61 e a Figura 62 mostram os resultados obtidos com a utilização do filtro gaussiano.

Figura 61 - Erro das medidas de profundidade com filtro gaussiano



Fonte: autor (2021).

Figura 62 - Ruído das medidas de profundidade com filtro gaussiano



Fonte: autor (2021).

Com a aplicação do filtro gaussiano nas imagens de profundidade obteve-se na média um aumento do erro em 0,54mm. Entretanto houve uma redução média de 0,77mm no ruído. Além disso, observou-se que o ruído reduz principalmente nas medidas obtidas em superfícies mais inclinadas, essa é uma característica que pode ser interessante para a aplicação deste trabalho, uma vez que as medidas utilizadas para a localização de marcos e portas são adquiridas normalmente em superfícies muito inclinadas.

Também foi analisado o tempo de processamento necessário para aplicar os filtros. A Tabela 9 mostra o tempo médio para cada modelo de filtro, o tempo médio foi estimado através da aplicação dos filtros em 100 imagens de profundidade.

Tabela 9 - Tempo médio da aplicação dos filtros

FILTRO	TEMPO (S)
Fankhauser	6,3500
gaussiano	0,0014

Fonte: autor (2021).

O filtro gaussiano necessita de um tempo de processamento muito baixo, que não causaria nenhum impacto significativo no sistema de localização de marcos artificiais durante a navegação. Já o filtro que utiliza o modelo de Fankhauser et al. (2015) demanda de um tempo de processamento muito grande o que inviabiliza a utilização desse filtro para a aplicação deste trabalho.

5.3 TESTE DO SISTEMA DE LOCALIZAÇÃO DE MARCOS ARTIFICIAIS

Com a finalidade de verificar o desempenho do sistema de localização absoluta, a cadeira foi disposta em diversas posições em relação ao marco artificial. A Tabela 10 mostra as posições da cadeira em relação ao marco em que foi colocada. A posição x e y real da cadeira foi obtida utilizando uma trena, e a orientação da cadeira foi adquirida medindo a rotação da cadeira utilizando o acelerômetro e o giroscópio a partir de uma orientação conhecida. É importante destacar que as medidas da posição da cadeira não são precisas, elas são apenas estimativas da posição da cadeira para fins de comparação com as medidas do sistema de localização.

Tabela 10 - Posições da cadeira no ensaio de localização de marcos artificiais

N	X (mm)	Y (mm)	θ (graus)
1	-2416	-803	0
2	-2416	-803	-10
3	-2416	-1205	5
4	-1608	-803	10
5	-1608	-1608	25
6	-402	-1608	90
7	0	-1608	70
8	0	-2413	115
9	804	-2010	125
10	2815	-1205	180
11	2815	-1608	175
12	2815	-1608	185

Fonte: autor (2021).

Para cada posição em que a cadeira foi disposta foram capturadas 100 imagens RGB-D. Em seguida foram adquiridas as medidas da posição da cadeira aplicando o algoritmo de localização de marcos artificiais nas imagens capturadas. Para cada posição foi calculado a média das medidas e o desvio padrão em cada eixo, além disso, também foi calculado a diferença entre as medidas obtidas manualmente e as medidas obtidas pelo sistema de localização. Primeiramente obteve-se as medidas sem aplicar nenhum filtro de ruído nas imagens de profundidade, a Tabela 11 mostra os resultados encontrados.

O maior erro médio encontrado no eixo x e y foi de 79,3mm com um desvio padrão de 16,4 mm. Esse erro não causa nenhum impacto negativo no processo de navegação da cadeira de rodas. Já na orientação da cadeira o maior erro encontrado foi de 1,2 graus com um desvio padrão de 0,7 graus. Apesar do erro parecer pequeno, o erro na estimativa da orientação é um ponto crítico, pois um erro de aproximadamente de 2 graus resulta em um erro de aproximadamente 0,35 metros nos eixos x e y a cada 10 metros que a cadeira se desloca. Desta forma, para realizar a navegação com segurança os marcos não podem estar muito longe uns dos outros.

Tabela 11 - Resultados da localização de marcos sem filtro de ruído

N	X (mm)	Y (mm)	θ (graus)	σ_X (mm)	σ_Y (mm)	σ_θ (graus)	ΔX (mm)	ΔY (mm)	$\Delta\theta$ (graus)
1	-2423	-796	-0,4	7,7	16,5	0,4	-6,6	7,4	-0,4
2	-2398	-796	-11,2	13,1	30,9	0,7	17,7	6,8	-1,2
3	-2437	-1202	5,0	12,8	21,2	0,5	-21,2	2,8	0,0
4	-1639	-822	10,2	6,0	8,7	0,3	-30,7	-18,6	0,2
5	-1600	-1648	25,8	10,6	10,3	0,4	8,1	-39,8	0,8
6	-390	-1637	89,4	8,8	2,3	0,3	11,5	-29,1	-0,6
7	4	-1632	69,5	13,0	1,5	0,5	4,0	-24,0	-0,5
8	38	-2436	115,3	19,6	1,9	0,5	37,8	-23,2	0,3
9	820	-2051	124,9	12,0	4,8	0,3	16,1	-40,9	-0,1
10	2889	-1185	180,0	11,5	25,0	0,5	73,5	20,4	0,0
11	2894	-1606	174,9	16,4	28,4	0,6	79,3	2,2	-0,1
12	2860	-1700	183,9	61,4	108,4	2,1	45,2	-92,3	-1,1
MÉDIA				16,1	21,7	0,6	19,6	-19,0	-0,2

Fonte: autor (2021).

Também foi avaliado o sistema de localização de marcos aplicando filtro gaussiano nas imagens de profundidade, a Tabela 12 mostra os resultados obtidos.

Tabela 12 - Resultados da localização de marcos com filtro de filtro gaussiano

N	X (mm)	Y (mm)	θ (graus)	σ_X (mm)	σ_Y (mm)	σ_θ (graus)	ΔX (mm)	ΔY (mm)	$\Delta\theta$ (graus)
1	-2422	-795	-0,4	7,4	17,1	0,4	-6,0	7,6	-0,4
2	-2402	-787	-11,5	13,3	34,4	0,8	14,4	15,7	-1,5
3	-2438	-1201	5,0	11,9	19,8	0,5	-21,8	3,6	0,0
4	-1639	-821	10,2	5,1	7,4	0,3	-30,7	-18,1	0,2
5	-1600	-1647	25,7	7,5	7,4	0,3	8,2	-39,5	0,7
6	-383	-1639	89,6	5,9	1,5	0,2	19,3	-30,8	-0,4
7	3	-1632	69,4	7,1	1,1	0,2	2,9	-23,7	-0,6
8	39	-2436	115,4	12,7	1,5	0,3	38,9	-22,9	0,4
9	819	-2050	124,9	8,4	3,4	0,2	15,4	-40,3	-0,1
10	2890	-1183	180,0	11,0	25,7	0,5	74,7	21,8	0,0
11	2893	-1607	174,9	14,4	25,4	0,5	78,3	0,7	-0,1
12	2856	-1705	183,9	66,3	119,7	2,3	41,3	-97,0	-1,1
MÉDIA				14,2	22,0	0,5	19,6	-18,6	-0,2

Fonte: autor (2021).

Além disso, foi verificada a quantidade de falhas na localização do marco para cada posição da cadeira. A Tabela 13 compara o número de falhas com e sem a aplicação do filtro de ruído.

Tabela 13 - Falhas na localização do marco

N	SEM FILTRO (%)	COM FILTRO (%)
1	28	36
2	66	71
3	13	11
4	22	24
5	4	4
6	1	1
7	10	10
8	9	9
9	2	2
10	29	30
11	31	29
12	52	61
MÉDIA (%)	22,25	24

Fonte: autor (2021).

Comparando as medidas obtidas não é possível verificar nenhuma melhoria significativa que justifique a aplicação do filtro de profundidade no algoritmo de localização de marcos artificiais. Analisando o número de falhas de localização é possível observar em alguns casos uma maior quantidade de falhas ao aplicar o filtro gaussiano.

Também foi feita uma avaliação do tempo de execução do algoritmo de localização de marcos. A Tabela 14 mostra os resultados do tempo médio de execução a partir do processamento de 100 imagens RGB-D.

Tabela 14 - Tempo de execução do algoritmo de localização de marcos

	TEMPO (ms)
SEM FILTRO	7,29
COM FILTRO	7,67

Fonte: autor (2021).

O algoritmo de localização de marcos apresenta um tempo de execução rápido o suficiente para realizar várias tentativas de localização enquanto o marco estiver no campo de visão da câmera, garantindo maior confiabilidade do sistema de localização absoluta.

No geral os resultados indicam que o sistema de localização absoluta por marcos artificiais é suficiente para estimar a localização inicial da cadeira no ambiente, e que também é capaz de fornecer uma estimativa melhor do que a odometria durante a trajetória.

5.4 TESTE DO SISTEMA DE LOCALIZAÇÃO DE PORTAS

Com o intuito de avaliar a performance do sistema de localização de portas, a cadeira foi posicionada em diversas posições em relação a uma porta. As posições em que a cadeira foi colocada são apresentadas na Tabela 15, as medidas foram obtidas em relação ao centro dos marcos externos da porta. Para cada região em que a cadeira foi disposta foi obtida a posição média da cadeira aplicando o algoritmo de localização de portas em 100 imagens de profundidade. Também foi calculado o desvio padrão das medidas e a diferença entre a posição média encontrada e a posição em que a cadeira foi colocada.

Tabela 15 - Posições da cadeira no ensaio de localização de portas

N	X (mm)	Y (mm)	θ (graus)
1	-2455	-1510	30
2	-1852	-1812	45
3	-1550	-2112	65
4	-1248	-1207	20
5	-945	-1812	90
6	-345	-1510	80
7	-45	-2112	90
8	860	-1510	135
9	1157	-1207	160
10	1462	-1812	110
11	2064	-1207	170
12	2969	-1510	175

Fonte: autor (2021).

Com o objetivo de verificar se a aplicação da rotina de localização do ponto mais próximo de uma descontinuidade após localizar uma porta por dois pontos de descontinuidades reduz o erro da estimativa da orientação da cadeira, primeiramente foram obtidas as medidas sem essa etapa. Os resultados encontrados são mostrados na Tabela 16.

Tabela 16 - Resultados sem etapa de localização por ponto mais próximo

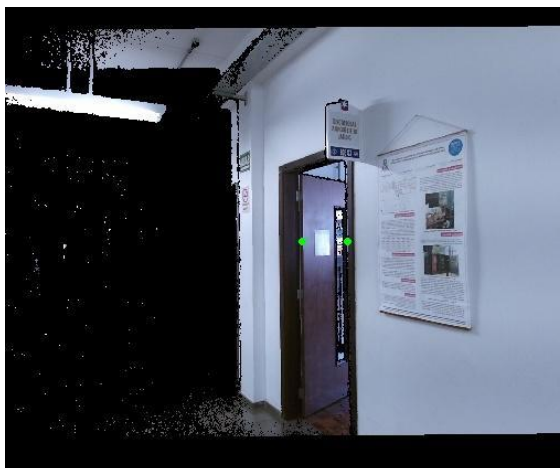
N	X (mm)	Y (mm)	θ (graus)	σX (mm)	σY (mm)	$\sigma \theta$ (graus)	ΔX (mm)	ΔY (mm)	$\Delta \theta$ (graus)
1	-2638	-1130	20,3	79,6	158,0	4,2	-182,9	380,1	-9,7
2	-2156	-1485	35,3	113,4	115,5	4,1	-303,5	326,8	-9,7
3	-1813	-1928	57,9	131,2	92,2	3,9	-263,2	184,4	-7,1
4	-1517	-939	6,4	4,9	15,0	0,5	-268,8	267,6	-13,6
5	-1122	-1719	84,4	472,6	148,6	13,8	-176,7	93,3	-5,6
6	-667	-1522	67,4	71,5	28,3	2,4	-321,6	-12,2	-12,6
7	-111	-2212	87,2	197,9	34,8	4,9	-65,9	-100,3	-2,8
8	1118	-1749	145,0	43,2	21,9	1,2	257,7	-239,2	10,0
9	1393	-1035	169,8	24,1	30,9	1,1	236,4	172,1	9,8
10	1827	-1872	121,3	45,1	55,8	1,5	365,1	-59,9	11,3
11	2252	-889	179,2	6,2	27,4	0,6	188,3	317,8	9,2
12	3175	-1015	184,7	15,8	30,9	0,5	206,0	495,2	9,7
MÉDIA				100,5	63,3	3,2	-27,4	152,1	-0,9

Fonte: autor (2021).

Através dos resultados extraídos verificou-se um erro na estimativa da orientação bastante elevado, sendo que o maior erro médio encontrado foi de 13,6 graus. A porta utilizada para realizar o ensaio possui um comprimento de 890mm e a largura do marco da porta possui uma largura de 65mm, com isso o erro esperado causado pela localização da porta por meio de um ponto descontinuidade interno e um externo do marco da porta era de aproximadamente 4,2 graus.

Além disso, nas situações em que a porta só pode ser encontrada pelo algoritmo que localiza a porta através de um ponto de descontinuidade era esperado um erro na estimativa da orientação muito menor. Verificando as bordas encontradas nessas situações por meio das imagens RGB como a mostrada na Figura 63, constatou-se que o erro ocorre pois a lateral da porta, que está próximo ao marco interno da porta, é o ponto mais próximo do ponto de descontinuidade encontrado, que está localizado no marco externo da porta.

Figura 63 - Porta localizada por um ponto de descontinuidade



Fonte: autor (2021).

Os erros obtidos nos eixos x e y também são bastantes elevados. Entretanto os erros nos eixos x e y são reflexos dos erros na estimativa da orientação da cadeira em relação à porta, que faz com que o eixo de coordenadas de localização calculado não fique alinhado perpendicularmente com a porta.

Em seguida foram obtidas as medidas da posição da cadeira utilizando o algoritmo que busca o ponto mais próximo de cada borda da porta encontrada através de dois pontos de descontinuidade. A Tabela 17 mostra os resultados obtidos.

Tabela 17 - Resultados com etapa de localização por ponto mais próximo

N	X (mm)	Y (mm)	θ (graus)	σ_X (mm)	σ_Y (mm)	σ_θ (graus)	ΔX (mm)	ΔY (mm)	$\Delta\theta$ (graus)
1	-2638	-1130	20,3	79,6	158,0	4,2	-182,9	380,1	-9,7
2	-2156	-1485	35,3	113,4	115,5	4,1	-303,5	326,8	-9,7
3	-1813	-1928	57,9	131,2	92,2	3,9	-263,2	184,4	-7,1
4	-1517	-939	6,4	4,9	15,0	0,5	-268,8	267,6	-13,6
5	-1122	-1719	84,4	472,6	148,6	13,8	-176,7	93,3	-5,6
6	-657	-1526	67,7	26,1	19,4	1,0	-311,8	-15,9	-12,3
7	-128	-2208	86,7	188,8	36,2	4,7	-83,2	-96,4	-3,3
8	1009	-1785	141,6	67,4	26,4	2,2	149,3	-274,8	6,6
9	1243	-1146	162,6	42,4	32,1	2,1	85,6	61,5	2,6
10	1629	-2000	115,7	98,3	59,9	2,7	166,8	-188,0	5,7
11	2163	-1043	174,3	53,7	83,5	2,8	98,5	163,7	4,3
12	3088	-1205	180,7	63,6	137,0	3,0	118,6	304,9	5,7
MÉDIA				111,8	77,0	3,8	-80,9	100,6	-3,0

Fonte: autor (2021).

A partir dos resultados obtidos constatou-se que nas situações onde a porta é encontrada através de dois pontos de descontinuidades, houve uma melhoria expressiva na estimativa da orientação e conseqüentemente da posição x e y da cadeira em relação à porta acrescentando a etapa de localização do ponto mais próximo dos pontos de descontinuidade encontrados.

Na seqüência, verificou-se o desempenho do sistema de localização de portas aplicando filtro gaussiano nas imagens de profundidade. A Tabela 18 exibe as medidas adquiridas após aplicar filtro nas imagens de profundidade.

Tabela 18 - Resultado da localização de portas com filtro de ruído

N	X (mm)	Y (mm)	θ (graus)	σ_X (mm)	σ_Y (mm)	σ_θ (graus)	ΔX (mm)	ΔY (mm)	$\Delta\theta$ (graus)
1	-2627	-1177	21,0	85,0	160,2	4,3	-171,8	332,8	-9,0
2	-1953	-1497	39,6	1096,6	93,3	24,7	-100,7	315,2	-5,4
3	-1827	-1937	57,4	152,3	107,0	4,5	-277,2	174,6	-7,6
4	-1512	-956	6,8	39,3	38,7	1,9	-264,4	251,4	-13,2
5	-1196	-1730	82,1	123,1	52,0	4,1	-251,2	81,8	-7,9
6	-655	-1515	67,6	4,2	3,4	0,2	-309,6	-4,6	-12,4
7	-424	-2184	79,4	11,7	4,9	0,3	-379,0	-71,6	-10,6
8	988	-1787	141,0	46,9	15,1	1,5	127,7	-277,0	6,0
9	1224	-1161	161,7	18,7	14,1	0,9	67,0	46,5	1,7
10	1560	-2051	113,9	54,4	30,0	1,5	97,8	-238,5	3,9
11	2122	-1111	172,3	29,1	43,1	1,5	58,5	95,8	2,3
12	3041	-1317	178,4	40,5	78,4	1,8	71,9	192,5	3,4
MÉDIA				141,8	53,3	3,9	-110,9	74,9	-4,1

Fonte: autor (2021).

Além disso, foi verificada a quantidade de falhas na localização da porta para cada posição da cadeira. A Tabela 13 compara o número de falhas com e sem a aplicação do filtro de ruído.

Tabela 19 - Falhas na localização de portas

N	SEM FILTRO (%)	COM FILTRO (%)
1	0	0
2	2	11
3	38	15
4	0	9
5	22	49
6	3	0
7	1	0
8	0	0
9	0	0
10	0	0
11	0	0
12	0	0
MÉDIA (%)	5,5	7,0

Fonte: autor (2021).

Analisando os resultados das medidas da posição e a comparação de falhas com e sem a aplicação de filtros, observou-se que em algumas situações a utilização do filtro gaussiano nas imagens de profundidade resultou em melhorias na localização da porta, entretanto em outras o filtro prejudicou o desempenho de localização. Desta forma não foi possível observar melhorias no desempenho do sistema de localização de portas que justifiquem a utilização de filtros de ruídos em imagens de profundidade.

5.5 TESTE DE PASSAGEM POR PORTAS

O teste de passagens por portas foi realizado a partir das posições em que a cadeira foi posicionada no teste de localização de portas. No ensaio de passagem por portas foi obtida a posição final e inicial da cadeira em relação ao centro porta medida manualmente e a posição registrada pelo sistema de passagem por portas. A posição final foi adquirida utilizando uma trena e a orientação da cadeira foi estimada por meio da variação angular registrada através do acelerômetro e do giroscópio.

Para avaliar a trajetória foram selecionadas 7 situações de posição da cadeira em relação porta, sendo elas: quando a cadeira está ao lado direito da porta e próximo à parede; quando está ao lado esquerdo da porta e próximo à parede; quando está perto da porta ao lado direito; quando está perto da porta ao lado esquerdo; quando

está longe da porta ao lado direito; quando está longe da porta ao lado esquerdo; e quando está muito próximo ao centro da porta. Os demais resultados são mostrados no Apêndice A.

Conforme verificado no ensaio de localização de portas, o erro da estimativa da cadeira em relação à porta é maior quando a cadeira está posicionada ao lado esquerdo do que quando está no lado direito. Por esse motivo foram selecionadas amostras em ambos os lados da porta a fim de verificar o impacto na trajetória causado pelo erro na orientação da cadeira.

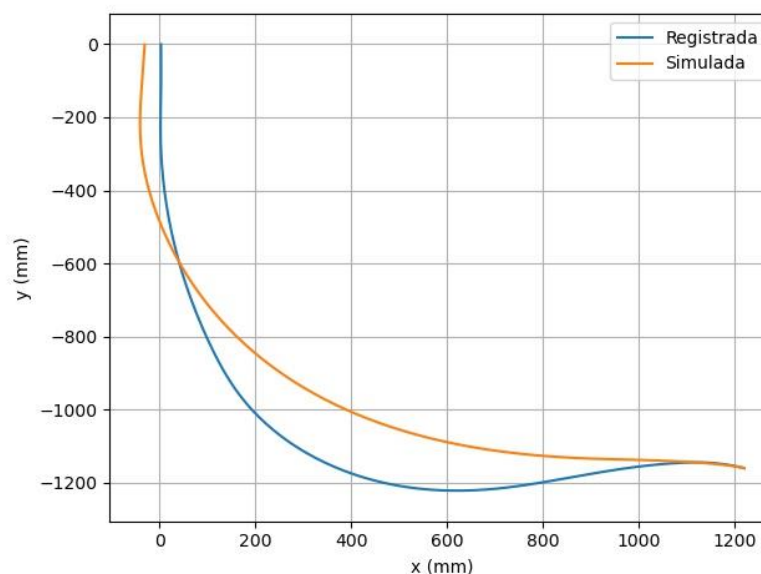
A situação em que a cadeira está muito próxima da parede ao lado direito pode ser analisada na amostra em que a cadeira está na posição 9. A Tabela 20 mostra as medidas da posição inicial e final da cadeira nesse caso e a Figura 64 mostra a trajetória registrada pelo sistema (azul) e a trajetória simulada através do modelo teórico (alaranjado).

Tabela 20 - Medidas trajetória posição 9

	X (mm)	Y (mm)	θ (graus)
Posição Inicial	1157	-1207	160
Posição Inicial Registrada	1220,6	-1159,5	161,7
Posição Final	22	-27	88,7
Posição Final Registrada	2,6	-0,6	90,4

Fonte: autor (2021).

Figura 64 - Trajetória posição 9



Fonte: autor (2021).

No início da trajetória a cadeira executou um movimento para a esquerda se distanciando da trajetória esperada, entretanto o controlador conseguiu corrigir a trajetória fazendo a cadeira atingir a posição (22, -27), em mm, e na orientação de 88,7°.

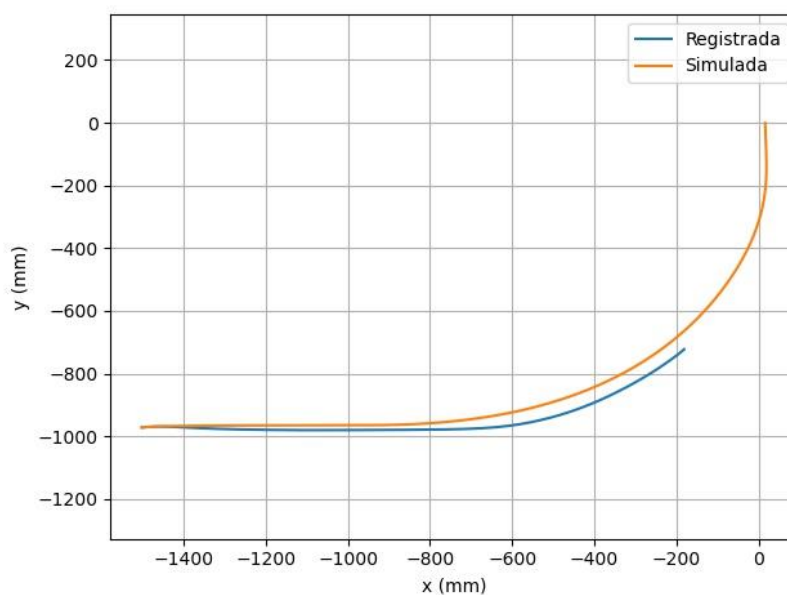
A Tabela 21 mostra as medidas da cadeira adquiridas na posição 4, na qual a cadeira está localizada próxima da parede ao lado esquerdo da porta. A figura 65 exhibe a trajetória executada pela cadeira. Nessa situação a trajetória resultou em uma colisão.

Tabela 21 - Medidas trajetória posição 4

	X (mm)	Y (mm)	θ (graus)
Posição Inicial	-1248	-1207	20
Posição Inicial Registrada	-1501,4	-971,9	7,5

Fonte: autor (2021).

Figura 65 - Trajetória posição 4



Fonte: autor (2021).

Nessa situação no início da trajetória a cadeira realizou um deslocamento angular para a direita muito pequeno se comparado à situação anterior, isso porque a orientação inicial da cadeira medida pelo sistema de localização de porta foi de 7,5 graus e para uma distância em x maior de 1000mm o controlador ajusta a orientação para 0 graus, com isso a correção da angulação foi realizada de forma mais suave.

Entretanto, nessa situação houve a colisão da cadeira. A colisão ocorreu devido ao erro da estimativa da orientação da cadeira juntamente com o fato de a cadeira estar muito próxima da parede. O erro na medida da orientação resultou em uma medida de distância no eixo x maior do que a distância real, o que retardou o movimento angular para a esquerda, desta forma a cadeira passou da linha central da porta e ao realizar a curva colidiu com a lateral da porta.

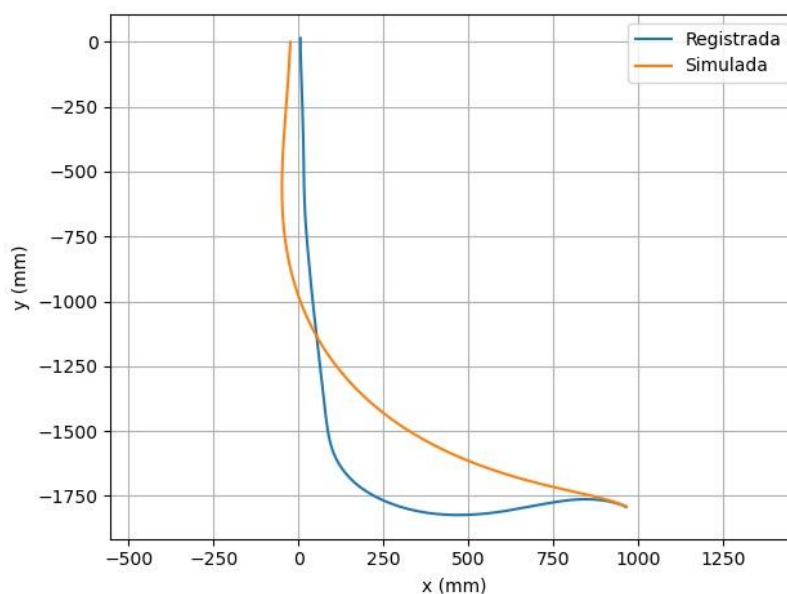
A situação em que a cadeira está perto da porta ao lado direito pode ser observada por meio da posição 8. A Tabela 22 exibe as medidas obtidas e a Figura 66 mostra a trajetória executada pela cadeira registrada pelo sistema.

Tabela 22 - Medidas trajetória posição 8

	X (mm)	Y (mm)	θ (graus)
Posição Inicial	860	-1510	135
Posição Inicial Registrada	964,8	-1792,3	140,4
Posição Final	36	65	84,9
Posição Final Registrada	5,9	14,1	90,2

Fonte: autor (2021).

Figura 66 - Trajetória posição 8



Fonte: autor (2021).

Nessa situação a cadeira realizou no início da trajetória um movimento angular brusco para a esquerda ultrapassando a orientação ideal, o que causou uma grande diferença entre a trajetória registrada e a simulada. Entretanto a posição final

registrada se aproximou mais da origem do que a verificada na simulação. Com relação a posição final da cadeira em relação a porta ela parou na coordenada (36, 65), em mm, e na orientação de $84,6^\circ$.

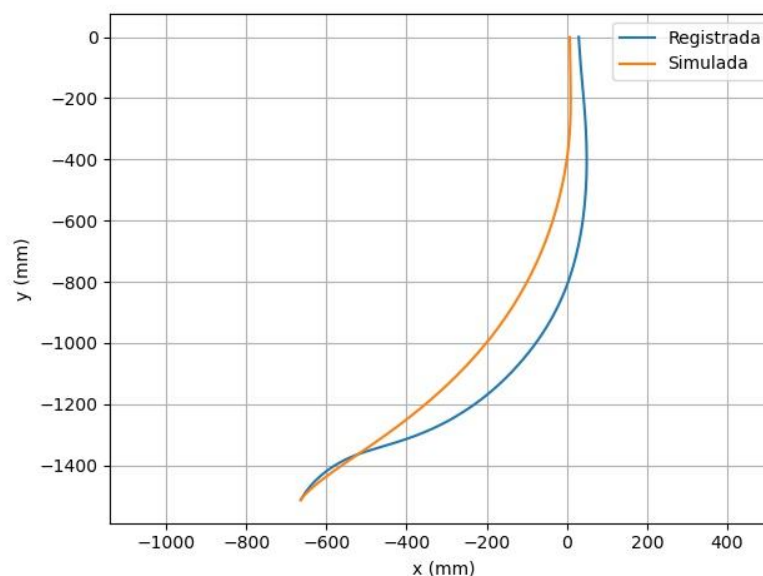
A Tabela 23 mostra as medidas obtidas na posição 6, nessa situação pode ser verificado o comportamento da trajetória da cadeira quando ela está perto da porta ao lado esquerdo. A trajetória registrada nessa situação pode ser observada através da Figura 67.

Tabela 23 - Medidas trajetória posição 6

	X (mm)	Y (mm)	θ (graus)
Posição Inicial	-345	-1510	80
Posição Inicial Registrada	-663,9	-1512,5	66,8
Posição Final	40	105	105,9
Posição Final Registrada	29,5	0,1	92,7

Fonte: autor (2021).

Figura 67 - Trajetória posição 6



Fonte: autor (2021).

Nesse caso a cadeira iniciou a trajetória com um movimento angular sutil para a direita, que fez com que a cadeira se deslocasse à direita da trajetória teórica. Nesse caso o controlador realizou o ajuste da orientação de forma lenta, com isso a posição final registrada ficou distante da origem em aproximadamente 29,5mm.

Observou-se que nessa situação, devido ao erro da estimativa da orientação, a cadeira acabou realizando a passagem de forma oblíqua parando na posição (40, 105), em mm, com um ângulo de 105,9° em relação à porta. Nessa situação o erro da orientação não causou colisão pois a cadeira estava distante o suficiente da parede para realizar a manobra de ajuste da orientação em relação à porta.

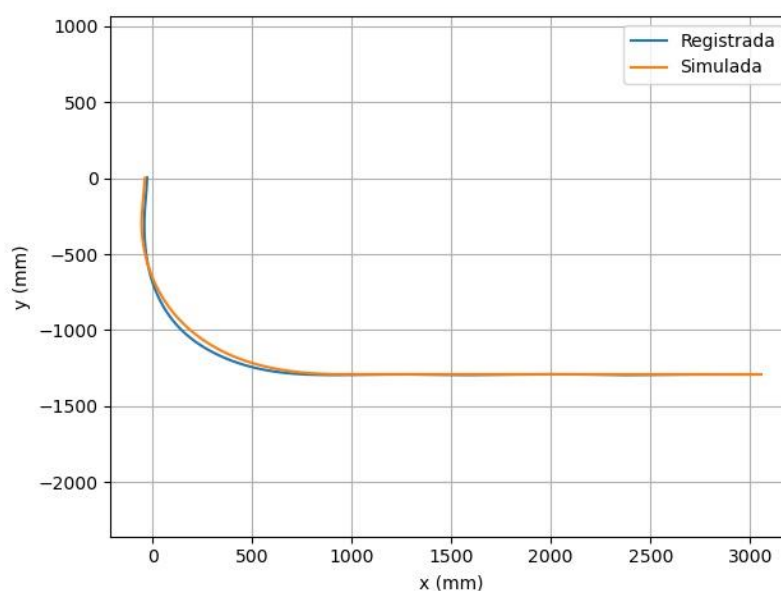
A situação em que a cadeira está localizada longe da porta ao lado direito pode ser analisada através da posição 12. As medidas adquiridas nessa situação podem ser vistas na Tabela 24 e a trajetória registrada pode ser observada por meio da Figura 68.

Tabela 24 - Medidas trajetória posição 12

	X (mm)	Y (mm)	θ (graus)
Posição Inicial	2969	-1510	175
Posição Inicial Registrada	3054,0	-1290,0	178,9
Posição Final	13	-15	84,3
Posição Final Registrada	-26,4	3,1	88,2

Fonte: autor (2021).

Figura 68 - Trajetória posição 12



Fonte: autor (2021).

Nessa situação observou-se que a cadeira não realizou nenhum ajuste na orientação no início da trajetória, desta forma a trajetória realizada se aproximou da

teórica. Ao final do deslocamento a cadeira parou na posição (13, -15), em mm, com uma orientação de $84,3^\circ$.

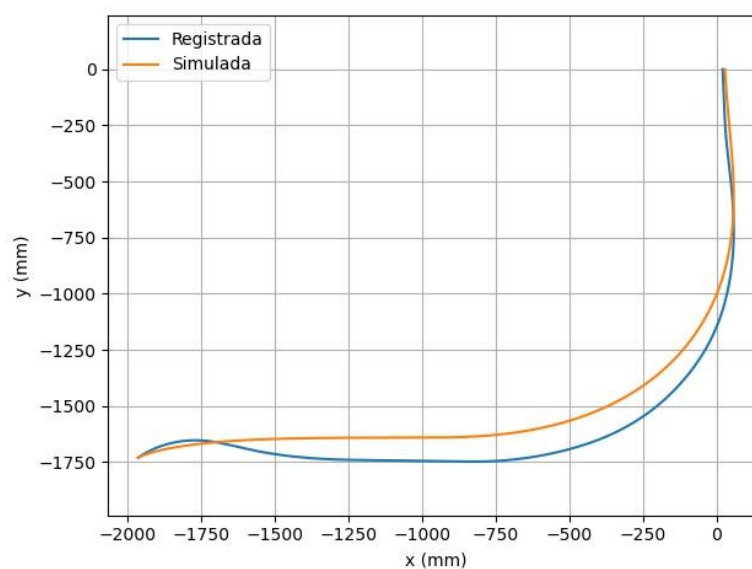
A Tabela 25 mostra as medidas obtidas na posição 2, onde pode-se observar a trajetória quando ela está longe da porta ao lado esquerdo. A Figura 69 exhibe a trajetória registrada nessa situação.

Tabela 25 - Medidas trajetória posição 12

	X (mm)	Y (mm)	θ (graus)
Posição Inicial	-1852	-1812	45
Posição Inicial Registrada	-1965,0	-1731,0	42,7
Posição Final	67	22	94,8
Posição Final Registrada	21,1	-0,1	92,5

Fonte: autor (2021).

Figura 69 - Trajetória posição 2



Fonte: autor (2021).

Nesse caso a cadeira realizou um movimento angular para a direita no início da trajetória para ajustar a orientação para 0 graus, o que fez com que a trajetória registrada se distanciasse da esperada. Entretanto verificou-se que ao longo da trajetória a cadeira realizou uma curvatura semelhante a simulada, com isso, a posição final registrada se aproximou da posição final obtida por meio do modelo teórico.

Nessa situação, observou-se que o erro da estimativa da orientação da cadeira foi de 2,3 graus, o que é um erro pequeno se comparado às outras situações em que a cadeira estava posicionada ao lado esquerdo da porta. Com isso, a cadeira realizou a passagem pela porta de forma quase que perpendicular, parando na posição (67, 22), em mm, com uma orientação de 92,5°.

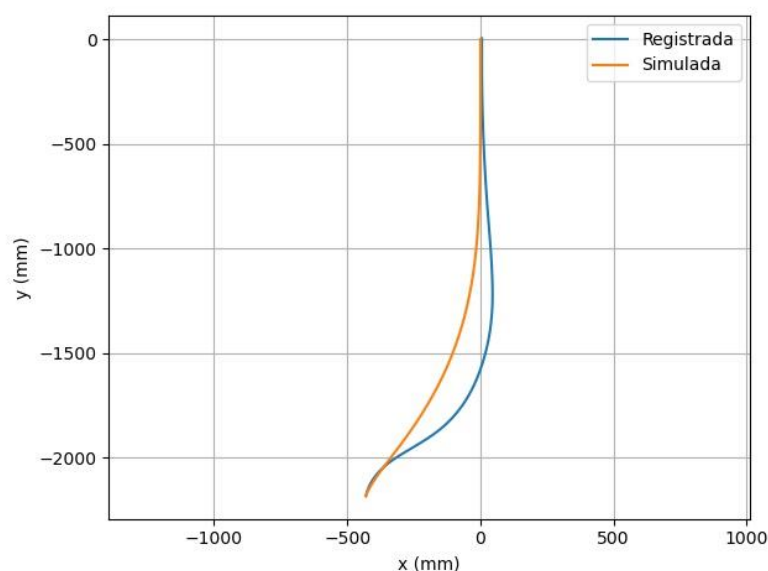
O último caso analisado é quando a cadeira se encontra próxima ao centro da porta, essa situação pode ser observada na posição 7. Na Tabela 26 são mostradas as medidas obtidas e a trajetória registrada pelo sistema é mostrada na Figura 70.

Tabela 26 - Medidas trajetória posição 7

	X (mm)	Y (mm)	θ (graus)
Posição Inicial	-45	-2112	90
Posição Inicial Registrada	-430,0	-2183,5	79,3
Posição Final	-20	112	101,1
Posição Final Registrada	5,7	3,5	90,5

Fonte: autor (2021).

Figura 70 - Trajetória posição 7



Fonte: autor (2021).

Nessa situação ocorreu um grande erro na estimativa da orientação o que causou um grande erro na estimativa da posição no eixo x. Com isso, ao invés da cadeira seguir uma trajetória retilínea acabou realizando um movimento angular para

a direita. Ao final da trajetória a cadeira parou na posição (-20, 112), em mm, com orientação de $101,1^\circ$ em relação à porta.

Comparando a trajetória registrada e a simulada observa-se que a cadeira se deslocou para a direita em relação a trajetória teórica. Como a cadeira estava muito longe da parede, o controlador conseguiu corrigir a trajetória atingindo uma posição final muito próxima ao esperado. Entretanto se a cadeira estivesse posicionada mais próximo da parede o erro da posição final seria maior.

A partir dos resultados constatou-se que no início da trajetória a cadeira realiza movimentos oscilatórios na tentativa de ajustar a orientação, e que quanto maior for a diferença da orientação inicial da cadeira e o ângulo ideal para a distância que ela está, maior é a oscilação. Essas oscilações ocorrem porque a resposta do controle de velocidade angular na desaceleração é mais lenta do na aceleração, isso faz com que a cadeira ultrapasse a orientação desejada.

Além disso, verificou-se que apesar de existirem situações que ocorrem grandes erros na estimativa da orientação da cadeira em relação à porta, a cadeira consegue realizar a passagem. Entretanto a passagem não é executada de forma perpendicular à porta, e para maior segurança é necessário que a cadeira esteja localizada a aproximadamente 1,5 metros de distância da parede.

5.6 TESTE DE NAVEGAÇÃO

Com o objetivo de avaliar o sistema completo, desde a determinação da localização absoluta inicial da cadeira no ambiente até a passagem pela porta do destino selecionado, com a cadeira posicionada na entrada do ambiente de navegação foi definida a sala 308 como destino. O ensaio do sistema de navegação completo foi realizado 5 vezes.

Nos testes iniciais verificou-se que quando o sistema localizava o marco artificial durante a realização de um movimento angular, a cadeira saía completamente da trajetória. Já ao localizar um marco na trajetória retilínea, também observou-se um erro na correção da trajetória, mas muito menor, entretanto ao localizar um marco com a cadeira com um ângulo maior 180 graus e menor que 0 graus em relação ao marco o erro geralmente era maior.

Uma hipótese que explicaria o problema observado é que a imagem RGB e a imagem de profundidade capturadas pelo Kinect são desalinhadas devido à falta de

sincronia entre elas. Com isso foi realizado um ensaio com o objetivo de confirmar a hipótese. A cadeira foi posicionada em frente a uma porta e foram capturadas uma sequência de imagens RGB-D enquanto a cadeira executava um movimento angular com velocidade de 0,4 rad/s. A Figura 71 mostra uma sequência de 3 imagens RGB-D captadas durante o movimento angular da cadeira.

Figura 71 - Sequência de imagens RGB-D durante movimento angular



Fonte: autor (2021).

Observando o vão da porta na imagem RGB e na de profundidade, verifica-se que na primeira captura a imagem de profundidade está atrasada em relação à RGB, na segunda as imagens estão aproximadamente alinhadas e na terceira a imagem de profundidade está adiantada em relação à RGB. Com isso constata-se que a biblioteca utilizada para a aquisição das imagens não fornece nenhuma garantia de sincronização entre as imagens RGB e de profundidade, e esse problema causa os erros na determinação da posição absoluta da cadeira durante o deslocamento.

Com a finalidade de amenizar os problemas causados pela falta de sincronia, algumas medidas foram tomadas. Os marcos artificiais foram colocados afastados das regiões onde a cadeira realiza curvas de 90 graus para evitar que ele seja detectado enquanto a cadeira executa movimentos angulares. Para maior segurança na navegação entre corredores, os marcos foram posicionados de forma intercalada nas paredes do corredor e próximo uns dos outros. Quando a orientação da cadeira em relação ao marco detectado pelo sistema de localização absoluta é menor que 0 graus ou maior que 180 graus a posição da cadeira não é atualizada. Após a atualização da

posição através da detecção de um marco, o sistema de localização absoluta é interrompido por 5 segundos para evitar que o mesmo marco seja detectado enquanto a cadeira está realizando a correção da trajetória.

Foram colocados no ambiente 5 marcos artificiais, a Tabela 27 mostra a posição em que cada marco foi posicionado. Além desses marcos, também foi disposto um marco na entrada do ambiente na posição (2, 1,8), em metros, com orientação de -90 graus, para a determinação da localização inicial da cadeira.

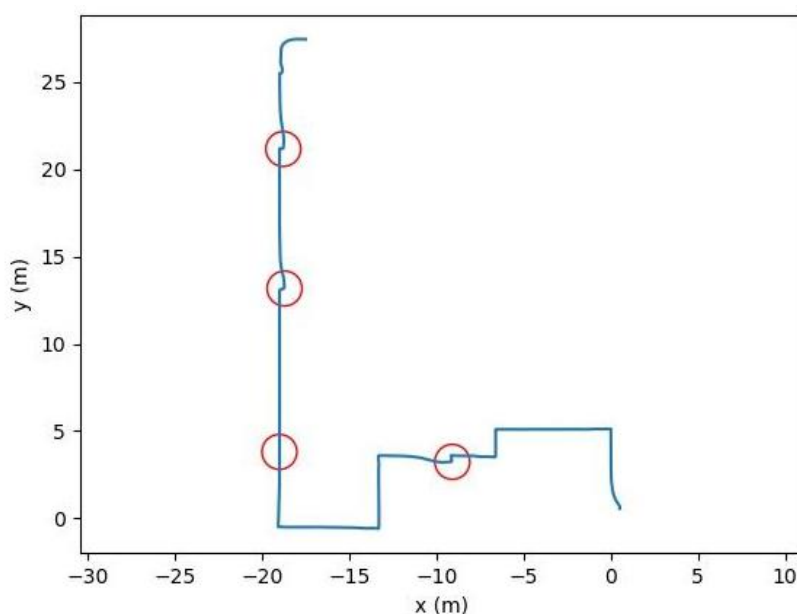
Tabela 27 - Posições dos marcos artificiais

MARCO	X (m)	Y (m)	α (graus)
1	-10,848	2,154	180
2	-20,516	3,054	90
3	-17,476	7,284	-90
4	-20,516	16,294	90
5	-17,476	23,544	-90

Fonte: autor (2021).

A Figura 72 mostra a trajetória registrada pelo sistema no primeiro ensaio do sistema de navegação autônoma implementado. Os círculos em vermelho indicam as regiões onde a posição da cadeira foi atualizada através da detecção de marcos artificiais.

Figura 72 - Trajetória no ambiente no ensaio 1



Fonte: autor (2021).

A Tabela 28 mostra a comparação entre a última posição registrada pela odometria e a posição corrigida pelo sistema de localização absoluta em cada marco detectado. As componentes (x_1, y_1, θ_1) representam a posição obtida com a odometria e (x_2, y_2, θ_2) representam a posição atualizada por meio da localização do marco. O erro é a distância euclidiana entre as coordenadas (x_1, y_1) e (x_2, y_2) e o erro θ é a diferença entre a orientação registrada pela odometria e a obtida através do sistema de localização absoluta. Os resultados dos demais ensaios podem ser vistos no Apêndice B.

Tabela 28 - Erro da posição registrada pelo sistema no ensaio 1

MARCO	X1 (m)	Y1 (m)	θ_1 (graus)	X2 (m)	Y2 (m)	θ_2 (graus)	Erro (m)	Erro θ (graus)
1	-9,17	3,60	179,09	-9,09	3,24	182,33	0,37	3,24
3	-19,00	3,88	89,61	-19,01	3,81	89,24	0,07	-0,37
4	-19,00	13,08	90,08	-18,72	13,16	92,54	0,30	2,46
5	-19,01	21,18	90,58	-18,78	21,15	85,83	0,22	-4,75

Fonte: autor (2021).

Por meio dos resultados obtidos observou-se que as diferenças entre as posições registradas pela odometria e a medida pelo sistema de localização de marcos são significativas, sendo que no pior caso foi registrada um erro 0,83 metros e uma diferença angular de 11,7 graus. É importante destacar que a maior parte da diferença da posição é causada devido aos erros de medidas do sistema de localização absoluta provocados pelo desalinhamento entre a imagem RGB e de profundidade.

Apesar dos erros na estimativa da posição cadeira no ambiente, nos 5 ensaios realizados o sistema de navegação autônoma implementado conseguiu chegar ao destino desejado e realizar a passagem pela porta da sala. Entretanto, os erros da estimativa da posição da cadeira causado pela falta de sincronia entre a captura da imagem RGB e de profundidade pelo Kinect podem resultar em colisão, tornando o sistema não confiável.

O sistema de localização absoluta por marcos artificiais consegue fornecer uma boa estimativa da posição da cadeira quando ela está parada, pois nesse caso o desalinhamento da imagem RGB-D não ocorre. Com isso, o sistema apresenta um desempenho satisfatório na determinação da posição inicial da cadeira no ambiente.

Todavia, constatou-se que a determinação da orientação é um ponto crítico na navegação autônoma de sistemas robóticos que utilizam a odometria, uma vez que um pequeno erro na estimativa da orientação resulta em grandes erros na estimativa da posição ao deslocar-se por grandes distâncias. Desta forma, mesmo que o desalinhamento das imagens RGB-D não ocorresse seria necessário uma grande quantidade de marcos no ambiente para corrigir a posição da cadeira, entretanto o sistema de navegação seria mais confiável.

6 CONCLUSÕES

Este trabalho buscou avaliar um sistema de navegação autônoma de uma cadeira de rodas motorizada, capaz de se deslocar entre corredores e portas. O sistema de navegação implementado utiliza um sensor Kinect v2 para determinar a posição absoluta da cadeira no ambiente por meio da detecção de marcos artificiais e para localizar portas. A posição relativa da cadeira durante a trajetória é estabelecida através da odometria com o uso de *enconders* nas rodas, giroscópio e acelerômetro.

A partir dos ensaios do controle de velocidade por meio de um controlador PI, verificou-se que devido às características do controle de velocidade nativo da cadeira motorizada, o controlador de velocidade com realimentação projetado possui uma resposta mais rápida para a aceleração do que para a desaceleração, o que prejudica a dinâmica de controle de trajetória da cadeira.

Em relação ao sistema de localização absoluta por marcos artificiais constatou-se que o sistema possui um desempenho satisfatório quando as imagens RGB-D são capturadas quando a cadeira está parada. Todavia, quando o marco é localizado enquanto a cadeira está em movimento ocorre erros significativos na estimativa da posição da cadeira, devido ao desalinhamento das imagens RGB e de profundidade causado pela dessincronização da captura entre as imagens. Com isso, conclui-se que a aplicação do sistema de localização absoluta durante a execução da trajetória não é confiável.

O controle de trajetória desenvolvido para a realização da passagem da cadeira por portas consegue alcançar o objetivo, desde que a cadeira esteja a uma distância suficiente da parede para realizar as manobras com segurança. Entretanto, o sistema de localização de portas possui erros na estimativa da orientação da cadeira em relação à porta, pois os pontos que são localizados como as bordas da porta não ficam alinhados paralelamente com a porta. Esse erro faz com que a cadeira realize a passagem pela porta de forma oblíqua.

Um ponto a ser observado é que a trajetória realizada pela cadeira na passagem pela porta diverge da trajetória simulada obtida através do modelo cinemáticos de robôs diferenciais. Essa diferença já era esperada, uma vez que o modelo teórico não considera a dinâmica de aceleração da cadeira de rodas motorizada utilizada.

Através dos testes de aplicação de filtros de ruídos nas imagens de profundidade, observou-se que os filtros não resultam em melhorias significativas que justifiquem sua aplicação tanto no sistema de localização de marcos quanto no sistema de localização de portas.

No teste do sistema de navegação autônoma completo, desde o ponto de partida da cadeira até a passagem pela porta do destino selecionado, em todos os ensaios a cadeira conseguiu atingir o objetivo. Apesar disso, verificou-se que os erros da estimativa da posição absoluta obtida pelo sistema de localização por marcos artificiais podem fazer com que a cadeira corrija a trajetória de forma errônea, podendo resultar em colisão.

Desta forma, conclui-se que a odometria por meio de *encoders*, giroscópio e acelerômetro juntamente com sensor Kinect mostra potencial na aplicação de navegação autônoma de robôs móveis. Entretanto é necessário encontrar soluções para o sincronismo das imagens RGB e de profundidade capturadas pelo Kinect, para que as imagens possam ser utilizadas para a localização de marcos artificiais durante a realização da trajetória.

Com a finalidade de dar sequência a este trabalho, uma sugestão seria ao invés de utilizar o sistema de acionamento dos motores nativo da cadeira de rodas motorizada, desenvolver um novo sistema de potência dos motores com controle de velocidade individual em cada roda. Isso melhoraria a dinâmica de controle de velocidade e também do controle de trajetória.

Na passagem por portas no sistema implementado a cadeira realiza a travessia de forma oblíquo. Para amenizar esse problema, uma alternativa melhor, seria determinar a posição da cadeira em relação à porta através da localização de um marco artificial posicionado ao lado da porta.

O maior problema do sistema de navegação autônoma está no desalinhamento das imagens RGB-D causado pelo fato de a biblioteca utilizada para a captura das imagens não garantir a sincronia entre a imagem RGB e a de profundidade. Uma possibilidade para a resolução do problema, seria modificar a biblioteca *freenect2* para registrar a hora em que cada imagem foi capturada. Com isso seria possível buscar imagens sincronizadas armazenadas em um buffer. Contudo este processo pode demandar muito tempo, o que pode torná-lo inviável.

REFERÊNCIAS BIBLIOGRÁFICAS

- BECKER, Marcel. **Aplicação de Tecnologias Assistivas e Técnicas de Controle em Cadeira de Rodas Inteligentes**. 2000. 192 f. Tese (Doutorado em Engenharia Mecânica) - Universidade Estadual de Campinas, Campinas, 2000.
- BEN-ARI, Mordechai; MONDADA, Francesco. **Elements of Robotics**. Switzerland: Springer, 2018, 324 p. ISBN 978-3-319-62532-4.
- BEZZERA, Glauber Gomes. **Localização de um Robô Móvel Usando Odometria é Marcos Naturais**. 2004. 112 f. Dissertação (Mestrado em Ciências) - Universidade Federal do Rio Grande do Norte, Natal, 2004.
- BORENSTEIN, Johann; FENG, Liqiang. Measurement and correction of systematic odometry errors in mobile robots. **IEEE Transaction on Robotics and Automation**, v. 12, n. 6, p. 869-880, dez. 1996.
- BOSCHETTI, Vinicius Klering. **Navegação de cadeira de rodas em passagens por portas com o uso do sensor Kinect**. 2019. 94 f. TCC (Bacharelado em Engenharia Elétrica) – Universidade de Caxias do Sul, Bento Gonçalves, 2019.
- BRAGA, Rodrigo Antonio Marques. **Plataforma de desenvolvimento de cadeiras de rodas inteligentes**. 2010. 246 f. Dissertação (Doutorado em Engenharia Informática) - Universidade do Porto, Porto, Portugal, 2010.
- CASTRO, André Luiz Figueiredo. **Uma nova abordagem para identificação e reconhecimento de marcos naturais utilizando sensores RGB-D**. 2017. 99 f. Dissertação (Mestrado em informática) - Universidade Federal da Paraíba, João Pessoa, 2017.
- DA MOTTA, Everton Simões. **Desenvolvimento de um método para captura de movimentos humanos usando uma câmera RGB-D**. 2016. 68 f. Dissertação (Mestrado em Ciência da Computação) - Universidade Estadual Paulista, São José do Rio Preto, 2016.
- DA SILVA, João Fernando Custódio; BARBOSA, Ricardo Luís; GALLIS, Rodrigo Bezerra de Araújo; PEREIRA, Leonardo Marino. Avaliação da qualidade da detecção de bordas em uma sequência de imagens de ruas e rodovias. **Revista Brasileira de Cartografia**, v. 56, n. 2, p. 96-103, dez. 2004.
- DE BARROS FILHO, Emânuel Guerra. **Controle inteligente aplicado a uma mesa de coordenadas de dois graus de liberdade**. 2011. 105 f. Dissertação (Mestrado em Engenharia Elétrica e Computação) - Universidade Federal do Rio Grande do Norte, Natal, 2011.
- EISENKRAEMER, Mateus Felipe. **Sistema de detecção de veículos e avaliação de eixos por meio de máquina de vetor de suporte**. 2016. 71 f. TCC (Bacharelado em Ciência da Computação) - Universidade de Santa Cruz do Sul, Santa Cruz do Sul, 2016.

FABRO, João Alberto. **Grupos neurais e sistemas nebulosos: aplicação à navegação autônoma**. 1996. 64 f. Dissertação (Mestrado em Engenharia Elétrica) - Universidade Estadual de Campinas, Campinas, 1996.

FANKHAUSER, Péter; BLOESH, Michael; RODRIGUEZ, Diego; KAESTNER, Hutter; SIEGWART, Roland. Kinect v2 for mobile robot navigation: evaluation and modeling. **2015 International Conference on Advanced Robotic**, Istanbul, Turquia, 2015. Disponível em: <https://ieeexplore.ieee.org/document/7251485>. Acesso em: 4 de jun. 2021.

FEHR, Linda; LANGBEIN, W. Edwin; SKAAR, Steven B. Adequacy of power wheelchair control interfaces for persons with severe disabilities: a clinical survey. **Journal of rehabilitation research and development**, v. 37, n. 3, p. 353-360, maio/jun. 2000.

PETRY, Marcelo Roberto. **Desenvolvimento de um protótipo e de metodologias de controlo de uma cadeira de rodas inteligente**. 2008. 77 f. Dissertação (Mestrado em Engenharia Eletrotécnica e de Computadores) - Universidade do Porto, Porto, Portugal, 2008.

RODRIGUES, Ana Rita Marques. **Planeamento de Trajetórias e Controlo de um Robô Omnidirecional**. 2017. 101 f. Dissertação (Mestrado em Engenharia Electrotécnica e Computadores) - Faculdade de Engenharia Universidade do Porto, Porto, Portugal, 2017.

SANDEEP, B.S.; SUPRIYA, P. Analysis of Fuzzy Rules for Robot Path Planning. **2016 Intl. Conference on Advances in Computing, Communications and Informatics**. Jaipur, India, set. 2016. Disponível em: <https://ieeexplore.ieee.org/document/7732065>. Acesso em: 4 jun. 2021.

SCHNEIDER, Danilo Gioacomin; STEMMER, Marcelo Ricardo. Sistema de localização de um robô móvel baseado em filtro de Kalman estendido para SLAM com Kinect em ambientes Internos. **Anais do Congresso Brasileiro de Automática**. v. 1, n. 1, 2019. ISSN 2525-8311.

SILVA, Rafael Leal. **Desenvolvimento de uma Interface Homem-Máquina Aplicada a uma Cadeira de Rodas Robótica por Meio de PDA**. 2007. 145 f. Dissertação (Mestrado em Engenharia Elétrica) - Universidade Federal do Espírito Santo, Vitória, 2007.

TANSCHKEIT, R. **Sistemas fuzzy**. Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro [2004?]. Disponível em: <http://paginapessoal.utfpr.edu.br/sumar/ensino/sistemas-fuzzy/sistemas-fuzzy/ICA-Sistemas%20Fuzzy.pdf>. Acesso em: 4 de jun. de 2021.

TOMASI JUNIOR, Darci Luiz. **Modelo de calibração para sistemas de odometria robótica**. 2016. 49 f. Dissertação (Mestrado em Informática) - Universidade Federal do Paraná, Curitiba, 2016.

VARELA, Renata Cristina Bertolozzi; OLIVER, Fátima Corrêa. A utilização de Tecnologia Assistiva na vida cotidiana de crianças com deficiência. **Ciência & Saúde Coletiva**, Rio de Janeiro, v. 18, n. 6, p. 1773-1784, 2013.

VIEIRA, Lucas Vinicius Coelho. **Navegação autônoma de cadeira de rodas motorizada em ambiente previamente conhecido**. 2019. 75 f. TCC (Bacharelado em Engenharia Elétrica) - Universidade de Caxias do Sul, Bento Gonçalves, 2019.

WORLD HEALTH ORGANIZATION. **Relatório mundial sobre a deficiência**; tradução Lexicus Serviços Lingüísticos. São Paulo: SEDPcD, 2012. 334 p. Disponível em:

https://apps.who.int/iris/bitstream/handle/10665/44575/9788564047020_por.pdf;jsessionid=6C99402A537DE2926F25B3B5A7F04FE?sequence=4. Acesso em: 4 de jun. de 2021.

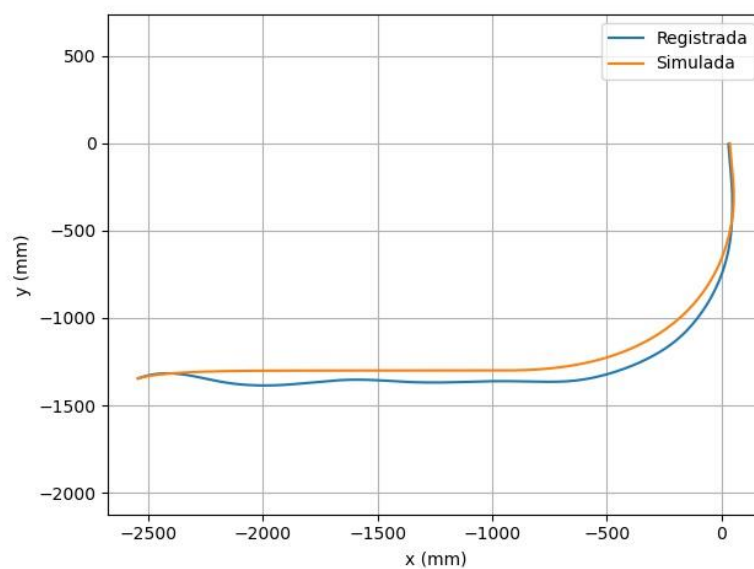
APÊNDICE A – RESULTADOS ENSAIO DE PASSAGEM POR PORTAS

Tabela 29 - Medidas trajetória posição 1

	X (mm)	Y (mm)	θ (graus)
Posição Inicial	-2455	-1510	30
Posição Inicial Registrada	-2545,0	-1345,8	25,3
Posição Final	75	35	97,4
Posição Final Registrada	31,3	-5,6	92,8

Fonte: autor (2021).

Figura 73 - Trajetória posição 1



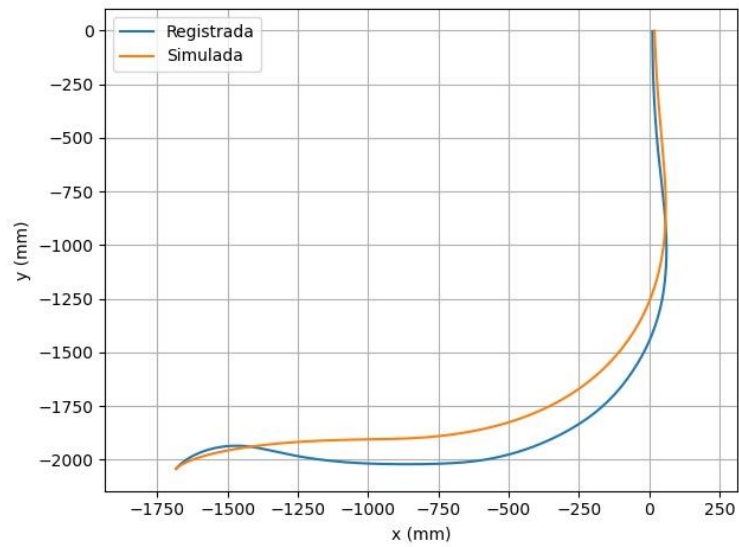
Fonte: autor (2021).

Tabela 30 - Medidas trajetória posição 3

	X (mm)	Y (mm)	θ (graus)
Posição Inicial	-1550	-2112	65
Posição Inicial Registrada	-1682,9	-2043,3	61,3
Posição Final	12	43	93,9
Posição Final Registrada	10,8	-3,6	90,2

Fonte: autor (2021).

Figura 74 - Trajetória posição 3



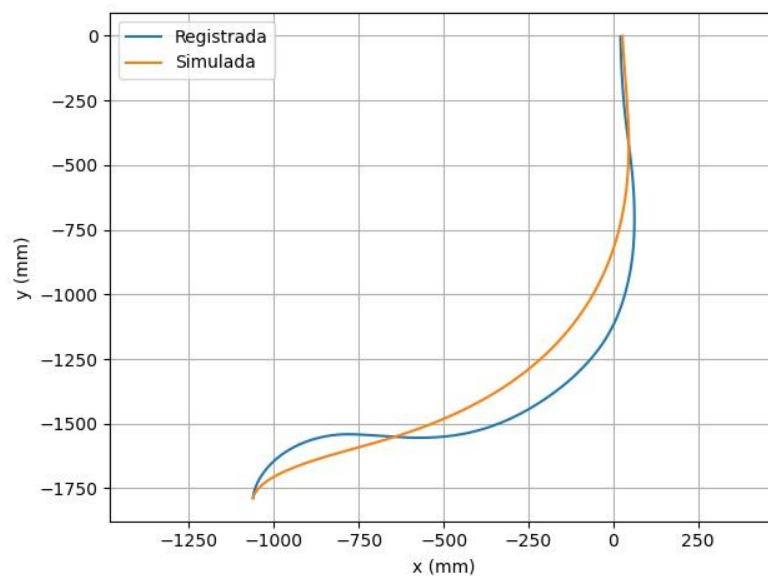
Fonte: autor (2021).

Tabela 31 - Medidas trajetória posição 5

	X (mm)	Y (mm)	θ (graus)
Posição Inicial	-945	-1812	90
Posição Inicial Registrada	-1061,5	-1787,4	86,7
Posição Final	60	39	94,9
Posição Final Registrada	20,8	-4,5	91,6

Fonte: autor (2021).

Figura 75 - Trajetória posição 5



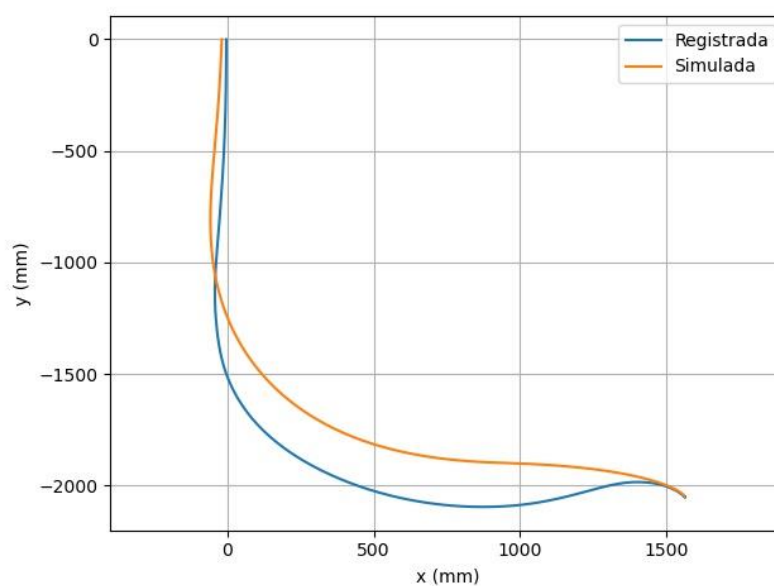
Fonte: autor (2021).

Tabela 32 - Medidas trajetória posição 10

	X (mm)	Y (mm)	θ (graus)
Posição Inicial	1462	-1812	110
Posição Inicial Registrada	1561,7	-2047,2	113,6
Posição Final	51	0	86,5
Posição Final Registrada	-3,2	-1,4	90,1

Fonte: autor (2021).

Figura 76 - Trajetória posição 10



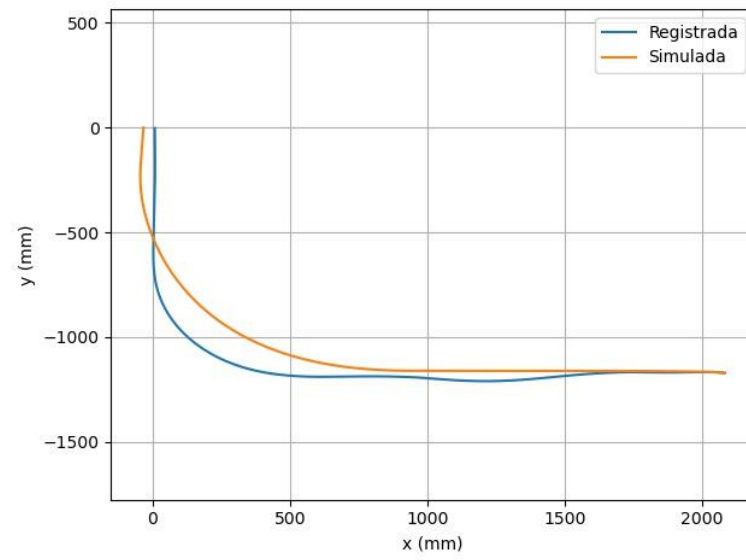
Fonte: autor (2021).

Tabela 33 - Medidas trajetória posição 11

	X (mm)	Y (mm)	θ (graus)
Posição Inicial	2064	-1207	170
Posição Inicial Registrada	2081,6	-1171,2	170,2
Posição Final	35	-38	90,5
Posição Final Registrada	7,2	-3,1	90,7

Fonte: autor (2021).

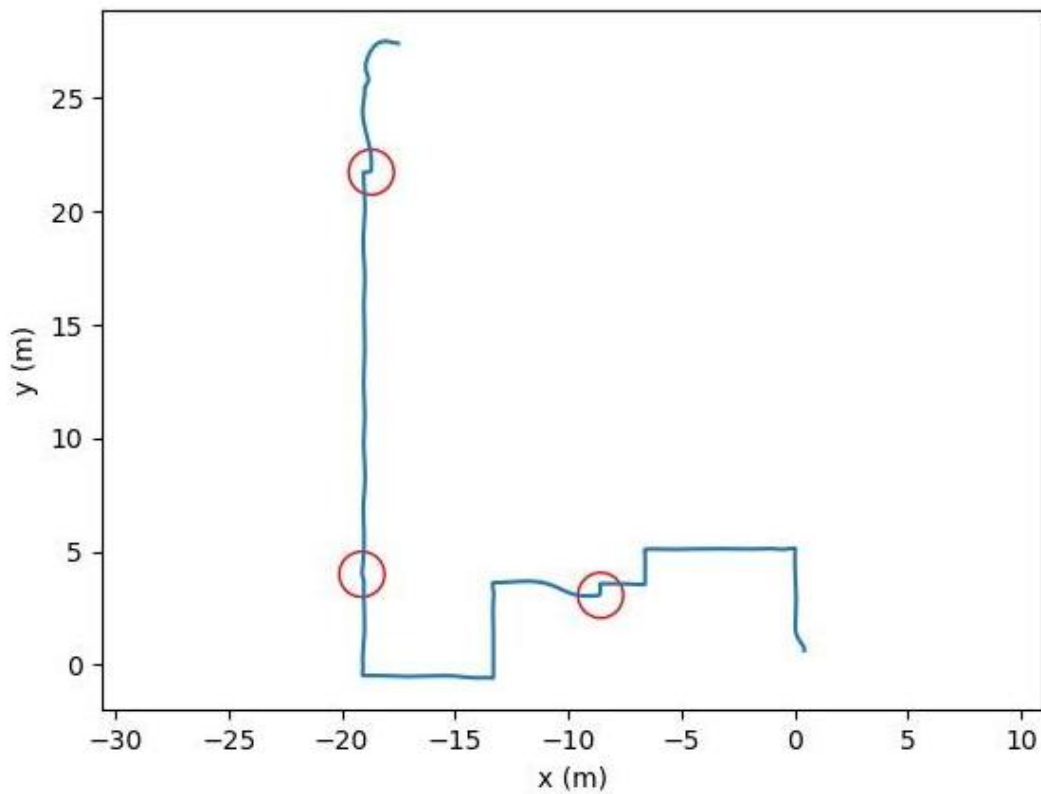
Figura 77 - Trajetória posição 11



Fonte: autor (2021).

APÊNDICE B – RESULTADOS ENSAIO DE NAVEGAÇÃO

Figura 78 - Trajetória no ambiente no ensaio 2



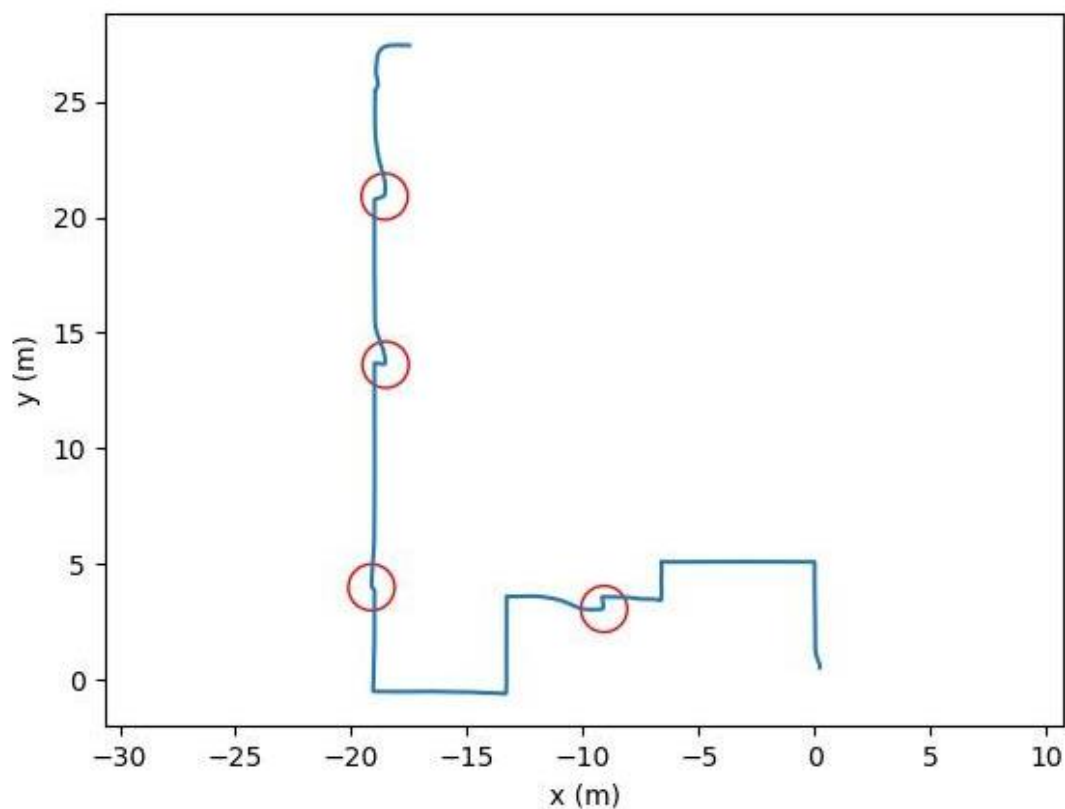
Fonte: autor (2021).

Tabela 34 - Erro da posição registrada pelo sistema no ensaio 2

MARCO	X1 (m)	Y1 (m)	θ_1 (graus)	X2 (m)	Y2 (m)	θ_2 (graus)	Erro	Erro θ (graus)
1	-8,59	3,58	178,50	-8,57	3,06	184,54	0,52	6,04
3	-19,02	3,68	86,97	-19,11	3,99	84,07	0,33	-2,90
5	-19,04	21,73	90,08	-18,69	21,74	89,26	0,35	-0,82

Fonte: autor (2021).

Figura 79 - Trajetória no ambiente no ensaio 3



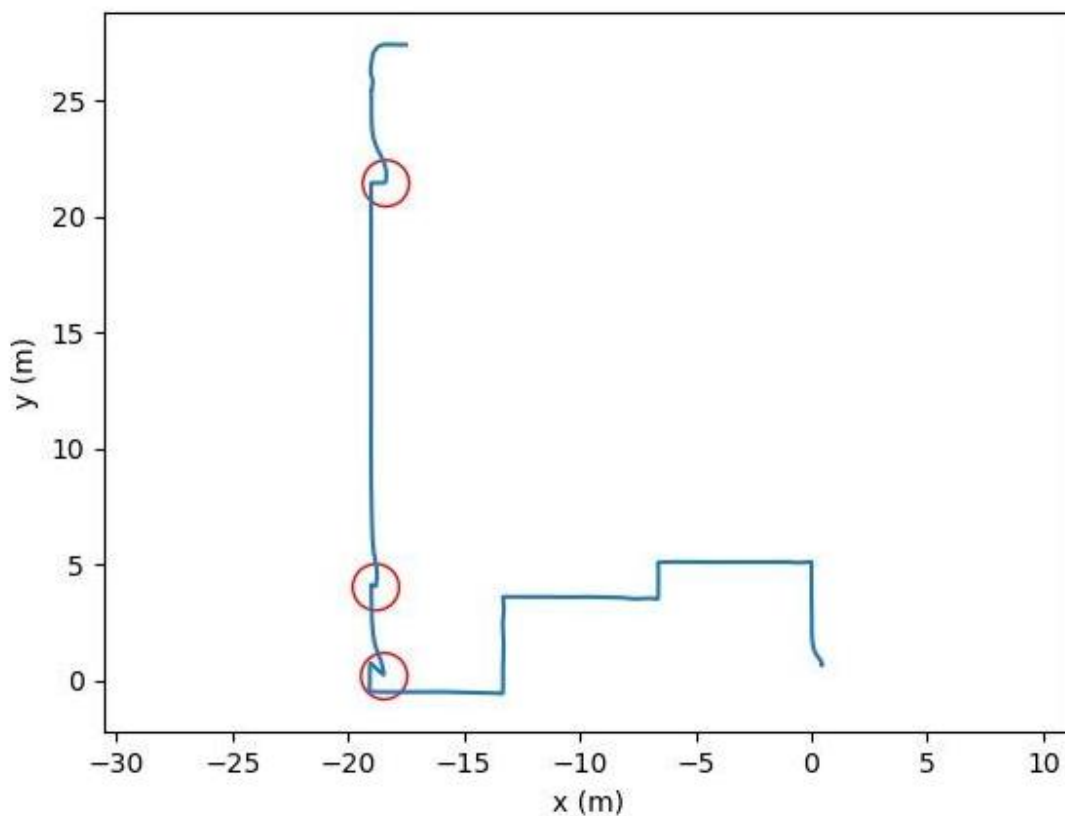
Fonte: autor (2021).

Tabela 35 - Erro da posição registrada pelo sistema no ensaio 3

MARCO	X1 (m)	Y1 (m)	$\theta 1$ (graus)	X2 (m)	Y2 (m)	$\theta 2$ (graus)	Erro (m)	Erro θ (graus)
1	-9,18	3,59	178,93	-9,09	3,07	184,21	0,53	5,29
3	-19,00	3,89	89,37	-19,13	4,00	88,31	0,17	-1,06
4	-19,00	13,69	90,13	-18,52	13,62	95,74	0,49	5,61
5	-19,00	20,76	90,02	-18,56	20,89	84,07	0,45	-5,95

Fonte: autor (2021).

Figura 80 - Trajetória no ambiente no ensaio 4



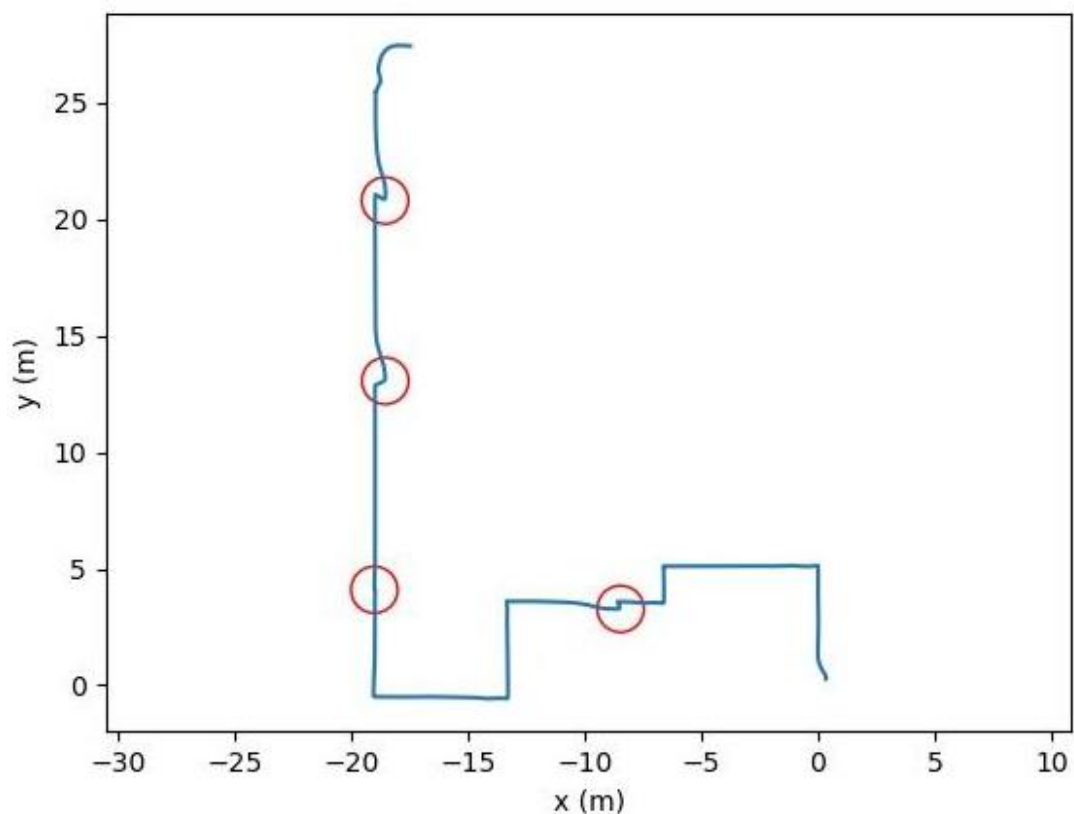
Fonte: autor (2021).

Tabela 36 - Erro da posição registrada pelo sistema no ensaio 4

MARCO	X1 (m)	Y1 (m)	$\theta 1$ (graus)	X2 (m)	Y2 (m)	$\theta 2$ (graus)	Erro (m)	Erro θ (graus)
2	-19,03	0,76	84,24	-18,44	0,18	95,94	0,83	11,70
3	-19,00	4,13	90,44	-18,79	4,02	83,54	0,23	-6,90
5	-19,00	21,47	90,03	-18,36	21,44	86,43	0,64	-3,60

Fonte: autor (2021).

Figura 81 - Trajetória no ambiente no ensaio 5



Fonte: autor (2021).

Tabela 37 - Erro da posição registrada pelo sistema no ensaio 5

MARCO	X1 (m)	Y1 (m)	$\theta 1$ (graus)	X2 (m)	Y2 (m)	$\theta 2$ (graus)	Erro (m)	Erro θ (graus)
1	-8,60	3,58	179,92	-8,47	3,27	180,93	0,34	1,01
3	-19,00	4,15	89,10	-19,03	4,09	88,18	0,07	-0,92
4	-19,00	12,87	90,06	-18,56	13,06	92,67	0,48	2,61
5	-19,00	21,07	89,38	-18,58	20,80	85,71	0,50	-3,67

Fonte: autor (2021).