

UNIVERSIDADE FEDERAL DE OURO PRETO  
INSTITUTO DE CIÊNCIAS EXATAS E BIOLÓGICAS  
DEPARTAMENTO DE COMPUTAÇÃO

ARTUR FERNANDES E CUNHA

Orientador: Prof. Dr. Jadson Castro Gertrudes

Coorientador: Prof. Dra. Valéria de Carvalho Santos

**UMA ABORDAGEM DE NAVEGAÇÃO TOPOLÓGICA DE ROBÔS  
MÓVEIS BASEADA EM ALGORITMO DE AGRUPAMENTO DE  
DADOS POR PARTICIONAMENTO**

Ouro Preto, MG  
2021

UNIVERSIDADE FEDERAL DE OURO PRETO  
INSTITUTO DE CIÊNCIAS EXATAS E BIOLÓGICAS  
DEPARTAMENTO DE COMPUTAÇÃO

ARTUR FERNANDES E CUNHA

**UMA ABORDAGEM DE NAVEGAÇÃO TOPOLÓGICA DE ROBÔS MÓVEIS  
BASEADA EM ALGORITMO DE AGRUPAMENTO DE DADOS POR  
PARTICIONAMENTO**

Monografia apresentada ao Curso de Ciência da Computação da Universidade Federal de Ouro Preto como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciência da Computação.

**Orientador:** Prof. Dr. Jadson Castro Gertrudes

**Coorientador:** Prof. Dra. Valéria de Carvalho Santos

Ouro Preto, MG  
2021



## FOLHA DE APROVAÇÃO

**Artur Fernandes e Cunha**

### **Uma abordagem de navegação topológica de robôs móveis baseada em algoritmo de agrupamento de dados por particionamento**

Monografia apresentada ao Curso de Ciência da Computação da Universidade Federal de Ouro Preto como requisito parcial para obtenção do título de Bacharel em Ciência da Computação

Aprovada em 23 de Agosto de 2021.

#### Membros da banca

Jadson Castro Gertrudes (Orientador) - Doutor - Universidade Federal de Ouro Preto  
Pedro Henrique Lopes Silva (Examinador) - Mestre - Universidade Federal de Ouro Preto  
Murillo Rehder Batista (Examinador) - Doutor - CTI Renato Archer

Jadson Castro Gertrudes, Orientador do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 23/08/2021.



Documento assinado eletronicamente por **Jadson Castro Gertrudes, PROFESSOR DE MAGISTERIO SUPERIOR**, em 24/08/2021, às 10:10, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do [Decreto nº 8.539, de 8 de outubro de 2015](#).



A autenticidade deste documento pode ser conferida no site [http://sei.ufop.br/sei/controlador\\_externo.php?acao=documento\\_conferir&id\\_orgao\\_acesso\\_externo=0](http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0), informando o código verificador **0210277** e o código CRC **E2DCD9F1**.

# Resumo

O uso de mapas topológicos para a tarefa de navegação de robôs móveis tem como finalidade simplificar a forma de controle de um robô, estabelecendo uma locomoção que parta de seu local inicial até o seu destino sem colidir em obstáculos e, ao mesmo tempo, sem exigir informações precisas do ambiente. A definição do controle da navegação topológica, em termos de ações, corresponde a comportamentos de reação local, como virar à esquerda ao encontrar um cruzamento. A partir da identificação do estado ao qual ele se encontra, é possível realizar uma nova ação e, dessa forma, seguir um percurso até o estado final. Considerando um mapa do tipo labirinto, um estado pode ser um corredor, uma curva à direita ou um cruzamento, por exemplo. Através dos sensores, o robô obtém dados do ambiente e um método de reconhecimento pode ser utilizado para identificar em qual estado o robô se encontra. Este projeto propõe o uso do algoritmo de agrupamento de dados por particionamento *K-means* para realizar a identificação de estado do robô. Algoritmos de agrupamento são capazes de identificar, sem o conhecimento prévio sobre classes (estados, neste caso), grupos naturais dentro do conjunto de dados. Como nenhum conhecimento prévio do estado é necessário, pode-se esperar que o algoritmo reconheça automaticamente o estado existente no mapa durante a fase de aprendizado. Durante a navegação, é esperado que o robô identifique o estado sem a intervenção de um especialista, ou seja, de forma autônoma. Após a validação dos resultados do algoritmo de agrupamento, a etapa de navegação do robô será executada no simulador robótico realístico, utilizando um robô *Pioneer3DX* com um sensor 2D Laser Scanner.

**Palavras-chave:** Agrupamento de dados por *K-means*, navegação topológica, robôs móveis.

# Abstract

The use of topological maps for mobile robot navigation aims to simplify the way of controlling a robot, establishing a locomotion that starts from its initial location to its destination without colliding with obstacles and, at the same time, without requiring accurate environmental information. The definition of topological navigation control, in terms of actions, corresponds to local reaction behaviors, such as turning left when encountering an intersection. After identifying the current state during navigation process, it is possible to perform a new action, and thus follow a path to the final state. Considering a maze-like map, a state can be a corridor, a right turn, or an intersection, for example. Through the sensors, the robot obtains data from the environment and a recognition method can be used to identify which state the robot is in. This project proposes the use of the *K-means* clustering algorithm to perform the robot's state identification. Clustering algorithms are able to identify, without prior knowledge about classes (states, in this case), natural groups within the data set. Since no prior knowledge to identify a new state is required, the algorithm can be expected to automatically recognize the current state on the map during the learning phase. During navigation, the robot is expected to identify the state without the intervention of a specialist. After validating the partitioning clustering results, the robot will be performed in a realistic robotic simulator, using a *Pioneer3DX* robot model with a 2D Laser Scanner sensor.

**Key words:** partitioning clustering by *K-means*, topological navigation, mobile robots.

# Lista de Ilustrações

Figura 2.1 – Representação dos centroides em um conjunto de dados para cada iteração. Fonte: Elaborado pelo autor. . . . .	6
Figura 2.2 – Posicionamento não otimizado de centroides iniciais. Fonte: Elaborado pelo autor. . . . .	8
Figura 2.3 – Representação dos dados 3-D para 2-D. Fonte: Elaborado pelo autor. . . . .	9
Figura 2.4 – Representação de um mapa métrico de ocupação. A figura (b) ilustra a sua composição por divisão em células regulares que representam o estado de ocupação no destaque ampliado (SANTOS, 2017). . . . .	12
Figura 2.5 – Representação de um mapa topológico em formato de grafo. Os nós representam locais característicos do ambiente e as arestas são a conectividade entre elas (SANTOS, 2017). . . . .	13
Figura 3.1 – Imagem Pioneer3DX com lidar Robotis Sick LMS 291. Fonte: elaborada pelo autor. . . . .	18
Figura 4.1 – Figura do mapa 1 com 1,6m de largura entre as paredes. Fonte: elaborada pelo autor. . . . .	21
Figura 4.2 – Figura do mapa 2 com 1,2m de largura entre as paredes. Fonte: elaborada pelo autor. . . . .	22
Figura 4.3 – Figura do mapa 3 com 2,4m de largura entre as paredes. Fonte: elaborada pelo autor. . . . .	23
Figura 4.4 – Figura do mapa 4 com 2m de largura entre as paredes. Fonte: elaborada pelo autor. . . . .	24
Figura 4.5 – Na figura de cima temos o percurso a ser realizado pelo robô, indicado pela seta, para realizar a captura de dados de estados referentes à encruzilhada. Abaixo temos o percurso a ser realizado para capturar estados do tipo corredor. Fonte: elaborada pelo autor. . . . .	25
Figura 4.6 – Representação de identificação dos estados em tempo real durante a navegação do robô. Fonte: elaborada pelo autor. . . . .	26
Figura 4.7 – Representação dos dados coletados do mapa 3 em PCA. Cada estado está sendo representado por uma cor diferente. Fonte: elaborada pelo autor. . . . .	30
Figura 4.8 – Representação correta do estado de encruzilhada à esquerda. Fonte: elaborada pelo autor. . . . .	31
Figura 4.9 – Representação errada do estado encruzilhada à esquerda. Fonte: elaborada pelo autor. . . . .	32
Figura 4.10 – Representação gráfica do <i>Ground Truth</i> da junção das bases de dados do mapa 1, mapa 2 e 3. Fonte: elaborada pelo autor. . . . .	33

Figura 4.11–Melhor percurso sugerido para alcançar um ponto final, partindo do ponto inicial . . . . . 34

# Lista de Tabelas

Tabela 4.1 – Descrição dos estados utilizados em cada configuração de banco de dados aplicados no projeto. . . . .	28
Tabela 4.2 – Resultado de validação dos agrupamentos pelo ARI do <b>mapa 1</b> . . . . .	28
Tabela 4.3 – Resultado de validação dos agrupamentos pelo ARI do <b>mapa 2</b> . . . . .	29
Tabela 4.4 – Resultado de validação dos agrupamentos pelo ARI do <b>mapa 3</b> . . . . .	29
Tabela 4.5 – Resultado de validação dos agrupamentos da junção do <b>mapa 1, mapa 2 e mapa 3</b> . . . . .	33



# Lista de Abreviaturas e Siglas

ABNT	Associação Brasileira de Normas Técnicas
DECOM	Departamento de Computação
UFOP	Universidade Federal de Ouro Preto
SSE	Sum of the Squared Error
PCA	Principal Component Analysis
PC	Principal Component
ARI	Adjusted Rand Index
RNA	Rede Neural Artificial
FSM	Finite State Machine
NCD	Normalized Compression Distance

# Lista de Símbolos

$\Gamma$	Letra grega Gama
$\Lambda$	Lambda
$\zeta$	Letra grega minúscula zeta
$\in$	Pertence

# Sumário

<b>1</b>	<b>Introdução</b>	<b>1</b>
1.1	Justificativa	2
1.2	Objetivos	2
1.3	Organização do Trabalho	2
<b>2</b>	<b>Revisão Bibliográfica</b>	<b>4</b>
2.1	Fundamentação Teórica	4
2.1.1	Algoritmo K-means	4
2.1.2	Reduzindo o SSE com pós-processamento	8
2.1.3	Análise de Componentes Principais (PCA)	9
2.1.4	Índice de Rand ajustado para validação dos agrupamentos	10
2.1.5	Navegação de robôs móveis	11
2.2	Trabalhos Relacionados	13
<b>3</b>	<b>Desenvolvimento</b>	<b>16</b>
3.1	Coleta de dados	16
3.2	Agrupamento de dados	16
3.2.1	Representação dos dados	17
3.3	Validação dos resultados	18
3.4	Navegação do robô	18
<b>4</b>	<b>Experimentos Computacionais</b>	<b>20</b>
4.1	Experimentos realizados	20
4.2	Resultados	27
4.2.1	Agrupamento de estados	27
4.2.2	Navegação do robô	30
<b>5</b>	<b>Considerações Finais</b>	<b>36</b>
5.1	Conclusão	36
5.2	Trabalhos Futuros	37
	<b>Referências</b>	<b>38</b>

# 1 Introdução

A tarefa de navegação de robôs móveis autônomos pode ser descrita pela capacidade de um robô de identificar, dentro de um ambiente, os estados referentes aos locais ao qual ele se encontra. As ações representam as tomadas de decisões, tendo como finalidade, definir o trajeto a ser percorrido para atingir o destino final. De maneira análoga, o robô adquire um comportamento mais inteligente, com características de aprendizado e adaptação. O uso da navegação autônoma é útil principalmente em ambientes de difícil acesso, como uma mina ou um local que desabou, pelo fato de, nestes cenários, ser possível a utilização do robô sem intervenção humana.

No trabalho anterior de Santos (2017), foi proposta uma abordagem híbrida para planejamento exploratório e execução de trajetórias de robôs móveis autônomos em ambientes indoor. A partir da exploração de mapas de composição topológica, mesmo apresentando maior abstração de dados do ambiente, é possível estabelecer uma navegação robótica de forma segura, ou seja, desenvolver uma mobilidade robótica que não haja colisão com obstáculos da cena. Dessa forma, as informações do ambiente são preservadas e a trajetória do robô é dada pelo encadeamento de comportamentos reativos locais.

Na navegação topológica, conforme a abordagem adotada em (SANTOS, 2017), o robô navega em um corredor e quando identifica uma mudança de estado, executa uma ação. Para identificar o estado atual do robô, algoritmos de reconhecimento de padrões são executados durante toda a navegação. Os algoritmos de reconhecimento de padrões investigados no trabalho de (SANTOS, 2017) foram: navegação com Redes Neurais Artificiais (SANTOS et al., 2014a), algoritmo baseado em Distância Normalizada de Compressão (SOARES et al., 2016) e algoritmo de Reconhecimento de Estados baseado em Ângulos de Referência (SANTOS; TOLEDO; OSÓRIO, 2014b).

Nos casos de identificação de estados utilizando Redes Neurais Artificiais (RNA), é requerido a rotulação de cada estado durante a fase de treinamento, além de ser uma abordagem supervisionada, envolvendo o conhecimento de um especialista. Os algoritmos por Distância Normalizada de Compressão não possuem a capacidade de reconhecer os estados ao qual o robô se encontra, apenas distingui-los se é um corredor ou não. O algoritmo de Reconhecimento de Estados baseado em ângulos de Referência é muito sensível à posição e rotação do robô.

O presente projeto de pesquisa tem como objetivo estender a abordagem proposta por Santos (2017), investigando o uso do algoritmo de agrupamento de dados *K-means* para identificação do estado do robô em ambientes indoor, baseado em mapas topológicos. Por ser uma abordagem não supervisionada, algoritmos de agrupamento de dados não exigem que os dados sejam rotulados para a fase de aprendizado. Isso reduz a interferência de um especialista e torna o sistema de navegação mais autônomo.

## 1.1 Justificativa

Existem poucos trabalhos relevantes dentro da literatura que utilizam algoritmos de agrupamento de dados com o auxílio de mapas topológicos. Os autores Nguyen, Starzyk e Goh (2013) apresentam um trabalho próximo a esta proposta, abordando o uso de agrupamento de dados para identificar locais de interesses a partir de mapas topológicos embasados em imagens capturadas por câmera. Entretanto, o custo computacional oriundo do processamento de imagens pode tornar o sistema caro. Neste projeto, as informações do ambiente são capturadas por sensores de menor custo computacional para processar.

Este projeto visa contribuir em pesquisas referentes ao uso de algoritmo de agrupamento de dados para navegação topológica, baseado em informações capturadas por sensores. Além de não exigir que os dados estejam rotulados na fase de aprendizagem, o uso da navegação não supervisionada irá permitir a navegação em ambientes de difícil acesso, como um local que sofreu um desastre. Locais com difícil captação de sinais de GPS, como áreas subterrâneas, podem vir a necessitar de métodos como este proposto para a exploração do ambiente.

## 1.2 Objetivos

O objetivo principal deste projeto de pesquisa é fornecer um método de navegação de robôs móveis baseado em agrupamento de dados em ambiente indoor, representado por um mapa topológico. A originalidade deste projeto está no uso do algoritmo de *K-means* para o contexto de navegação de robôs móveis em mapas topológicos.

Os objetivos específicos deste projeto são:

- Investigar o uso de algoritmos de agrupamento de dados no contexto de reconhecimento de estados do ambiente no qual o robô está inserido para navegação topológica;
- Gerar bases de dados obtidas pelo sensor do robô, compostas por amostras de cada tipo de estado do mapa em investigação;
- Aplicar o algoritmo de agrupamento de dados *K-means* nos bancos de dados coletados, para gerar grupos de dados que referenciam os estados.
- Utilizar a base de dados coletadas para validar o método proposto em simuladores robóticos realísticos.

## 1.3 Organização do Trabalho

Este trabalho é organizado da seguinte forma: no Capítulo 2 é apresentado o referencial bibliográfico, relacionado ao tema do projeto e necessário para o entendimento do trabalho. No

Capítulo 3 é apresentado de forma detalhada os procedimentos metodológicos que serão utilizados para alcançar o objetivo proposto. No Capítulo 4 são apresentados os resultados preliminares alcançados no trabalho. Finalmente, no Capítulo 5, é apresentada a conclusão obtida após a pesquisa investigada neste projeto.

## 2 Revisão Bibliográfica

Este Capítulo destina-se à revisão bibliográfica e servirá como base para contextualizar a pesquisa proposta, apresentando os principais conceitos relativos ao tema. A Seção 2.1 aborda a fundamentação teórica. A Seção 2.2 discorre sobre os trabalhos relacionados a pesquisa.

### 2.1 Fundamentação Teórica

O agrupamento de dados (*clustering*) tem por objetivo descobrir, sem conhecimento prévio sobre o conjunto de dados, grupos naturais em que elementos de um mesmo grupo sejam semelhantes (ou relacionados) entre si e diferentes (ou não relacionados) aos elementos de outros grupos (TAN; STEINBACH; KUMAR, 2016). Por meio da mineração de dados, realizamos agrupamentos de forma automatizada segundo o seu grau de similaridade. A aplicação desta abordagem também é ampla em campos como reconhecimento de padrões, análises de imagens, bioinformática, compressão de dados, computação gráfica e *machine learning*. Na Subseção 2.1.1 abordamos um dos principais métodos de agrupamento utilizados na área de mineração de dados: o algoritmo *K-means*. Além disso, a Subseção 2.1.2 relata as possíveis estratégias para sua otimização.

Para que o algoritmo de agrupamento tenha um funcionamento adequado, em alguns casos, faz-se necessária a transformação dos dados, de modo a remover redundâncias e correlações entre os atributos. Por isso, na Subseção 2.1.3 é apresentada a técnica de Análise de Componentes Principais (*Principal Component Analysis* - PCA), um dos principais métodos utilizados no processo de transformação de dados.

Índices de validação são utilizados em cenários em que seja necessária a avaliação do agrupamento de forma quantitativa. Tal validação pode ser por meio da análise de relações entre os elementos dos grupos (validação interna ou relativa) e por meio da comparação com um referencial externo, denominado validação externa. Na Subseção 2.1.4 será apresentado o Índice de Rand Ajustado (*Adjusted Rand Index* - ARI) para validação dos agrupamentos.

Todo aparato descrito anteriormente (algoritmo de *K-means*, método de otimização, transformação e validação) será utilizado para aplicação ao problema da navegação de robôs móveis, de forma a apresentar ao final do projeto uma abordagem não supervisionada para a navegação de robôs. O tema será abordado na Seção 2.1.5.

#### 2.1.1 Algoritmo K-means

O algoritmo *K-means* possui um modelo de aprendizagem não-supervisionado. Como o próprio nome já diz, a idéia é determinar um número  $k$  de grupos de dados (*clusters*) a partir

de um banco de dados que contém  $n$  elementos. Cada um destes grupos serão caracterizados a partir das similaridades apresentadas entre os dados neles inseridos. Neste algoritmo, definimos em cada um dos grupos a presença de um centroide, definido pelo ponto central de um grupo. A tendência é que, no final, cada grupo contenha os pontos mais próximos ao seu respectivo centroide.

Para o funcionamento do algoritmo *K-means*, primeiramente é definido como parâmetro inicial os  $k$  centroides, inicializados de forma aleatória dentro do conjunto de dados. Ao final, os centroides serão compostos pela média dos pontos contidas em um grupo de dados. Cada ponto será atribuído ao centroide mais próximo, e cada coleção de pontos atribuída a um centroide é considerada um grupo (TAN; STEINBACH; KUMAR, 2016). A idéia é fazer com que os grupos sofram eventuais alterações baseadas nesses pontos, até que não ocorra nenhuma mudança de pontos entre um grupo e outro, de forma que os valores dos centroides permanecem inalteráveis. De maneira geral, podemos descrever os passos do algoritmo da seguinte maneira:

---

**Algorithm 1** Algoritmo básico de K-means.

---

- 1: Selecionar  $k$  pontos como centroides iniciais
  - 2: **repeat**
  - 3:     Formar os  $k$  grupos atribuindo cada ponto ao seu centroide mais próximo.
  - 4:     Recomputar o centroide de cada grupo.
  - 5: **until** centroides não se alterem.
- 

No primeiro passo da ilustração representada na Figura 2.1, é usado como parâmetro inicial um valor de três centroides. A representação final dos grupos é exibida após a quarta iteração, onde a localização dos centroides são indicadas pelo símbolo de "+" e cada ponto pertencente a um mesmo grupo é representado pelo mesmo marcador. Para cada iteração executada, cada ponto é atribuído ao seu centroide mais próximo. Após esta operação, o valor de cada centroide é definida pela média dos valores pertencentes ao seu respectivo grupo. O resultado desta operação pode ser observada a partir da redefinição da localização dos centroides durante as iterações 1, 2 e 3 da Figura 2.1. O algoritmo termina na iteração de número 4 pelo fato de não haver mais ocorrência de mudanças, indicando que os centroides encontraram os agrupamentos naturais dos pontos.



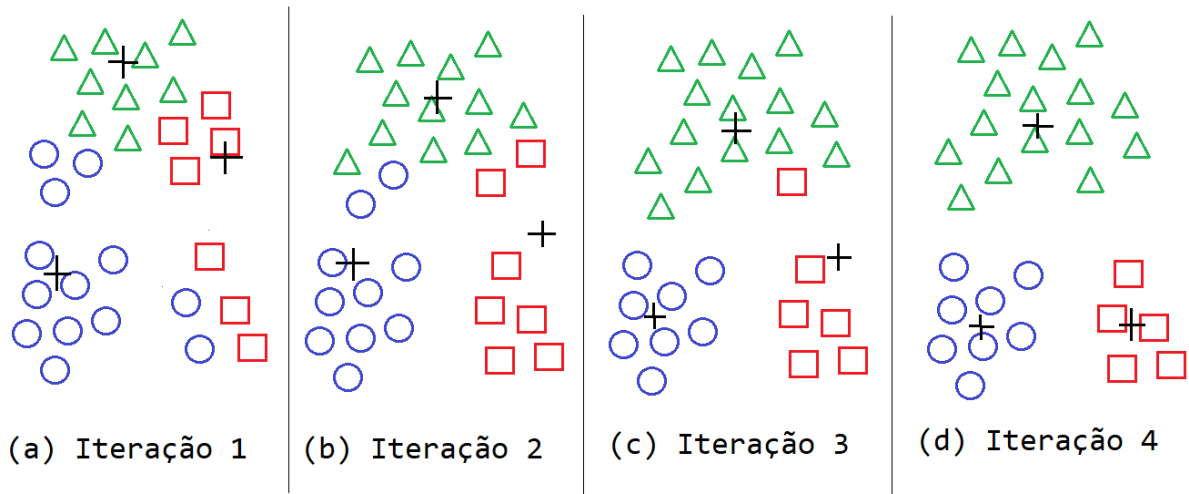


Figura 2.1 – Representação dos centroides em um conjunto de dados para cada iteração. Fonte: Elaborado pelo autor.

É possível então perceber que cada grupo poderá ter diferentes pontos comparados ao passo anterior. Resumidamente, esses novos pontos são submetidos ao processo iterativo até que eventualmente não haja mudança nos valores dos centroides, de forma a garantir que tais pontos foram agrupados precisamente. Como a maioria dos pontos convergem nas primeiras iterações, é bastante comum que a linha 5 do Algoritmo 1 seja substituída por condições mais simples, como por exemplo, repetir até que somente 1% ou menos dos pontos sofram as alterações de um grupo para outro (TAN; STEINBACH; KUMAR, 2016). Outro critério de parada seria determinar previamente um determinado número de iterações para serem executadas.

Para que seja possível atribuir os pontos ao seu centroide mais próximo, é preciso aplicar algum tipo de medida de proximidade, de modo a estabelecer uma distância dos pontos aos centroides. Uma das aplicações seria o uso da Distância Euclidiana ou o uso da similaridade do cosseno. No entanto, o último é um pouco mais apropriado para documentos, assim como a medida de Jaccard.

Para aplicar a medida de proximidade no Espaço Euclidiano, é preciso estabelecer uma função objetivo. No caso demonstrado no livro de Tan, Steinbach e Kumar (2016), a função objetivo irá medir a qualidade dos agrupamentos de dados por meio da soma dos erros ao quadrado (*Sum of the Squared Error* - SSE). Nele, para calcular o erro dos pontos, pode ser aplicado a Distância Euclidiana de cada ponto ao centroide mais próximo e depois computado a soma total dos erros quadráticos. Esta soma é chamada de SSE total. Quanto menor for este erro quadrático, melhor será a representação dos pontos em seu grupo. A SSE total é formalmente

definida da seguinte forma:

$$SSE = \sum_{i=1}^K \sum_{x \in C_i} dist(c_i, x)^2, \quad (2.1)$$

onde  $dist$  é a Distância Euclidiana entre um ponto  $x$  ao centroide  $c_i$  do seu respectivo grupo  $C_i$ , delimitado pelo número  $K$  de grupos. Sabemos que o valor do centroide é definido pelo somatório de todos os pontos pertencentes ao seu respectivo grupo, dividido pela quantidade de pontos nela contida. Este valor é representado na Equação 2.2, onde  $m_i$  representa o número de pontos dentro do  $i$ -ésimo grupo.

$$c_i = \frac{1}{m_i} \sum_{x \in C_i} x \quad (2.2)$$

A minimização da SSE já é realizada durante as iterações executadas nos passos 3 e 4 do Algoritmo 1, porém só é garantido encontrar o seu mínimo local, já que em cada momento são efetuadas escolhas de pontos e centroides específicos, ao invés de todas as possíveis escolhas (TAN; STEINBACH; KUMAR, 2016).

Para extrair os melhores resultados do algoritmo *K-means*, é necessário definir as posições iniciais dos centroides antes da execução. Uma das abordagens mais comum é escolhê-los de forma aleatória. Porém, isto não garante que os grupos resultantes irão apresentar uma boa qualidade. É possível entender que diferentes execuções do algoritmo *K-means* com posições de centroides iniciais aleatórios irão tipicamente produzir valores diferentes de SSEs, já que o seu erro é baseado na distância entre cada ponto e o seu centroide mais próximo. Tan, Steinbach e Kumar (2016) apresentam um exemplo, ilustrado na Figura 2.2, mostrando que, mesmo aparentando uma distribuição inicial melhor em comparação com a Figura 2.1 (em ambas as figuras, os centroides são representadas pelas cruces), o resultado da quarta iteração apresenta um agrupamento de dados subótimo, com um erro quadrático maior.

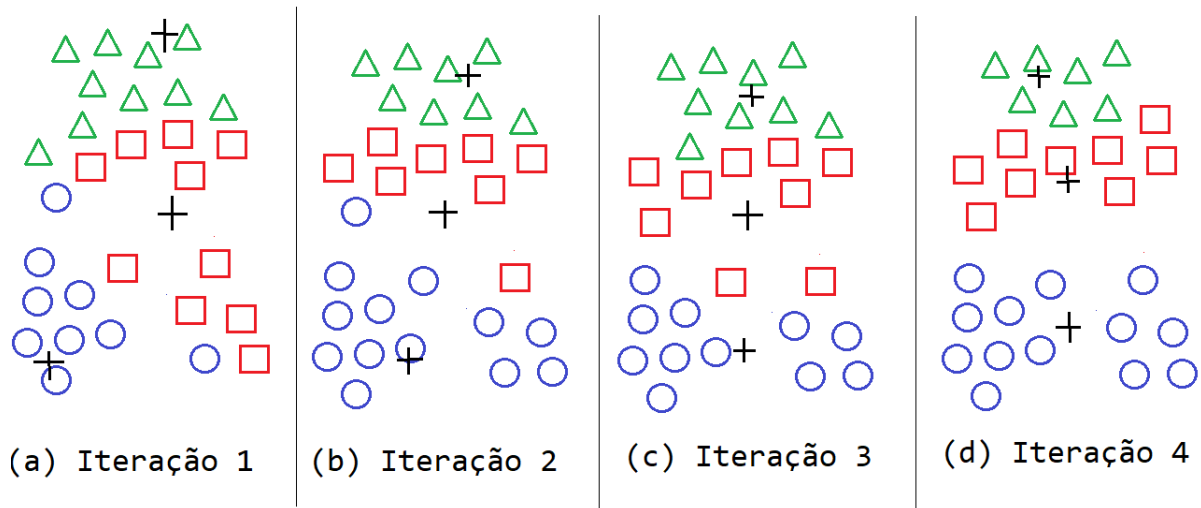


Figura 2.2 – Posicionamento não otimizado de centroides iniciais. Fonte: Elaborado pelo autor.

Uma estratégia que poderia ser utilizada para cobrir os limites da inicialização aleatória no *K-means* seria aplicar múltiplas execuções, no qual cada teste irá conter posições iniciais de centroides diferentes umas das outras. Dessa forma, é possível selecionar a execução em que o conjunto de grupos apresenta o menor valor total da SSE.

### 2.1.2 Reduzindo o SSE com pós-processamento

De acordo com [Bahnsen e Torroledo \(2017\)](#), a maneira mais simples de reduzirmos o valor de SSE seria aumentar o número de grupos, mas na maioria dos casos esta não é a forma mais apropriada ([TAN; STEINBACH; KUMAR, 2016](#)). Então, a maneira mais interessante seria otimizar o valor da SSE total, ou seja, deixar este valor no seu mínimo possível sem predefinir um valor grande de  $k$ . Tais métodos são possíveis pelo fato de o algoritmo de *K-means* tipicamente convergir para o mínimo local. Para isso, as estratégias focam na SSE individual de cada grupo, já que a SSE total se baseia na soma da SSE de todos os grupos.

Em ([TAN; STEINBACH; KUMAR, 2016](#)), duas estratégias são apresentadas para diminuir o SSE total pelo aumento do número de grupos:

- **Dividir um grupo:** O grupo que possuir o maior valor de SSE é escolhido para sofrer a divisão.
- **Introduzir um novo centroide de grupo:** O ponto que apresenta a maior distância de qualquer centro de grupo é geralmente escolhido para ser o novo centroide. Uma maneira de determinar tal ponto seria adicionando uma condição de rastreamento para obter o ponto de maior SSE durante as operações de distâncias.

Outras duas estratégias são utilizadas com a função de diminuir o número de grupos e, ao mesmo tempo, minimizar o valor total da SSE:

- **Dispersar um grupo:** A ideia é remover o centroide que corresponde ao grupo que provoca o maior aumento da SSE. Após a remoção, os pontos que pertenciam à ele são reatribuídos a outros grupos.
- **Fundir dois grupos:** Os grupos que apresentarem os centroides mais próximas entre si são escolhidos para a mesclagem. Outra opção seria fundir os dois grupos que apresentarem o menor aumento da SSE total.

### 2.1.3 Análise de Componentes Principais (PCA)

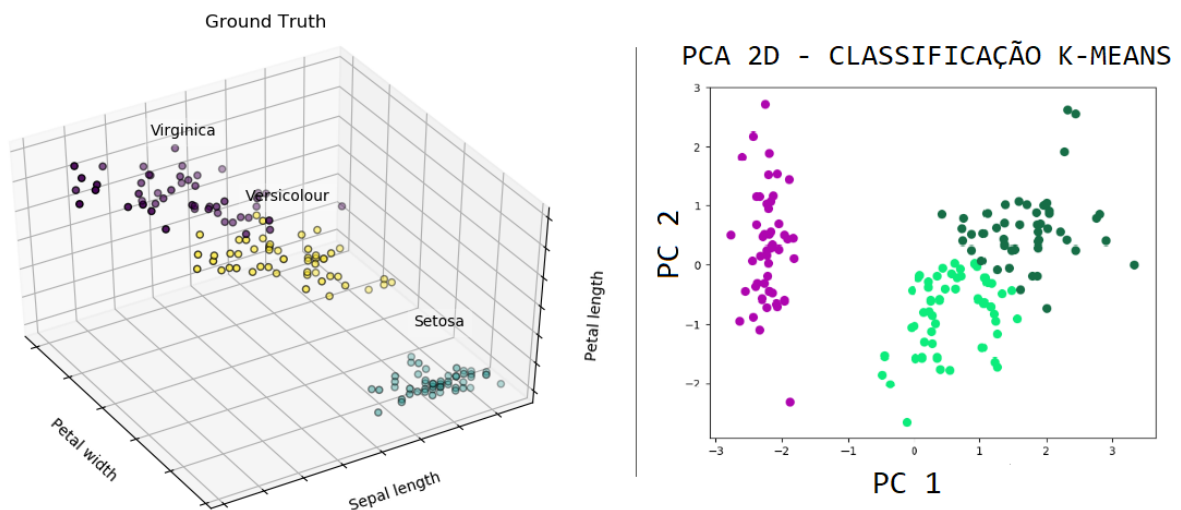


Figura 2.3 – Representação dos dados 3-D para 2-D. Fonte: Elaborado pelo autor.

De acordo com Tan, Steinbach e Kumar (2016), o objetivo do PCA (JOLLIFFE, 2011) é projetar os dados em um conjunto reduzido de dimensões, preservando o máximo possível de informações. A Figura 2.3 ilustra a redução de dimensão, projetando um conjunto de dados da dimensão 3-D para 2-D. A imagem da esquerda representa, em cada eixo, o comprimento da sépala, a largura e comprimento da pétala retirados de um banco de dados flores de íris (FISHER, 1936). A da direita é uma plotagem utilizando a transformação para Componentes Principais (*Principal Component - PC*), que é baseada na covariância e correlação da matriz de dados (JOLLIFFE, 2011), onde o número de dimensões dessa matriz foi reduzido para 2 componentes. Podemos utilizar PCA para encontrar a melhor aproximação planar de dados que possuem, por exemplo, 3 dimensões, ou a melhor aproximação de dados que possuem 100 dimensões, sendo bastante útil quando não há uma estrutura de grupo clara nos dados (JOLLIFFE, 2011).

### 2.1.4 Índice de Rand ajustado para validação dos agrupamentos

O ARI (HUBERT; ARABIE, 1985) é largamente utilizado com o intuito de avaliar a qualidade de uma partição produzida por um algoritmo de agrupamento de dados em relação a um resultado esperado pelo especialista no domínio (*Ground Truth*) (SANTOS; EMBRECHTS, 2009).

De acordo com Santos e Embrechts (2009), temos a formulação do Índice de Rand (*Rand Index* - RI) da seguinte forma:

$$RI = \frac{a + d}{a + b + c + d} \quad (2.3)$$

onde as variáveis representam as quatro possíveis ocorrências durante a verificação de paridade dos elementos existentes dentro grupo de dados *A*, que contém os elementos do *Ground Truth* e o grupo de dados *B*, que contém os elementos gerados por um algoritmo de agrupamento a ser validado. A variável *a* representa o número de pares iguais atribuídos entre um grupo de dados *A* e *B*. A variável *b* representa o número de pares iguais no grupo de dados *A* e diferentes no grupo de dados *B*. A variável *c* representa o número de pares diferentes atribuídos no grupo de dados *A* e iguais no grupo de dados *B*. E por fim a variável *d* representa o número de pares atribuídos diferentes tanto no grupo de dados *A* como no *B*.

O ARI é visto como uma versão melhorada do RI. Considerando um conjunto de *n* objetos, calculamos o seu ARI a partir do número total de possíveis combinações de pares  $\binom{n}{2}$  dado pelo conjunto, junto à representação dos resultados dos quatro tipos diferentes de pares vistos em RI. Embora o RI só possa produzir um valor entre 0 e 1, o ARI pode produzir valores negativos se o índice for menor do que o índice esperado (HUBERT; ARABIE, 1985), podendo ser representado pela equação 2.4 abaixo:

$$ARI = \frac{\binom{n}{2} (a + d) - [(a + b)(a + c) + (c + d)(b + d)]}{\binom{n}{2}^2 - [(a + b)(a + c) + (c + d)(d + b)]} \quad (2.4)$$

Nos códigos abaixo temos a chamada da função ARI como exemplo, onde o seu primeiro parâmetro é um vetor composto de rótulos definidos pelo *Ground Truth* e o segundo parâmetro é composto por rótulos gerados a partir do processo de agrupamento de dados (PEDREGOSA et al., 2011).

```
1 >>> from sklearn.metrics.cluster import adjusted_rand_score
2 >>> adjusted_rand_score([0, 0, 1, 1], [0, 0, 1, 1])
3 1.0
4 >>> adjusted_rand_score([0, 0, 1, 1], [1, 1, 0, 0])
5 1.0
```

Listing 2.1 – Exemplo de validação com resultado ótimo (PEDREGOSA et al., 2011).

```
1 >>> adjusted_rand_score([0, 0, 1, 2], [0, 0, 1, 1])
2 0.57...
```

Listing 2.2 – Validação penalizada por rotulagem errada (PEDREGOSA et al., 2011).

O resultado gerado a partir de uma avaliação é representado por um valor entre [0;1] (PEDREGOSA et al., 2011). No Código 2.1 é observado que rótulos que correspondem perfeitamente recebem pontuação total. Já no Código 2.2, é notado que, mesmo que o rótulo de identidade 1 tenha sido atribuído ao segundo grupo de forma correta pelo terceiro elemento, ele não está puro, pelo fato do quarto elemento ter sido classificado de forma errada. Portanto, há uma penalização pelo processo de avaliação.

O ARI é simétrico, então rotulagens que possuem grupos puros com elementos vindos das mesmas classes, mas divisões desnecessárias, são penalizadas (PEDREGOSA et al., 2011), conforme apresentado no Código 2.3:

```
1 >>> adjusted_rand_score([0, 0, 1, 1], [0, 0, 1, 2])
2 0.57...
```

Listing 2.3 – Validação penalizada por divisão desnecessária (PEDREGOSA et al., 2011).

Se os membros das classes são completamente divididos em diferentes grupos, a atribuição é totalmente incompleta, portanto, o ARI é muito baixo, conforme mostra o Código 2.4:

```
1 >>> adjusted_rand_score([0, 0, 0, 0], [0, 1, 2, 3])
2 0.0
```

Listing 2.4 – Exemplo de validação com resultado ruim (PEDREGOSA et al., 2011).

### 2.1.5 Navegação de robôs móveis

O problema da navegação de um robô móvel envolve a capacidade deste de agir com base em seus conhecimentos e informações captadas por sensores, com a finalidade de alcançar uma posição objetivo de forma mais eficiente e confiável possível (SIEGWART; NOURBAKHS; SCARAMUZZA, 2011). Para solucionar este problema da navegação, é necessário compreender e aplicar os conceitos que tratam o planejamento de trajetórias e desvio de obstáculos. A idéia geral é desenvolver o nível cognitivo do robô para representar tomadas de decisões que buscam a conclusão de seu objetivo. O planejamento da rota possui a finalidade de fazer com que o robô saia de uma posição inicial para uma posição final a partir da identificação de uma trajetória, de modo que ele consiga decidir o que fazer ao longo do percurso para atingir a sua meta. Já para não haver colisões com obstáculos no ambiente, é necessário modelar a trajetória do robô a partir das leituras dos sensores em tempo real.

O presente projeto tem como objetivo auxiliar a navegação baseada em mapas por meio do processamento dos sinais coletados pelo robô, onde estes servirão de referência para executar

estados como curva à direita, curva à esquerda, cruzamento, etc. Os tipos de mapas utilizados em robótica estão separados pelo nível de abstração dos elementos utilizados em sua construção.

De acordo com Santos (2017), um dos principais tipos de mapas são os mapas métricos, que apresentam baixo nível de abstração de seus elementos, possuindo então uma capacidade maior em fornecer informações espaciais com bastante detalhe da sua ambientação, podendo ser analisado com alta precisão as áreas livres para a navegação. Na imagem 2.4 temos a representação de um mapa métrico de ocupação, tendo este a função de limitar as áreas onde o robô pode navegar em um determinado ambiente.

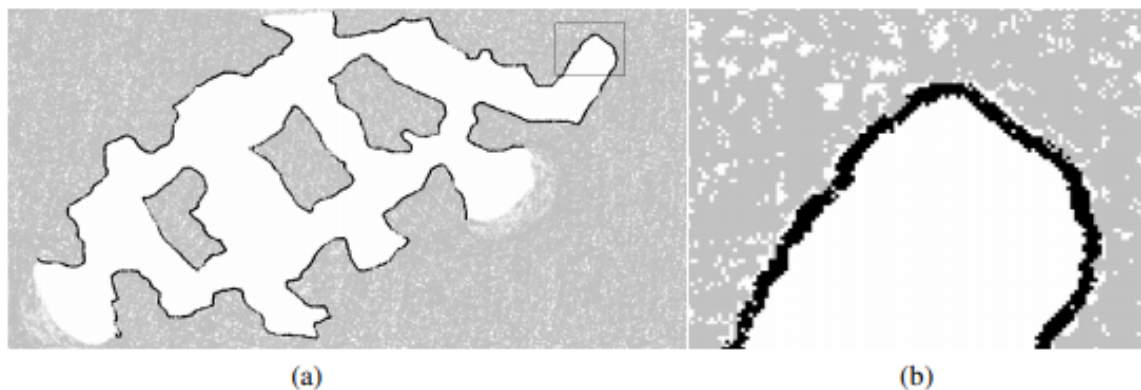


Figura 2.4 – Representação de um mapa métrico de ocupação. A figura (b) ilustra a sua composição por divisão em células regulares que representam o estado de ocupação no destaque ampliado (SANTOS, 2017).

Neste trabalho, serão utilizados mapas topológicos para tratar tal problema, por estes possuírem a função de representar a conectividade entre determinados locais do ambiente através de um alto nível de abstração dos elementos nele contido. Normalmente são constituídos de grafos onde os nós representam os pontos em que o robô terá de tomar decisões devido a presença de uma região de importância, como curvas ou bifurcações, e os arcos ou arestas representam a conexão entre os nós (SANTOS, 2017). Na Figura 2.5, nota-se uma composição com poucos detalhes, enfatizando regiões de interesse para navegação do robô.

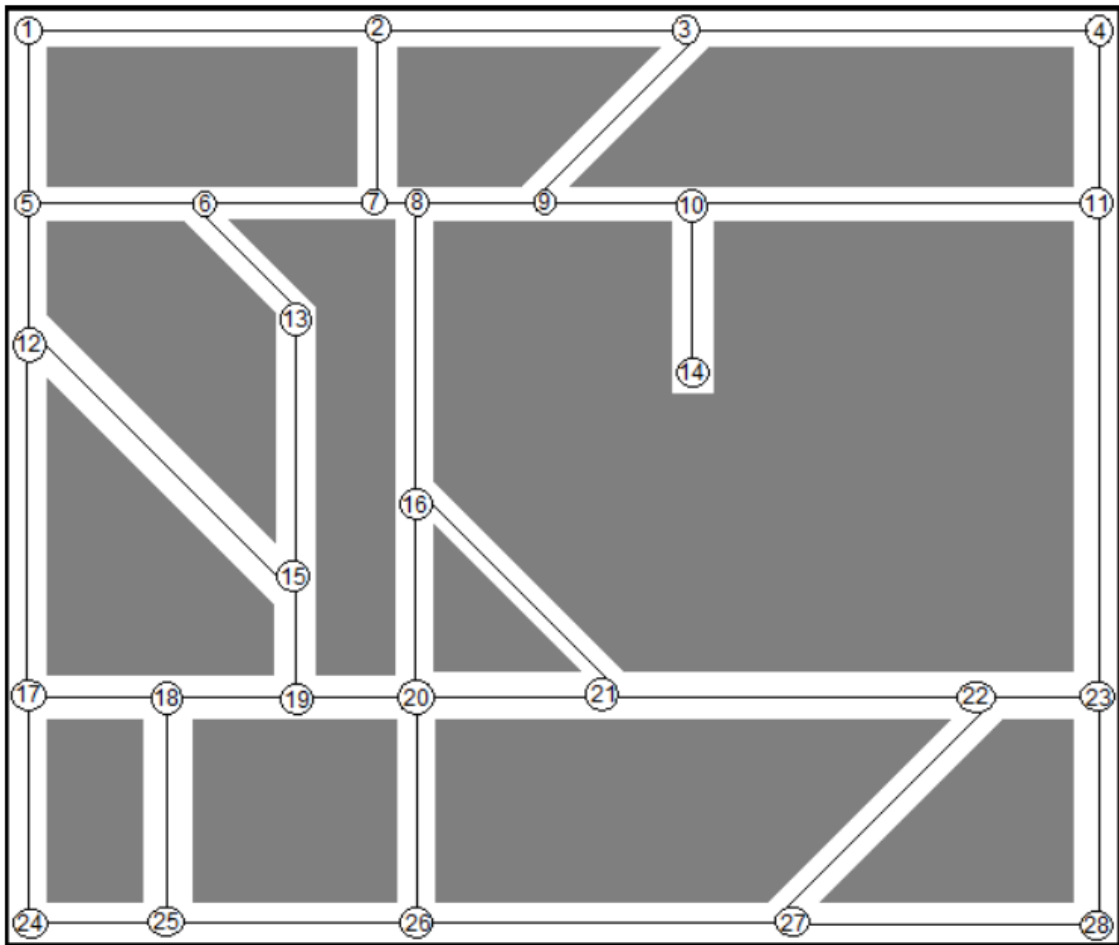


Figura 2.5 – Representação de um mapa topológico em formato de grafo. Os nós representam locais característicos do ambiente e as arestas são a conectividade entre elas (SANTOS, 2017).

## 2.2 Trabalhos Relacionados

O trabalho desenvolvido por Ravankar et al. (2012), tem como objetivo discutir como o agrupamento de dados por *K-means* pode ser efetivamente usado, apresentando vantagens e limitações. O autor utiliza agrupamento de dados para escaneamento por sensor de alcance a laser em seus experimentos. O ambiente experimental construído para a geração do mapa consiste em paredes retas. O robô é movido dentro da área para gerar os pontos de dados. Uma limitação encontrada foi que os ruídos presentes nos dados do sensor podem desorientar a formação do grupo de dados e gerar erros, ressaltando que é bastante importante reduzir ruídos antes da aplicação da técnica de agrupamento. É relatado que algoritmos de filtragem podem ajudar a solucionar este problema. Além disso, é adicionado que a redução de dados pode aumentar a precisão na fase de agrupamento de dados. Por final, conclui-se que o método *K-means*, apesar de ser uma técnica simples, é poderosa e eficiente para gerar mapas de linhas retas.



Em (HEAP; PAGNUCCO, 2012), os autores aplicaram agrupamento de dados por *K-means* com o agrupamento de dados por ligações simples (*Single linkage*), levando em consideração os efeitos de distância em linha reta e real, caminho a ser percorrido na formação dos grupos de dados para a alocação de um time de robôs autônomos móveis em um ambiente de escritório, com o objetivo de entregar mensagens entre salas de forma eficiente. Para o problema tratado, foi constatado que encontrar um bom valor de  $k$  é um desafio para ambos métodos, pelo fato de um valor grande de  $k$  resultar em grupos que contenham poucas tarefas, ou seja, seriam consideradas pouca sinergia entre as tarefas. Por outro lado, pequenos valores de  $k$  podem gerar grupos que contenham muitas tarefas e, conseqüentemente, os robôs acabam sendo sobrecarregados, quando na verdade seria mais adequado que tais tarefas fossem executadas por outros robôs. É concluído que uma abordagem de agrupamento de dados que busca um equilíbrio entre esses dois desafios seria ideal.

Outra forma de resolver o problema de planejamento de rota em ambientes desconhecidos é o método proposto por Saha e Dasgupta (2018), que desenvolveram um framework chamada *Incremental State Discovery via Clustering* (ISDC), que combina as técnicas de *option augmented meta-point affinity propagation* (OMP-AP) e *k-ésimo Vizinho mais Próximo* (*K-Nearest Neighbor* - K-NN) para identificar obstáculos em diferentes cenários, priorizando um tempo de planejamento e navegação acelerado para o robô. Nele, a idéia principal do OMP-AP é fazer com que os pontos de dados que estão próximos no espaço possam ser agrupados e substituídos por um único meta-ponto. Segundo Ott e Ramos (2013), um meta-ponto é considerado um ponto de um grupo, uma vez que representa pontos de dados brutos suficientes, caso contrário, é considerado um ponto de ruído. Com isso, é possível descartar pontos gerados a partir de observações aleatórias, como leituras falsas de um scanner a laser. Em outras palavras, é possível detectar e ignorar discrepâncias nas observações. O classificador K-NN determina o padrão de obstáculos combinado em uma dessas classes, de modo a determinar uma coleção de pontos 2D que irá representar o contorno do obstáculo percebido.

Na avaliação do algoritmo ISDC, Saha e Dasgupta (2018) utilizaram métricas de planejamento de tempo, tempo total e distância total. Planejamento de tempo se refere ao tempo que o robô gasta para classificar um tipo de obstáculo e selecionar a melhor opção disponível para este determinado estado. Tempo total combina o tempo de planejamento e o tempo gasto pelo robô para executar a navegação até a opção previamente determinada por ele. Distância total se refere ao percurso completo do robô desde o ponto de partida até o seu destino. Os resultados experimentais do ISDC validam os efeitos combinados das técnicas de OMP-AP e K-NN de forma positiva, quando comparados com a técnica de navegação hierárquica proposta pelos mesmos autores em (SAHA; DASGUPTA, 2017), ressaltando uma redução no planejamento geral do caminho e os tempos de navegação no robô, juntamente com a distância total reduzida, levando a uma eventual melhoria no desempenho do aprendizado.

Para a identificação de estados durante a navegação de robôs, foi investigado em (SANTOS,

2017) abordagens como a navegação com Redes Neurais Artificiais, algoritmo baseado em Distância Normalizada de Compressão e algoritmo de Reconhecimento de Estados baseado em Ângulos de Referência, no que tange à descoberta de novos caminhos, definindo uma rota a partir de um ponto de partida e um ponto de chegada.

Em (SANTOS, 2017), a navegação com Redes Neurais Artificiais foi utilizada no intuito de detectar os estados atuais e as mudanças de estado a partir dos sensores do robô. A idéia é que a rede neural artificial (RNA) consiga classificar os padrões de entrada referentes às estruturas do ambiente (como por exemplo a percepção de um corredor) e saiba definir a melhor ação a ser tomada no momento. No modelo apresentado por Sales et al. (2010), o sistema proposto (*Neural Finite-State Automata* - Neuro-FSM) é caracterizado por apresentar uma estrutura composta de estados e ações que definem o plano de um trajeto, chamado de sequenciador. É a partir do sequenciador que é possível, diante de um contexto, tomar a melhor ação para uma determinada situação. As mudanças de estado/ação ocorrem a partir das entradas que o *Finite State Machine* (FSM) recebe do módulo neural, representadas pelas leituras do ambiente pelos sensores do robô. Apesar desta abordagem resultar em um sistema robusto, é relatado que esta abordagem não fornece uma estratégia para a exploração do ambiente e nem de planejamento de trajetórias dentro de plataformas de hardware complexos.

O algoritmo baseado em Distância Normalizada de Compressão (*Normalized Compression Distance* - NCD) (CILIBRASI; VITÁNYI, 2005) tem como objetivo principal encontrar relacionamentos entre dados, com intuito de identificar dados similares, além de detectar mudanças nos padrões de dados, como também o agrupamento de padrões similares para permitir o reconhecimento destes. A partir de amostras (variáveis, arquivos, etc), é aplicado um algoritmo de compressão que irá buscar a similaridade entre cada uma delas através das suas características dominantes, onde a distância entre uma amostra e outra é representado por um valor entre  $[0; 1 + \varepsilon]$ . Este número representa o grau de diferença entre duas amostras, onde  $\varepsilon$  é o limite superior para o erro do compressor usado (SANTOS, 2017).

A navegação com Reconhecimento de Estados baseado em ângulos de Referência em Santos (2017) foi aplicada para tratar estados que apresentavam bifurcações em diferentes angulações, reconhecidos através da leitura do sensor do robô. Nele, o algoritmo tem a finalidade de encontrar ângulos livres, ou seja, ângulos que não irão interceptar os obstáculos do ambiente. O sensor faz uma varredura dentro do seu intervalo de alcance, onde para cada ângulo, é calculada a distância entre o robô e o obstáculo, de modo que o ângulo de referência será o valor médio dos ângulos livres. Por final, a nova área livre é classificada como a mais próxima ao ângulo de referência.

## 3 Desenvolvimento

Neste capítulo, é descrito a forma metodológica de todo o processo que será realizado para garantir a navegação final do robô autônomo. A metodologia é dividida em quatro fases: (1) a coleta de dados de um ambiente virtual, (2) o agrupamento dos dados para rotular os estados a serem identificados para a análise, (3) a avaliação do método, utilizando ARI para comparação do agrupamento de dados com o seu ground-truth, (4) o uso dos dados agrupados para a navegação do robô propriamente dita.

### 3.1 Coleta de dados

Neste presente projeto, será utilizado o simulador aberto e multiplataforma de robótica *Webots*, desenvolvido pela empresa *Cyberbots*. Nele, será simulado diferentes ambientes virtuais do tipo labirinto, contendo em cada um destes, possíveis rotas alternativas.

Para que seja efetuado a coleta de dados destes ambientes, é necessário primeiramente, definir o robô a ser escolhido para atuar dentro do cenário. O modelo a ser escolhido será o *Pioneer3DX*, que além de ser um modelo clássico para ambientes *indoor*, foi utilizado em experimentos realizados por Santos (2017). Acoplado a ele, será utilizado um lidar de modelo *Sick LMS 291*, que é capaz de coletar dados de distância sob uma área que cobre até 180 graus. Através deste sensor, é possível coletar os dados a cada passo da simulação e armazená-los em um arquivo de extensão *.csv*, onde, posteriormente, serão usados na fase de agrupamento de dados.

Os dados armazenados pelo lidar representam a distância do robô ao obstáculo mais próximo, calculado por cada grau, em uma única posição. Desta forma, utilizando uma configuração de escaneamento de 180 graus de cobertura como exemplo, o lidar geraria um vetor de 180 pontos, que representaria um único estado. Cada estado representa uma cena que permite o robô a identificar o seu próximo movimento. Nos experimentos a serem realizados, serão analisados 7 tipos de estados: corredor, saída à direita, saída à esquerda, saída à esquerda e à direita, encruzilhada à direita, encruzilhada à esquerda e encruzilhada à direita e à esquerda.

### 3.2 Agrupamento de dados

A partir da coleta de dados efetuada na fase anterior, é possível obter um banco de dados para rotular os possíveis estados através de agrupamento de dados, utilizando o algoritmo *k-means*.

Antes da aplicação direta do algoritmo sobre os dados, será considerado a opção da

utilização de um normalizador de dados. O cálculo da distância em *k-means* pondera cada dimensão igualmente e, portanto, deve-se tomar cuidado para garantir que a unidade de dimensão não distorça a proximidade relativa dos pontos. É importante ressaltar que, durante a fase de testes, será realizado e comparado experimentos utilizando ou não o normalizador de dados, de forma a estabelecer, pela especificidade do mapa, qual irá obter melhores resultados.

Para agrupar os dados, será utilizado o algoritmo de *K-means* fornecido por Pedregosa et al. (2011), que aplica os conceitos descritos na Seção 2.1.1. Este algoritmo requer que o número de grupos a serem criados sejam especificados previamente. Dessa forma, para não cometer nenhum equívoco, é necessário verificar, no banco de dados, quantos tipos de estados diferentes poderão existir exatamente. É possível estimá-los a partir das informações de estados já contidos no banco de dados. Assim, o algoritmo tem a função de dividir um grupo de pontos em conjuntos disjuntos, cada um descrito pelos seus centroides.

É possível configurar diferentes parametrizações para a sua execução, sendo permitido a escolha de opções que melhor encaixam no banco de dados a ser trabalhado. As principais delas são: seleção do número de vezes que o algoritmo será executado, partindo de diferentes centroides iniciais, onde o resultado final será a execução com menor valor da SSE; seleção do número máximo de iterações que uma única execução é capaz de fazer; seleção de diferentes métodos de inicialização, que podem influenciar o lado de convergência.

### 3.2.1 Representação dos dados

Os bancos de dados a serem testados irão possuir 180 atributos, dados pela distância coletada por cada posição do sensor. Porém, fica inviável uma representação gráfica da distribuição destes dados em um plano com múltiplas dimensões. Esta fase do projeto é importante, pois a representação gráfica serve para a identificação de possíveis *outliers*, que seriam os pontos que representam valores muito distintos em relação aos outros pontos. A média é extremamente sensível aos extremos, portanto é necessário identificar dados que fogem da normalidade e que possam causar anomalias nos resultados obtidos.

Assim, será utilizado o PCA como método de redução de dimensionalidade. A ideia é transformar estes conjuntos de dados  $n$  dimensionais em conjuntos reduzidos, perdendo o mínimo de informações. Para cada coluna de dados que representava um eixo de coordenadas, terá agora o seu comprimento normalizado de acordo com um critério de escala, determinado pelo grau de variância. Dessa forma, o primeiro PC possuirá o maior grau de variância, e assim consecutivamente. Os PCs utilizados para a plotagem serão aqueles que irão conter a maior parte das informações. Assim, serão utilizados 3 PCs para uma representação gráfica 3-D.

É importante ressaltar que a plotagem sem a normalização de dados pode distinguir da plotagem com a normalização, assim como os resultados de validação. Qualquer motivo de resultados imprecisos poderá ser observado em ambas representações gráficas. Portanto, será

analisado, caso a caso, a especificidade de cada um.

### 3.3 Validação dos resultados

No final do processamento do *K-means*, será gerado um vetor de rótulos que representam os estados ao qual cada grupo de dados foi atribuído. Este vetor será usado para validação, a partir da utilização do método de ARI. Esta função permite avaliar a qualidade do agrupamento de dados a partir de sua comparação com o *GroundTruth*, como descrito na subseção 2.1.4. A pontuação é introduzida para determinar se os resultados de dois agrupamentos são semelhantes entre si. O valor pode variar até  $-1$  quando os agrupamentos são feitos de forma aleatória, e é igual a 1 quando o resultado dos agrupamentos são idênticos aos resultados do *GroundTruth*.

### 3.4 Navegação do robô

Nesta etapa, serão utilizados os dados agrupados durante a fase anterior, a fim de tornar a navegação do robô possível. O mapa utilizado para a coleta de dados durante o agrupamento de dados terá características semelhantes ao mapa que será utilizado para a navegação, para que seja avaliado de forma precisa a robustez e a capacidade de generalização deste método.

Uma simulação na navegação completa será desenvolvida, utilizando as configurações do *Pioneer3DX* para que ele possa partir de um local inicial até o local objetivo. Para que sejam efetuados os cálculos durante sua movimentação, será necessário o uso do lidar acoplado a ele, que terá o papel de capturar as informações dos locais recorrentes.



Figura 3.1 – Imagem Pioneer3DX com lidar Robotis Sick LMS 291. Fonte: elaborada pelo autor.

Através do sensor do robô, o algoritmo inicia o processo de coleta de sinais para a identificação dos estados, capturando o ponto referente à coordenada atual do robô. Depois, será calculada a distância deste ponto para cada um dos centroides estabelecidos durante a fase de agrupamento dos dados, de forma a encontrar o centroide mais próximo à ele. Assim, este ponto se assimilará às características do grupo de dados deste centroide. Ou seja, o estado deste ponto será baseado na distância do centroide mais próximo. Por fim, com base no estado identificado, o sistema deverá tomar a decisão de mover o robô.

## 4 Experimentos Computacionais

Neste capítulo serão apresentados os procedimentos usados para a geração da base de dados e experimentos realizados durante a condução da validação do agrupamento dos dados gerados, seguido do processo de navegação do robô. Na seção 4.2, será apresentado os resultados obtidos a partir dos experimentos criados. Toda a implementação poderá ser vista no *Github*.

### 4.1 Experimentos realizados

Para complementar os estudos experimentais com base na metodologia, foram criados quatro mapas do tipo labirinto, compostos por objetos do tipo parede, que representam os obstáculos. Cada mapa, além de serem labirintos diferentes, apresentam diferentes medidas de distâncias de larguras entre as paredes. O mapa 1, visualizado na Figura 4.1, por exemplo, possui largura de  $1,6m$  entre as paredes. O mapa 2 possui  $1,2m$ , o mapa 3 possui  $2,4m$  e o mapa 4 possui  $2m$ . Nestes três primeiros mapas, foram realizadas coletas de dados em seus diferentes locais, para que no final fosse possível realizar diversos tipos de experimentos, como por exemplo, a tentativa de navegação do robô utilizando a base de dados do próprio mapa em questão ou a utilização desta base de dados em outro mapa. Dessa forma, poderíamos verificar se este método teria eficácia quando aplicado em mapas diferentes com características parecidas. No quarto mapa não houve coleta de dados e, além de ter sido utilizado para os demais testes, foram efetuados testes nesta ambientação utilizando uma base de dados gerada pela junção dos dados dos três primeiros mapas, para que fosse avaliado o nível de desempenho em relação aos demais experimentos.

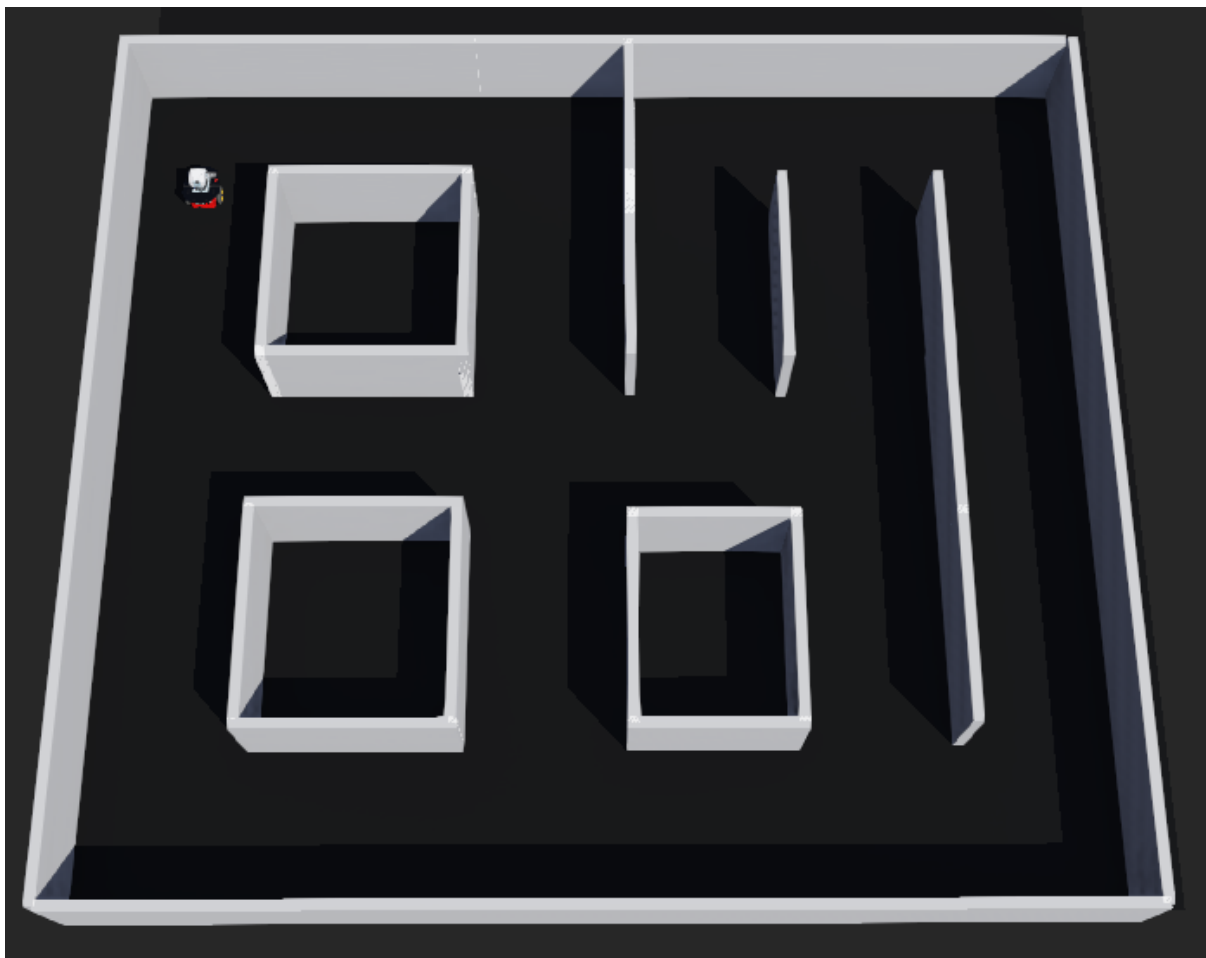


Figura 4.1 – Figura do mapa 1 com  $1,6m$  de largura entre as paredes. Fonte: elaborada pelo autor.



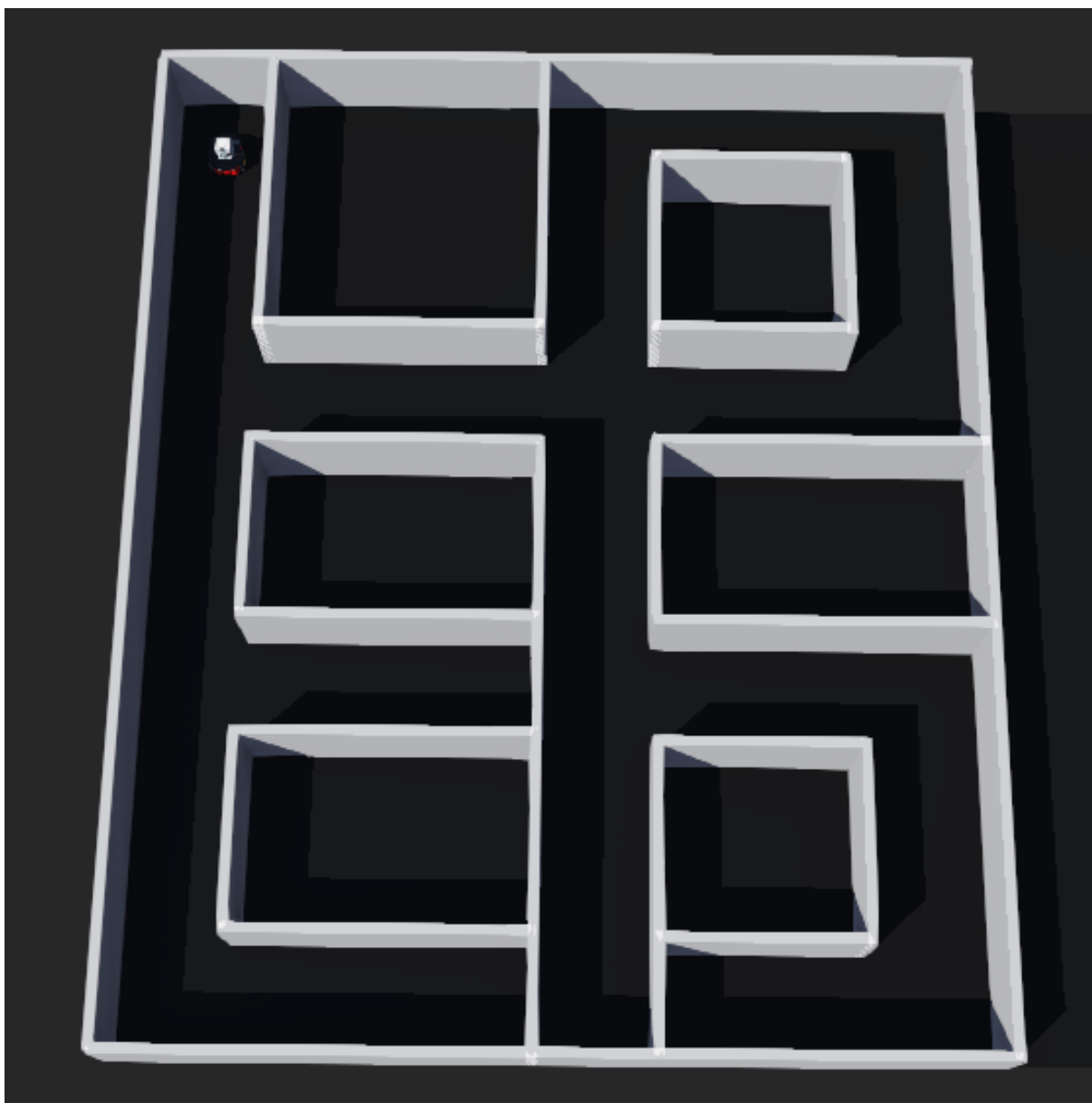


Figura 4.2 – Figura do mapa 2 com  $1,2m$  de largura entre as paredes. Fonte: elaborada pelo autor.

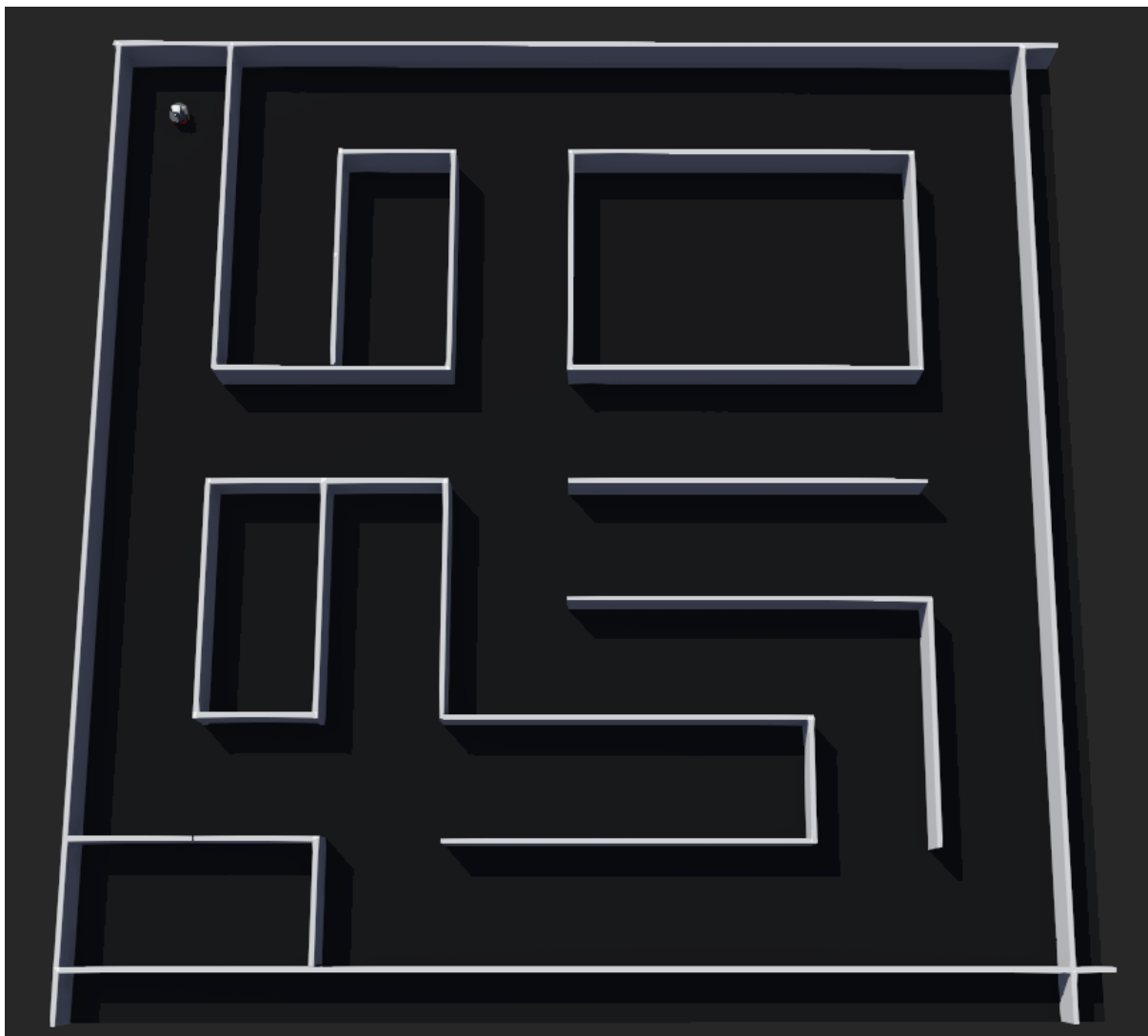


Figura 4.3 – Figura do mapa 3 com 2,4m de largura entre as paredes. Fonte: elaborada pelo autor.

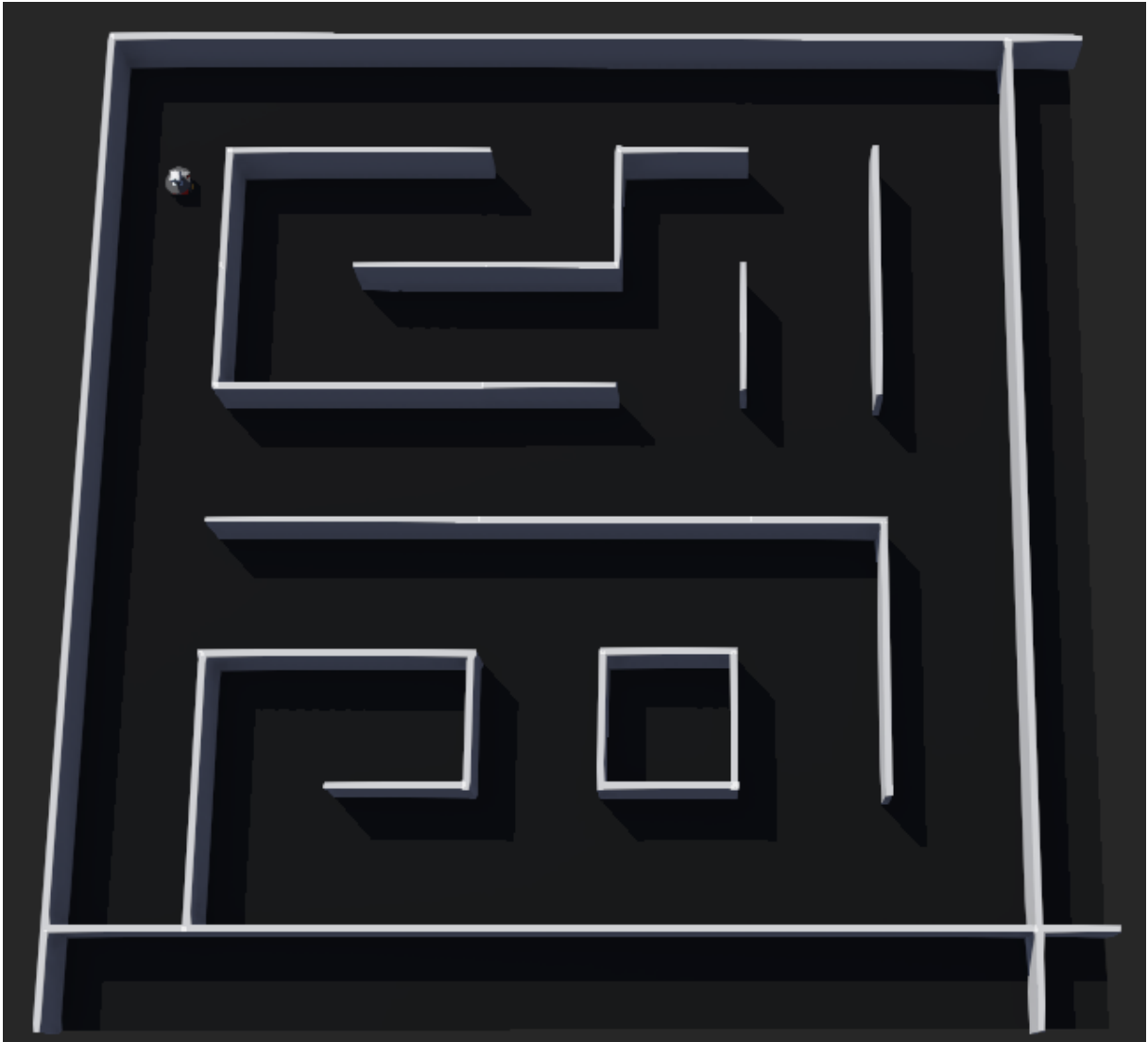


Figura 4.4 – Figura do mapa 4 com  $2m$  de largura entre as paredes. Fonte: elaborada pelo autor.

A coleta de dados foi efetuada durante a captura de dados do sensor de modelo *Sick LMS 291* acoplado ao *Pioneer3DX*, em tempo real, durante a simulação efetuada dentro da plataforma *Webots*. O robô foi colocado manualmente em diferentes locais do mapa para iniciar a coleta. Uma função básica de movimentação foi utilizada para que o robô fizesse a captura dos dados enquanto estivesse em movimento. A Figura 4.5 mostra como é executada a coleta de dados para armazenar informações correspondentes a um determinado estado. Durante a sua movimentação, foram armazenados a coleta de pontos que representam informações dos 180 ângulos da respectiva coordenada atual do robô. De forma análoga, cada ponto representa um tipo de estado coletado. Para armazená-los de forma organizada, inserimos pontos que representassem um determinado tipo de estado no mesmo arquivo de tipo *.csv*. Pontos que representam o estado de corredor, por exemplo, foram coletados em diferentes locais do labirinto. Dessa forma, foram gerados sete arquivos *.csv*, representando respectivamente todos os possíveis estados do ambiente. Esta

abordagem foi utilizada para que fosse permitida a rotulagem dos pontos, além de ser necessária para obter o *Ground Truth* para o experimento de validação dos resultados. Por fim, foi realizada a junção destes sete arquivos em um só.

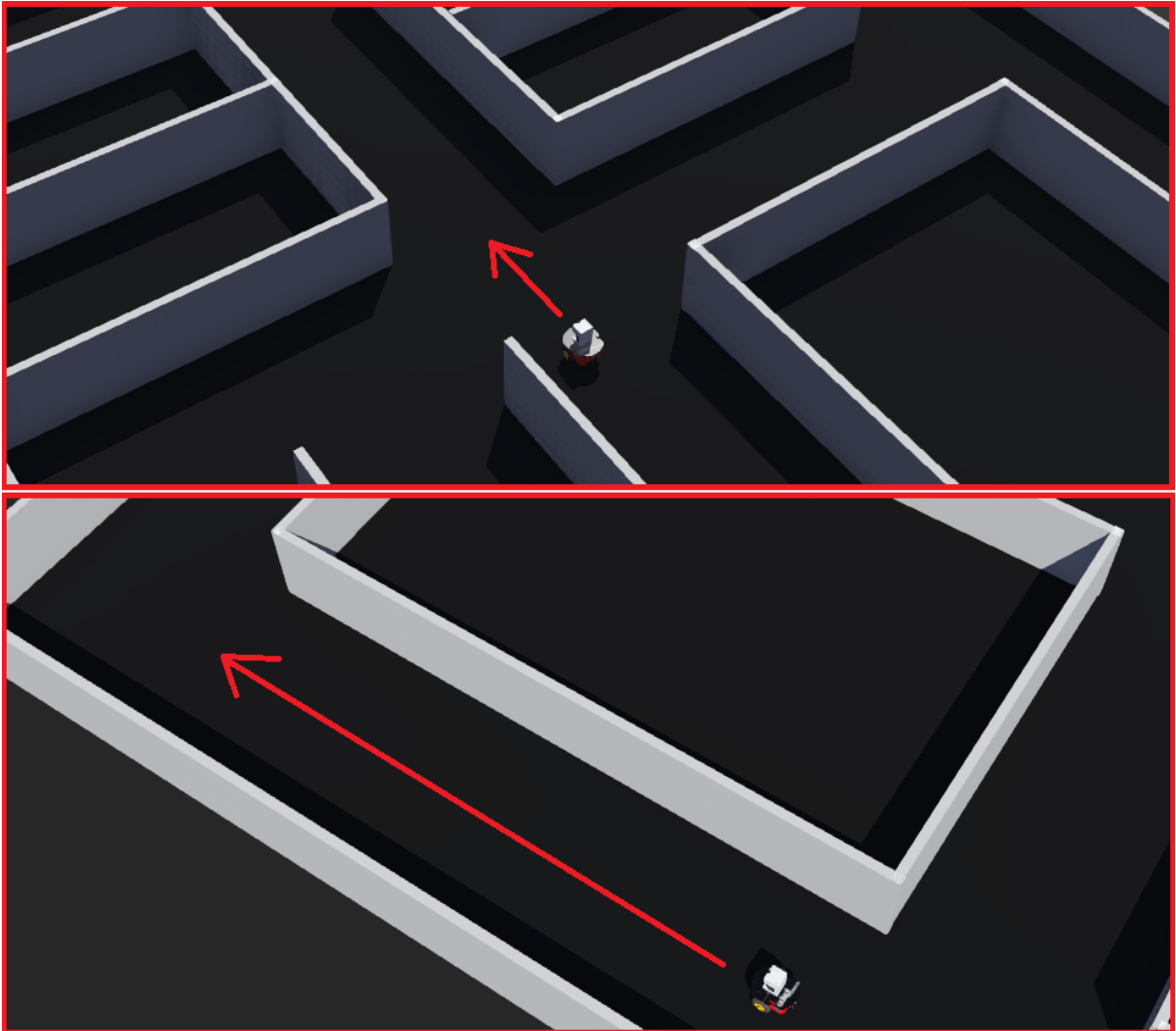


Figura 4.5 – Na figura de cima temos o percurso a ser realizado pelo robô, indicado pela seta, para realizar a captura de dados de estados referentes à encruzilhada. Abaixo temos o percurso a ser realizado para capturar estados do tipo corredor. Fonte: elaborada pelo autor.

Antes de utilizarmos esta base de dados para a navegação propriamente dita, foi necessário averiguarmos a qualidade do agrupamento do *k-means* a partir do resultado de validação pelo ARI. Foram executadas diferentes configurações de pré-processamento antes da execução do algoritmo *K-means*, sendo eles a inclusão ou não da normalização de dados e redução de dimensionalidade por PCA. A função de normalização utilizada neste processo foi fornecida por Pedregosa et al. (2011), podendo ser visto com maior detalhamento em *preprocessing-normalization*. Apesar da utilização de PCA ser bastante útil para agilização de cálculos, ele gerou, em todos os experimen-

tos, resultados piores do que a sua não utilização. Esta explicação se dá pela sua simplificação dos dados, de forma que, ao agrupá-los, foram gerados informações menos precisas quando validados pelo ARI. Logo, o uso de PCA foi direcionado apenas para estudo e análise durante a representação dos dados em 3-D ou 2-D, possibilitando a comparação de estados por observação gráfica. Para o agrupamento, foi utilizado o algoritmo de *K-means* fornecido por Pedregosa et al. (2011). Em sua parametrização, foi determinado, como principal argumento, o número de estados diferentes existentes na base de dados. O resultado de saída do agrupamento dos dados, junto com o seu *GroundTruth*, foram utilizados para a aplicação do ARI.

Após confirmarmos que o agrupamento realizado apresentou uma boa qualidade, foi possível darmos continuidade para o processo de simulação implementado no *Webots*. Dentro da implementação do controlador acoplado ao *Pioneer3DX*, inicialmente é realizado apenas uma vez a normalização de dados da base de dados coletada, seguida da execução do *K-means*, para que fossem gerados grupos de estados diferentes. Através da renderização do sensor, novos dados são coletados em tempo real a todo momento. Estes dados são normalizados e levados para uma função que irá calcular qual é o grupo de estados que cada ponto irá pertencer. Outra função irá buscar o estado recorrente mais frequente encontrado dentro destes novos dados gerados. A partir da identificação deste novo estado, ele será traduzido para a execução de uma nova ação que será realizada pelo robô. As ações incluídas neste projeto são compostas por funções básicas que executam três tipos de movimentos diferentes: navegação reta, onde o robô mantém a mesma velocidade nas duas rodas; curva à direita, onde a velocidade da roda direita é menor que a roda esquerda, possibilitando a execução de uma curva de 90° para direita; curva à esquerda, onde a velocidade da roda esquerda é menor que a roda direita, possibilitando uma curva de 90° para esquerda. Dessa forma, foi possível criarmos uma interação onde o robô constantemente capta informação em tempo real e reage conforme a identificação de estados diferentes ao previamente detectado. Na Figura 4.6 abaixo podemos observar a identificação dos estados em tempo real, durante o percurso realizado pelo robô.

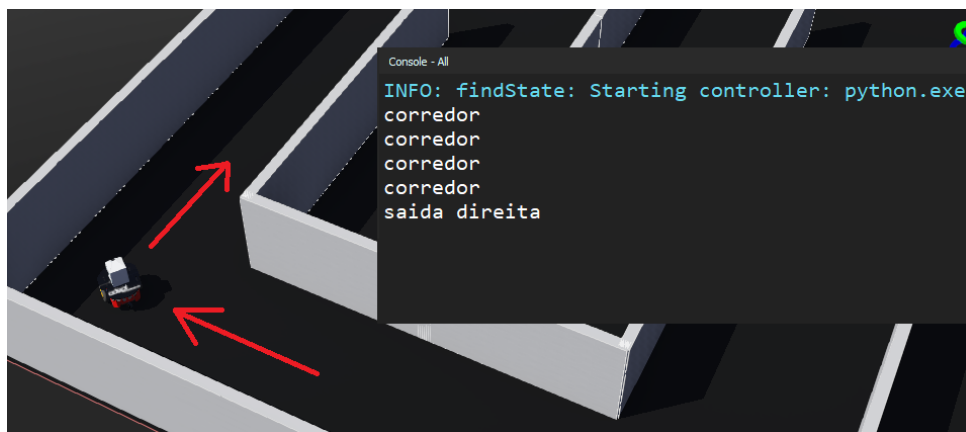


Figura 4.6 – Representação de identificação dos estados em tempo real durante a navegação do robô. Fonte: elaborada pelo autor.

## 4.2 Resultados

Neste capítulo são apresentados, interpretados e analisados todos os resultados alcançados no projeto. A Subseção 4.2.1 descreve os experimentos realizados durante a fase de agrupamento de dados, envolvendo testes que buscam atingir a melhor pontuação pelo método de ARI. A Subseção 4.2.2 utiliza as bases de dados para a simulação de navegação topológica do robô móvel.

### 4.2.1 Agrupamento de estados

Os processos realizados nesta seção, tem como função principal, relatar todos os experimentos executados referentes à qualidade do agrupamento de dados pelo *K-means* e, ao final, destacar aqueles que apresentarem a melhor condição para serem utilizados na fase de navegação do robô. Os resultados serão apresentados por meio do índice ARI. Como mencionado na seção 3.3, o melhor resultado é aquele que apresenta o valor de ARI mais próximo de 1. Vale a pena ressaltar que, em critério de desempate, o mais importante é priorizar os agrupamentos onde o número de grupos criados equivalem ou se aproximam do número de estados existentes no banco de dados.

Os experimentos dos dados coletados de cada mapa serão exibidos em várias composições, a fim de mostrar a qualidade do agrupamento no momento em que havia apenas dois tipos de estados existentes até sete, representando a base de dados final de um respectivo mapa. A Tabela 4.1 ilustra a composição dos experimentos, de forma que, o mais simples deles, por exemplo, possui apenas dois tipos de estados existentes: curva à direita ou curva à esquerda. O banco de dados de três estados, além de conter os dois estados anteriores, foi adicionado mais um estado referente à curva à direita e à esquerda. Dessa forma, o próximo banco de dados será constituído pelos estados anteriores e mais um novo estado. Assim, o último banco de dados é constituído de sete estados. Os experimentos foram realizados desta forma para que fosse possível detectar alguma particularidade entre um estado e outro que pudesse gerar um agrupamento ruim. As tabelas foram divididas, onde a primeira, foram expostas todos os resultados sem a normalização de dados, e a segunda, todos os resultados utilizando a normalização de dados.

Tabela 4.1 – Descrição dos estados utilizados em cada configuração de banco de dados aplicados no projeto.

Descrição do estado	Número de estados					
	2	3	4	5	6	7
Curva à direita	•	•	•	•	•	•
Curva à esquerda	•	•	•	•	•	•
Curva à direita e à esquerda		•	•	•	•	•
Encruzilhada à esquerda			•	•	•	•
Encruzilhada à direita				•	•	•
Encruzilhada					•	•
Corredor						•

Os experimentos apresentados nas Tabelas 4.2, 4.3 e 4.4 exibem os resultados obtidos dos seis bancos de dados diferentes, do mapa 1, mapa 2 e mapa 3, respectivamente, caracterizados pelo número de estados que cada um possui especificamente. Para complementar este experimento, foram avaliados agrupamentos com diferentes números de grupos criados, variando de dois até sete, a fim de detectar possíveis melhorias na qualidade do agrupamento. Os valores em destaques representam os agrupamentos de melhor qualidade pelo ARI.

Tabela 4.2 – Resultado de validação dos agrupamentos pelo ARI do **mapa 1**.

Graus	Número de grupos criados	Número de estados existentes					
		2	3	4	5	6	7
180° sem normalização de dados	2	<b>1.000</b>	0.752	0.504	0.422	0.342	0.301
	3	0.571	<b>1.000</b>	0.721	0.571	0.575	0.522
	4	0.333	0.714	<b>1.000</b>	0.814	0.714	0.717
	5	0.375	0.483	0.782	<b>1.000</b>	0.861	0.788
	6	0.333	0.478	0.684	0.823	<b>1.000</b>	0.949
	7	0.271	0.469	0.628	0.728	0.848	<b>0.996</b>
180° com normalização de dados	2	<b>1.000</b>	0.758	0.518	0.430	0.342	0.305
	3	0.571	<b>1.000</b>	0.874	0.738	0.590	0.533
	4	0.500	0.714	<b>1.000</b>	0.916	0.825	0.727
	5	0.375	0.615	0.782	<b>1.000</b>	0.937	0.870
	6	0.276	0.478	0.684	0.823	<b>1.000</b>	0.950
	7	0.276	0.363	0.631	0.730	0.851	<b>1.000</b>

Tabela 4.3 – Resultado de validação dos agrupamentos pelo ARI do **mapa 2**.

Graus	Número de grupos criados	Número de estados existentes					
		2	3	4	5	6	7
180° sem normalização de dados	2	<b>0.880</b>	0.750	0.501	0.441	0.398	0.389
	3	0.378	<b>0.940</b>	0.870	0.727	0.677	0.517
	4	0.482	0.585	<b>0.960</b>	0.871	0.818	0.714
	5	0.320	0.486	0.782	<b>0.943</b>	0.876	0.865
	6	0.276	0.477	0.577	0.823	<b>0.954</b>	0.901
	7	0.201	0.448	0.503	0.674	0.739	<b>0.771</b>
180° com normalização de dados	2	<b>1.000</b>	0.893	0.709	0.459	0.423	0.359
	3	0.444	<b>1.000</b>	0.945	0.810	0.699	0.540
	4	0.482	0.618	<b>1.000</b>	0.921	0.873	0.799
	5	0.320	0.486	0.673	<b>0.943</b>	0.876	0.906
	6	0.233	0.411	0.557	0.735	<b>0.954</b>	0.901
	7	0.245	0.368	0.558	0.624	0.777	<b>0.962</b>

Tabela 4.4 – Resultado de validação dos agrupamentos pelo ARI do **mapa 3**.

Graus	Número de grupos criados	Número de estados existentes					
		2	3	4	5	6	7
180° sem normalização de dados	2	<b>1.000</b>	0.754	0.508	0.423	0.340	0.296
	3	0.571	<b>0.806</b>	0.713	0.570	0.579	0.523
	4	0.416	0.517	<b>0.638</b>	0.544	0.706	0.680
	5	0.210	0.551	0.631	<b>0.727</b>	0.661	0.579
	6	0.263	0.357	0.635	0.700	<b>0.781</b>	0.729
	7	0.198	0.467	0.549	0.679	0.734	<b>0.803</b>
180° com normalização de dados	2	<b>1.000</b>	0.761	0.525	0.437	0.350	0.313
	3	0.571	<b>1.000</b>	0.876	0.744	0.599	0.541
	4	0.500	0.714	<b>1.000</b>	0.913	0.823	0.727
	5	0.375	0.615	0.782	<b>1.000</b>	0.935	0.868
	6	0.263	0.571	0.684	0.823	<b>1.000</b>	0.948
	7	0.276	0.467	0.626	0.725	0.809	<b>1.000</b>

Em geral, foi possível perceber que os agrupamentos de dados pelo algoritmo de *K-means* apresentaram bons valores pelo método de validação do ARI para os experimentos onde o número de estados existentes é equivalente ao número de grupos criados, demonstrando um bom sinal na qualidade das bases de dados. Isto foi possível devido ao processo sistemático da coleta de dados, que foi prosseguida de maneira a respeitar corretamente os espaços de coletas que correspondem a cada estado diferente. Na Figura 4.7 abaixo podemos observar a plotagem em três dimensões usando PCA, ilustrando a distribuição dos agrupamentos do mapa 3 com dados normalizados, separados pelas cores. Nesta base de dados em particular, foram coletados dois locais diferentes para cada estado, possuindo valor de 1 pelo ARI.



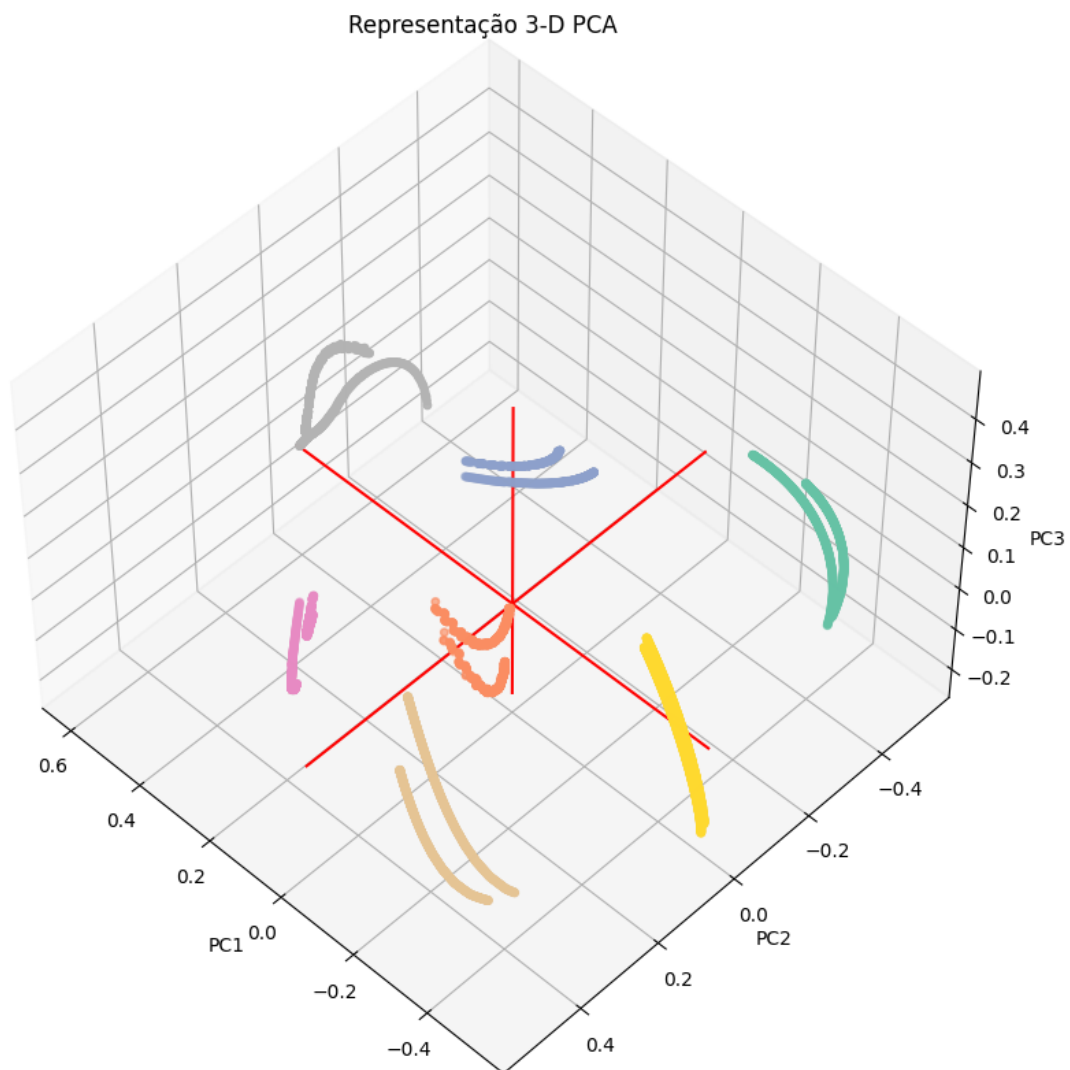


Figura 4.7 – Representação dos dados coletados do mapa 3 em PCA. Cada estado está sendo representado por uma cor diferente. Fonte: elaborada pelo autor.

Outra observação destacada foi que, os resultados utilizando a normalização dos dados em relação a sua não normalização, apresentaram resultados diferentes, especialmente no mapa 3, onde a distância de largura entre as paredes eram maiores. Considerando que o processo de coleta de dados foi o mesmo para todos os mapas, nota-se que o aumento da diferença dos resultados desta relação (dados normalizados versus dados não normalizados) foi consideravelmente maior em mapas mais abertos, como a do mapa 3 em questão. Neste sentido, observa-se a não decadência da qualidade dos agrupamentos quando se utiliza a normalização dos dados.

#### 4.2.2 Navegação do robô

Visto que as bases de dados geradas apresentaram boa qualidade, estes foram testados durante a simulação, inserindo-os no algoritmo de navegação descrito no final da seção 4.1.

Inicialmente foram executadas as simulações dentro de cada mapa usando a sua respectiva base de dados. Depois, foram experimentados navegação utilizando a base de dados de outros mapas (como por exemplo a navegação do mapa 3 com a base de dados do mapa 1). Nestes experimentos, de forma geral, o algoritmo possui uma ótima eficácia em detectar estados, obtendo uma acurácia de 100% para estados com saídas simples (como curva à direita, curva à esquerda e corredor) e 100% para estados com saídas compostas (como encruzilhada à direita, encruzilhada à esquerda, encruzilhada e saída a direita e esquerda) onde as proporções relativas das saídas não sejam extremamente discrepantes entre si. Por exemplo, ao passar por uma encruzilhada à esquerda, o robô identifica este estado corretamente caso o comprimento do percurso reto não seja inferior a aproximadamente 20% do percurso da curva à esquerda. Caso este caminho reto seja inferior a este valor, o robô identifica este estado como curva à esquerda, desconsiderando que existe um pequeno percurso reto. Na Figura 4.8 podemos observar a identificação correta do estado “encruzilhada à esquerda”, notando que a curva à esquerda neste cenário possui um comprimento de aproximadamente 34% do comprimento da saída reta.

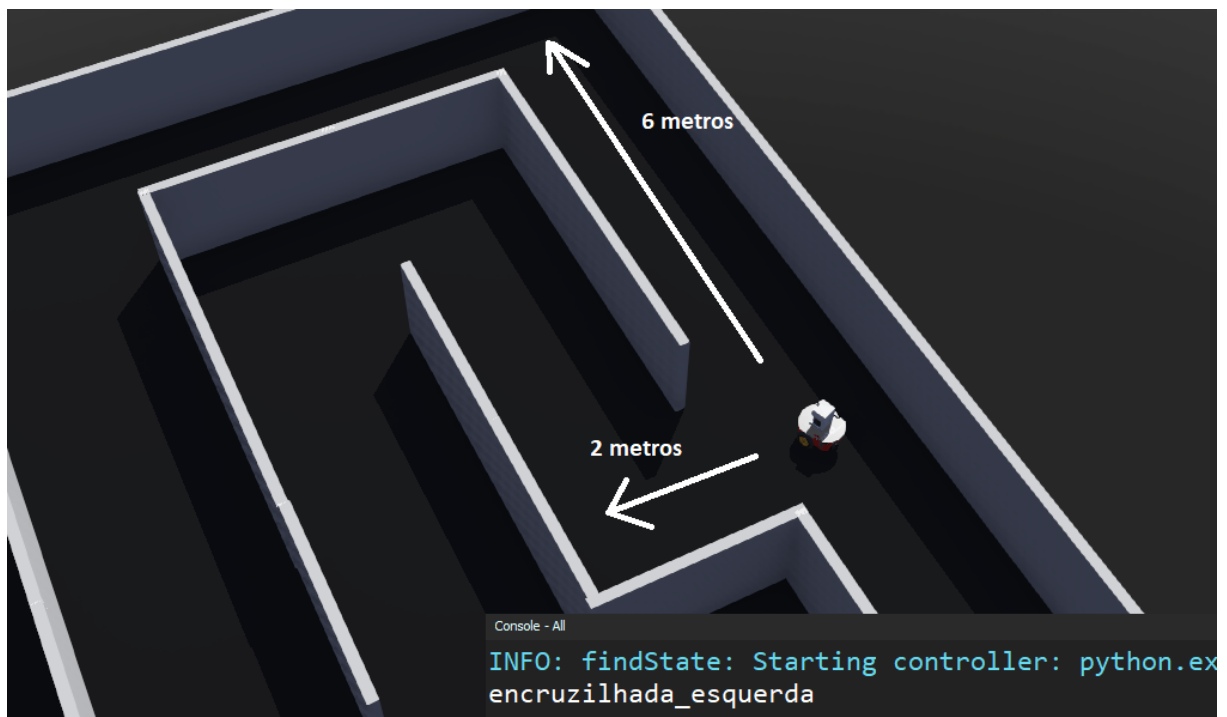


Figura 4.8 – Representação correta do estado de encruzilhada à esquerda. Fonte: elaborada pelo autor.

Já na Figura 4.9 notamos que houve uma identificação incorreta do estado, classificando-a como “curva à esquerda”. Neste cenário, notamos que o comprimento do percurso da saída reta é de aproximadamente 18% do comprimento do percurso da saída à esquerda. Logo, a possibilidade de seguir reto foi ignorada.

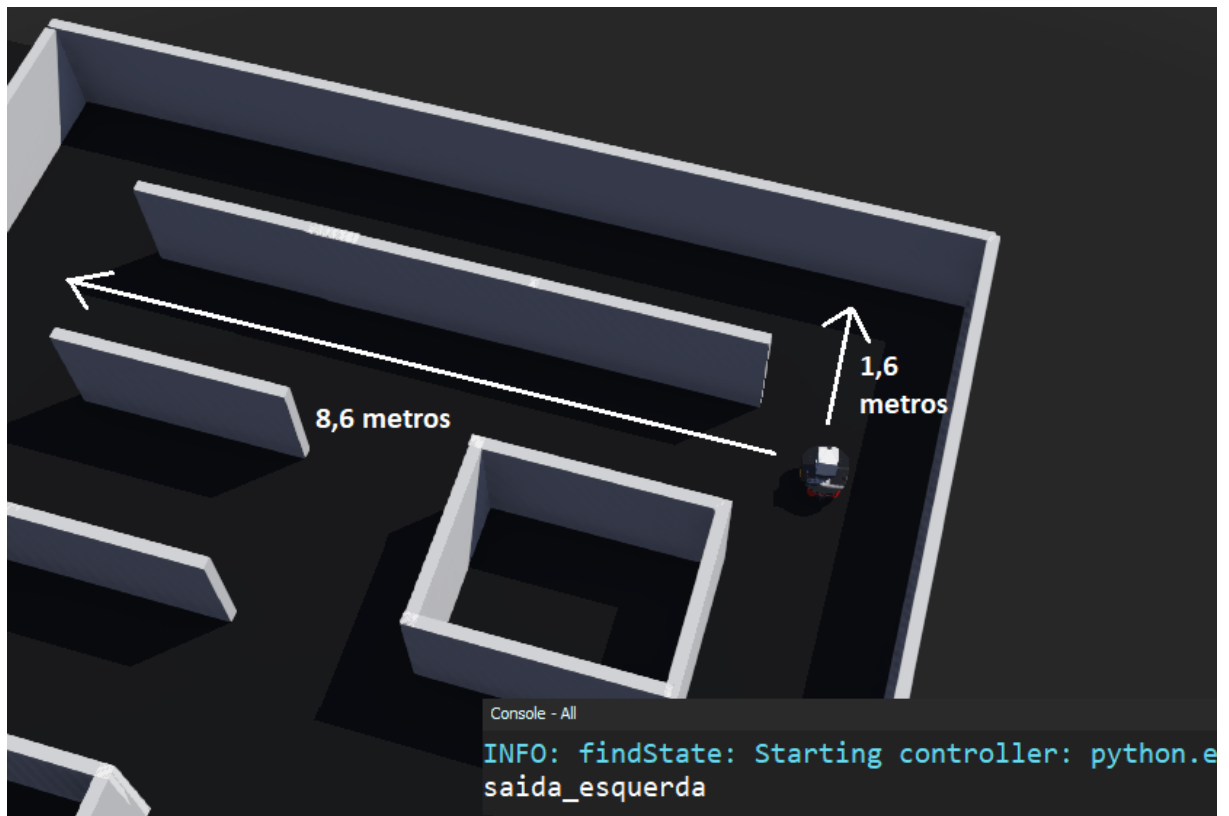


Figura 4.9 – Representação errada do estado encruzilhada à esquerda. Fonte: elaborada pelo autor.

Este tipo de limitação é característico do algoritmo de *K-means*, sendo possível observar que, mesmo as bases de dados sendo de boa qualidade, não foi possível cobrir os problemas gerados perante à existência de saídas compostas altamente desproporcionais. Na tentativa de solucionar este problema, acrescentamos na base de dados do mapa 1, mais coletas de dados referentes à locais onde estavam sendo identificados estados errôneos durante a simulação, como o da Figura 4.9. Ao verificar a qualidade desta nova base de dados, tivemos uma piora de 1 para 0.858 na qualidade do agrupamento pelo ARI. Ao colocar em prática este mesmo cenário, não foi possível obter a identificação correta do estado para este tipo de caso em particular. Contudo, ainda sim é possível obtermos identificação correta de estados para os demais casos.

Além dos experimentos até agora realizados, utilizamos o mapa 4, onde não foi feita nenhuma coleta de dados. Este mapa possui como características paredes de larguras de 2 metros. Ao utilizarmos as bases de dados já existentes, a acurácia dos resultados de identificação foram os mesmos que os previamente mencionados. Um outro teste em adicional foi acrescentado, onde fizemos a fusão das bases de dados do mapa 1, mapa 2 e mapa 3 a fim de testarmos uma simulação neste novo ambiente. A qualidade de agrupamento desta junção de dados pode ser observado na Tabela 4.5 a seguir:

Tabela 4.5 – Resultado de validação dos agrupamentos da junção do **mapa 1**, **mapa 2** e **mapa 3**.

Graus	Número de grupos criados	Número de estados existentes					
		2	3	4	5	6	7
180° sem normalização de dados	2	<b>1.000</b>	0.753	0.576	0.479	0.398	0.340
	3	0.476	<b>0.520</b>	0.911	0.756	0.585	0.577
	4	0.314	0.525	<b>0.560</b>	0.587	0.892	0.750
	5	0.034	0.227	0.601	<b>0.473</b>	0.509	0.623
	6	0.022	0.222	0.528	0.454	<b>0.448</b>	0.530
	7	0.017	0.241	0.277	0.378	0.444	<b>0.532</b>
180° com normalização de dados	2	<b>1.000</b>	0.750	0.507	0.471	0.388	0.336
	3	0.510	<b>1.000</b>	0.869	0.731	0.582	0.558
	4	0.495	0.670	<b>0.993</b>	0.913	0.821	0.635
	5	0.345	0.608	0.745	<b>0.988</b>	0.935	0.807
	6	0.240	0.546	0.608	0.767	<b>0.990</b>	0.948
	7	0.265	0.383	0.585	0.700	0.806	<b>0.992</b>

Nota-se que, a qualidade do agrupamento utilizando a normalização de dados é muito superior que a sua não utilização, onde na presença dos sete estados, o primeiro obteve um resultado de 0.992 e o segundo 0.532. Isto mostra que é imprescindível o uso da normalização de dados para se obter a melhor qualidade possível caso for necessário a junção de base de dados de diferentes mapas. A diferença da representação gráfica do *Ground Truth* entre a normalização e não normalização desta nova base de dados pode ser observado na Figura 4.10 abaixo:

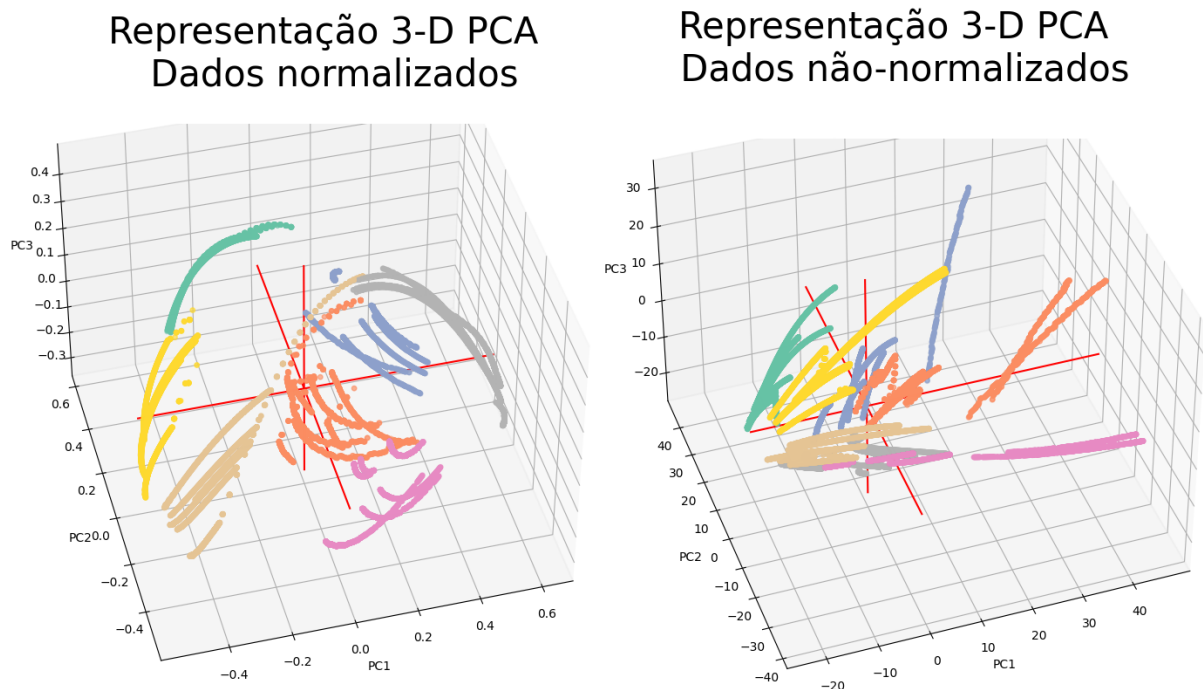


Figura 4.10 – Representação gráfica do *Ground Truth* da junção das bases de dados do mapa 1, mapa 2 e 3. Fonte: elaborada pelo autor.

Experimentos utilizando a normalização destes dados foram executados. Apesar de sua boa qualidade, ao usá-lo durante a simulação, foi possível percebermos o aumento do limiar que decide se uma saída composta é discrepante em relação à outra, sendo um ponto negativo pelo fato de aumentar a frequência do robô em ignorar saídas de menores tamanhos dentro das identificações de saídas compostas. Contudo, para saídas simples, o robô ainda permaneceu identificando os estados com uma acurácia de 100%.

Comparado à utilização de base de dados de mapas individuais em relação a junção da base de dados de dois ou mais mapas, foram observados melhores desempenhos da primeira quando se trata de uma identificação de estados mais assertiva para saídas compostas. Dessa forma, um último experimento foi executado, usando a base de dados do mapa 1 para a realização de uma navegação com rotas pré-definidas dentro do mapa 4. Foi usado como entrada adicional ao algoritmo, um vetor de dados que contém informações dos melhores caminhos para se chegar a um ponto final, partindo de um ponto inicial. Este vetor representa, na teoria, o percurso mais curto dentre as diversas existentes, que um algoritmo detector de melhor rota poderia gerar. A Figura 4.11 ilustra, através das setas vermelhas, o que seriam percursos escolhidos por este tipo de algoritmo, onde o círculo verde representa uma posição inicial do robô e o círculo vermelho uma posição final que será alcançado.

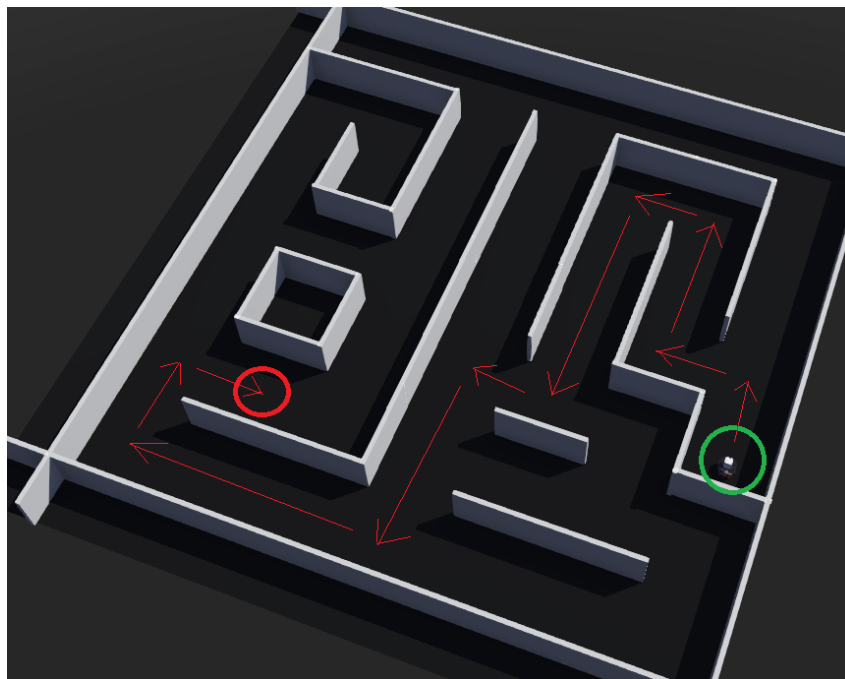


Figura 4.11 – Melhor percurso sugerido para alcançar um ponto final, partindo do ponto inicial

Fonte: elaborada pelo autor.

Sendo assim, com a utilização deste vetor representando os caminhos que devem ser percorridos, o algoritmo de navegação foi alterado para interpretar a sua leitura e seguir o percurso sugerido, atrelando-o à identificação de estados. Para a visualização desta navegação

em específico, onde foi utilizado rotas pré-definidas, pode ser visualizado no link <<https://drive.google.com/file/d/12I6oM7PKFOXc0sULWxcLoGAuNOhFfNan/view?usp=sharing>>.

# 5 Considerações Finais

Neste capítulo, são explicitadas as considerações finais do trabalho. Na seção 5.1, foi recapitulada a conclusão de cada um dos objetivos específicos. A seção 5.2 discorre sobre possíveis trabalhos futuros que podem ser adicionados à este, de forma a acrescentar mais estudos ao projeto.

## 5.1 Conclusão

Conforme apresentado, o objetivo geral deste trabalho consiste no desenvolvimento de um sistema inteligente que seja capaz de executar uma navegação robótica a partir de agrupamento de dados gerados pelo algoritmo *K-means*, utilizando informações de um mapa topológico. Após a revisão de literatura, pôde ser observado que o agrupamento de dados para o contexto de reconhecimento de estados durante a navegação topológica é uma abordagem diferente das demais, por ser um método que demonstra originalidade perante à tomada de decisão do robô autônomo.

A partir da metodologia aplicada, foi possível gerar bases de dados que representassem os respectivos estados das ambientações. Utilizando o método proposto, nota-se que, pelo agrupamento destes estados pelo método de *K-means*, foi obtido bons resultados partindo da configuração de agrupamento com a normalização de dados, podendo observar uma boa qualidade no que se refere à uma coleta de dados bem executada.

Os experimentos realizados dentro do simulador robótico realístico demonstraram uma acurácia de 100% para identificação de estados em saídas simples e 100% para saídas compostas onde o percurso de uma das saídas seja de pelo menos aproximadamente 20% do tamanho da outra. Caso não seja atendido esta condição, o robô ignora a saída de menor tamanho e, conseqüentemente, identifica o estado erroneamente. Além disso, entre os experimentos utilizando bases de dados de mapas individuais em comparação com a junção dos demais mapas, o primeiro entra em destaque, possuindo uma melhor liminar que atenda a condição de identificação de estados corretos com maiores frequências de sucesso para saídas compostas.

O experimento utilizando rotas pré-definidas revela que é possível construir um sistema de navegação com identificação de estados para se atingir um destino final. Dado uma coordenada inicial, junto com um conjunto de dados que descreve o percurso mais curto para o destino, o robô fez a leitura dos dados de forma correta e chegou no destino final com êxito. Neste contexto, nota-se que, a identificação de estados atrelados à pré-definição de rotas para os tipos de mapas estudados, é um método que pode trazer eficiência no processo de navegação robótica.

De forma geral, conclui-se que é possível utilizarmos estes dados para a identificação

dos estados. Além disso, esta abordagem promove uma metodologia onde não é requerido um processamento operacional custoso.

## 5.2 Trabalhos Futuros

Nesta seção, é discutido possíveis trabalhos futuros que podem ser implementados ao projeto, a fim de prover melhorias e levar este projeto adiante. Desta forma, as atividades seriam: (1) adicionar ruídos dentro dos mapas e verificar o comportamento do robô perante às identificações de estados em momentos onde o sensor capta estes objetos ruidosos, em tempo real. Os ruídos podem ser representados como objetos de diversas formas, espalhadas pelo mapa, mantendo o critério de uma navegação sem colisão; (2) navegação robótica dentro de mapas onde as distâncias das larguras entre as paredes variam em diferentes locais da ambientação; (3) melhoria na complexidade das funções que executam as ações básicas do robô, de modo que ele consiga sempre se manter distante de possíveis colisões, sejam estas paredes ou ruídos.



# Referências

- BAHNSEN, A. C.; TORROLEDO, I. *Machine Learning Algorithms Explained – K-Means Clustering*. 2017. <<https://blog.easysol.net/machine-learning-algorithms-3/>>.
- CARVALHO, M. de. *Construindo o saber: técnicas de metodologia científica*. [S.l.]: Papyrus Editora, 1989. ISBN 9788530800710.
- CILIBRASI, R.; VITÁNYI, P. M. Clustering by compression. *IEEE Transactions on Information theory*, IEEE, v. 51, n. 4, p. 1523–1545, 2005.
- FISHER, R. *Iris Data Set*. 1936. <https://archive.ics.uci.edu/ml/datasets/iris> [Acessado em 10 de Abril de 2020]. Disponível em: <<https://archive.ics.uci.edu/ml/datasets/iris>>.
- HEAP, B.; PAGNUCCO, M. Analysis of cluster formation techniques for multi-robot task allocation using sequential single-cluster auctions. In: SPRINGER. *Australasian Joint Conference on Artificial Intelligence*. [S.l.], 2012. p. 839–850.
- HUBERT, L.; ARABIE, P. Comparing partitions. *Journal of classification*, Springer, v. 2, n. 1, p. 193–218, 1985.
- JOLLIFFE, I. Principal component analysis. In: \_\_\_\_\_. *International Encyclopedia of Statistical Science*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. p. 1094–1096. ISBN 978-3-642-04898-2. Disponível em: <[https://doi.org/10.1007/978-3-642-04898-2\\_455](https://doi.org/10.1007/978-3-642-04898-2_455)>.
- NGUYEN, V. A.; STARZYK, J. A.; GOH, W.-B. A spatio-temporal long-term memory approach for visual place recognition in mobile robotic navigation. *Robotics and Autonomous Systems*, Elsevier, v. 61, n. 12, p. 1744–1758, 2013.
- OTT, L.; RAMOS, F. Real-time clustering for long-term autonomy. In: SPRINGER. *Experimental Robotics*. [S.l.], 2013. p. 129–141.
- PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011.
- RAMPAZZO, L. *Metodologia científica*. [S.l.]: Edições Loyola, 2005. ISBN 9788515024988.
- RAVANKAR, A. A.; HOSHINO, Y.; EMARU, T.; KOBAYASHI, Y. Robot mapping using k-means clustering of laser range sensor data. *Bulletin of Networking, Computing, Systems, and Software*, v. 1, n. 1, p. pp–9, 2012.
- SAHA, O.; DASGUPTA, P. Real-time robot path planning around complex obstacle patterns through learning and transferring options. In: IEEE. *2017 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. [S.l.], 2017. p. 278–283.
- SAHA, O.; DASGUPTA, P. Efficient real-time robot navigation using incremental state discovery via clustering. In: *The Thirty-First International Flairs Conference*. [S.l.: s.n.], 2018.

- SALES, D. O.; SHINZATO, P.; PESSIN, G.; WOLF, D. F.; OSORIO, F. S. Vision-based autonomous navigation system using ann and fsm control. In: IEEE. *2010 Latin American Robotics Symposium and Intelligent Robotics Meeting*. [S.l.], 2010. p. 85–90.
- SANTOS, J. M.; EMBRECHTS, M. On the use of the adjusted rand index as a metric for evaluating supervised classification. In: SPRINGER. *International conference on artificial neural networks*. [S.l.], 2009. p. 175–184.
- SANTOS, V. d. C. *Uma abordagem híbrida para planejamento exploratório de trajetórias e controle de navegação de robôs móveis autônomos*. Tese (Doutorado) — Universidade de São Paulo, 2017.
- SANTOS, V. de C.; SALES, D. O.; TOLEDO, C. F. M.; OSÓRIO, F. S. A hybrid ga-ann approach for autonomous robots topological navigation. In: *Proceedings of the 29th Annual ACM Symposium on Applied Computing*. [S.l.: s.n.], 2014a. p. 148–153.
- SANTOS, V. de C.; TOLEDO, C. F. M.; OSÓRIO, F. S. A hybrid approach for path planning and execution for autonomous mobile robots. In: IEEE. *2014 Joint Conference on Robotics: SBR-LARS Robotics Symposium and Robocontrol*. [S.l.], 2014b. p. 124–129.
- SIEGWART, R.; NOURBAKHSI, I. R.; SCARAMUZZA, D. *Introduction to autonomous mobile robots*. [S.l.]: MIT press, 2011.
- SOARES, A.; SANTOS, V.; TOLEDO, C.; OSÓRIO, F.; DELBEM, A. Nd-ncd: Environmental characteristics recognition and novelty detection for mobile robots control and navigation. In: *Robotics*. [S.l.]: Springer, 2016. p. 192–209.
- TAN, P.-N.; STEINBACH, M.; KUMAR, V. *Introduction to data mining*. [S.l.]: Pearson Education India, 2016.