MINISTÉRIO DA CIÊNCIA, TECNOLOGIA, INOVAÇÕES E COMUNICAÇÕES
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m21c/2019/08.07.15.41-TDI

ESTUDO DE PROCESSOS DE AMOSTRAGEM NÃO UNIFORME/INFORMADA EM ÁRVORES ALEATÓRIAS DE EXPLORAÇÃO RÁPIDA VISANDO O PLANEJAMENTO AUTOMÁTICO DE ROTAS PARA VEÍCULOS AÉREOS NÃO TRIPULADOS

Luiz Gustavo Diniz de Oliveira Véras

Tese de Doutorado do Curso de Pós-Graduação em Computação Aplicada, orientada pelos Drs. Lamartine Nogueira Frutuoso Guimarães, e Felipe Leonardo Lôbo Medeiros, aprovada em 09 de agosto de 2019.

URL do documento original:
<http://urlib.net/8JMKD3MGP3W34R/3TQ87S5>

INPE
São José dos Campos
2019

PUBLICADO POR:

Instituto Nacional de Pesquisas Espaciais - INPE
Gabinete do Diretor (GBDIR)
Serviço de Informação e Documentação (SESID)
CEP 12.227-010
São José dos Campos - SP - Brasil
Tel.:(012) 3208-6923/7348
E-mail: pubtc@inpe.br

CONSELHO DE EDITORAÇÃO E PRESERVAÇÃO DA PRODUÇÃO INTELECTUAL DO INPE - CEPPII (PORTARIA Nº 176/2018/SEI-INPE):

Presidente:

Dra. Marley Cavalcante de Lima Moscati - Centro de Previsão de Tempo e Estudos Climáticos (CGCPT)

Membros:

Dra. Carina Barros Mello - Coordenação de Laboratórios Associados (COCTE)

Dr. Alisson Dal Lago - Coordenação-Geral de Ciências Espaciais e Atmosféricas (CGCEA)

Dr. Evandro Albiach Branco - Centro de Ciência do Sistema Terrestre (COCST)

Dr. Evandro Marconi Rocco - Coordenação-Geral de Engenharia e Tecnologia Espacial (CGETE)

Dr. Hermann Johann Heinrich Kux - Coordenação-Geral de Observação da Terra (CGOBT)

Dra. Ieda Del Arco Sanches - Conselho de Pós-Graduação - (CPG)

Silvia Castro Marcelino - Serviço de Informação e Documentação (SESID)

BIBLIOTECA DIGITAL:

Dr. Gerald Jean Francis Banon

Clayton Martins Pereira - Serviço de Informação e Documentação (SESID)

REVISÃO E NORMALIZAÇÃO DOCUMENTÁRIA:

Simone Angélica Del Ducca Barbedo - Serviço de Informação e Documentação (SESID)

André Luis Dias Fernandes - Serviço de Informação e Documentação (SESID)

EDITORAÇÃO ELETRÔNICA:

Ivone Martins - Serviço de Informação e Documentação (SESID)

Cauê Silva Fróes - Serviço de Informação e Documentação (SESID)

MINISTÉRIO DA CIÊNCIA, TECNOLOGIA, INOVAÇÕES E COMUNICAÇÕES
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

sid.inpe.br/mtc-m21c/2019/08.07.15.41-TDI

ESTUDO DE PROCESSOS DE AMOSTRAGEM NÃO UNIFORME/INFORMADA EM ÁRVORES ALEATÓRIAS DE EXPLORAÇÃO RÁPIDA VISANDO O PLANEJAMENTO AUTOMÁTICO DE ROTAS PARA VEÍCULOS AÉREOS NÃO TRIPULADOS

Luiz Gustavo Diniz de Oliveira Véras

Tese de Doutorado do Curso de Pós-Graduação em Computação Aplicada, orientada pelos Drs. Lamartine Nogueira Frutuoso Guimarães, e Felipe Leonardo Lôbo Medeiros, aprovada em 09 de agosto de 2019.

URL do documento original:
<http://urlib.net/8JMKD3MGP3W34R/3TQ87S5>

INPE
São José dos Campos
2019

Dados Internacionais de Catalogação na Publicação (CIP)

Véras, Luiz Gustavo Diniz de Oliveira.

V581e Estudo de processos de amostragem não uniforme/informada em árvores aleatórias de exploração rápida visando o planejamento automático de rotas para veículos aéreos não tripulados / Luiz Gustavo Diniz de Oliveira Véras. – São José dos Campos : INPE, 2019.

xxviii + 239 p. ; (sid.inpe.br/mtc-m21c/2019/08.07.15.41-TDI)

Tese (Doutorado em Computação Aplicada) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2019.

Orientadores : Drs. Lamartine Nogueira Frutuoso Guimarães, e Felipe Leonardo Lôbo Medeiros.

1. Planejamento de rota. 2. RRT. 3. Grades de Sukharev. 4. Vértices Convexos. 5. Amostragem não uniforme/informada. I.Título.

CDU 519.243:629.014.8

Esta obra foi licenciada sob uma Licença Creative Commons Atribuição-NãoComercial 3.0 Não Adaptada.

This work is licensed under a Creative Commons Attribution-NonCommercial 3.0 Unported License.

Aluno (a): ***Luiz Gustavo Diniz de Oliveira Véras***

Título: **"ESTUDO DE PROCESSOS DE AMOSTRAGEM NÃO UNIFORME/INFORMADA EM ÁRVORES ALEATÓRIAS DE EXPLORAÇÃO RÁPIDAVISANDO O PLANEJAMENTO AUTOMÁTICO DE ROTAS PARA VEÍCULOS AÉREOS NÃO TRIPULADOS"**

Aprovado (a) pela Banca Examinadora em cumprimento ao requisito exigido para obtenção do Título de *Doutor(a)* em

Computação Aplicada

Dr. Thales Sehn Körting

Presidente / INPE / São José dos Campos - SP

( ) Participação por Video - Conferência

(X) Aprovado ( ) Reprovado

Dr. Lamartine Nogueira Frutuoso Guimarães

Orientador(a) / INPE / SJCampos - SP

( ) Participação por Video - Conferência

(X) Aprovado ( ) Reprovado

Dr. Felipe Leonardo Lôbo Medeiros

Orientadpr(a) / CTA / São José dos Campos - SP

( ) Participação por Video - Conferência

(X) Aprovado ( ) Reprovado

Dr. Solon Venâncio de Carvalho

Membro da Banca / INPE / SJCampos - SP

( ) Participação por Video - Conferência

(X) Aprovado ( ) Reprovado

Este trabalho foi aprovado por:

( ) maioria simples

(X) unanimidade

São José dos Campos, 09 de agosto de 2019

À minha sobrinha ***Maria Eduarda***

AGRADECIMENTOS

Agradeço primeiramente a meus pais, Cláudia Maria e Antônio José, e a meu irmão, Thiago, pelo apoio e compreensão pela minha frequente ausência durante esses anos de dedicação ao doutorado.

Agradeço ao meu orientador e amigo Dr. Felipe Medeiros, por ter me iniciado neste tema tão interessante que é a navegação autônoma de VANTs, e pela sempre imediata disposição em me ajudar nas mais diversas dúvidas e problemas que surgiram durante meu doutoramento.

Agradeço ao professor Dr. Lamartine Guimarães, pelo apoio e suporte oferecido para que fosse possível a execução deste trabalho.

Agradeço ao professor Dr. Élcio Shiguemori, que me concedeu a oportunidade de atuar como docente na Universidade Paulista (UNIP), onde vivi um período de grande aprendizado e experiência.

Agradeço aos meus amigos(as) Anna Karina, Sóstenes, Rudinei, Luis Fernando, Vitor Conrado e Marquinhos, pelo apoio, suporte e bons momentos durante meu período em São José dos Campos.

Agradeço à minha namorada Daniele Justino, pelo apoio incondicional e paciência demonstrada frente à minha necessidade de dedicação às minhas atividades de doutorado.

Agradeço ao Instituto Federal de São Paulo (IFSP), especialmente ao campus Bragança Paulista, pelo apoio e permissão de licença para qualificação que me foi aprovada. Sem isso, a execução deste trabalho, com toda a certeza, teria sido muito mais árdua.

Agradeço ao INPE por permitir o uso de suas instalações e infraestrutura para acesso à artigos acadêmicos, livros e materiais de pesquisa consultados durante o desenvolvimento deste trabalho.

Agradeço à secretaria de pós graduação do INPE (SPG), pelo auxílio na preparação de documentações e comprovantes solicitados durante meu período como aluno de pós graduação.

Agradeço à secretaria do curso de pós graduação em Computação Aplicada do INPE,

especialmente à secretária Jéssica, pelos bons serviços prestados e apoio nos mais diversos processos burocráticos.

Agradeço ao corpo docente do curso de pós graduação em Computação Aplicada do INPE pelos ensinamentos ao longo das disciplinas cursadas.

Agradeço ao Instituto de Estudos Avançados (IEAv), em especial à divisão C4ISR, pelo permissão de acesso às instalações do instituto para o desenvolvimento deste trabalho.

Agradeço à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pela bolsa de fomento disponibilizada a mim para a execução deste projeto de doutorado.

RESUMO

A etapa de planejamento de rota na navegação de um Veículo Aéreo Não Tripulado (VANT) garante que uma rota cinematicamente e dinamicamente viável e sem colisão seja planejada entre um ponto inicial e um ponto final em um ambiente de navegação. Um dos algoritmos mais utilizados nessa etapa é o Árvore Aleatória de Exploração Rápida (RRT, do inglês *Rapidly-exploring Random Tree*), o qual constrói iterativamente uma estrutura de dados em árvore onde cada um de seus nós é coletado aleatoriamente como uma amostra do ambiente de navegação até que os pontos de navegação inicial e final sejam conectados através deles. O algoritmo RRT* (lê-se RRT estrela) é uma variação do algoritmo RRT que garante o planejamento da rota com o menor comprimento para o VANT baseado em realocação de arestas de nós vizinhos, porém, devido a essa nova operação, exige um alto custo computacional. Alguns autores afirmam que ao direcionar a coleta de novas amostras para regiões específicas no ambiente de navegação, seria possível melhorar o tempo de planejamento desse algoritmo. Processos que utilizam esse tipo de abordagem são conhecidos como amostragem não uniforme/informada. Nesta tese, é apresentado um estudo do uso de dois processos de amostragem não uniforme/informada no algoritmo RRT*, um baseado em grade de Sukharev (um tipo de grade que gera amostras com dispersão ótima) e outro nos vértices convexos das envoltórias de segurança dos obstáculos. A influência de cada um dos processos considerados no tempo de planejamento e comprimento das rotas planejadas pelo algoritmo RRT* é estudada. Com base nessas verificações, é apresentado um novo algoritmo baseado no RRT*, denominado RRT* Sukharev Vértices (RRT*-SV), no qual são utilizados ambos os processos estudados neste trabalho. Testes computacionais foram realizados para verificar se o algoritmo RRT*-SV apresenta menor tempo de planejamento do que o algoritmo RRT* e outros algoritmos da literatura como, por exemplo, os algoritmos RRT*-Smart, o Informed-RRT* e o BIT* (do inglês *Batch Informed Trees Star*), que também utilizam processos de amostragem não uniforme/informada. Os testes computacionais foram realizados considerando diversos ambientes de navegação com representação contínua e bidimensional e com diferentes quantidades e distribuições espaciais de obstáculos estáticos poligonais. Os resultados indicam que o processo de amostragem proposto acelera o tempo de planejamento do RRT* e demonstram que o algoritmo RRT*-SV possui melhor desempenho em vários tipos de ambientes de navegação quando comparado aos outros algoritmos considerados. Pelo desempenho demonstrado, o algoritmo RRT*-SV se mostra adequado para uso em aplicações de navegação em tempo real de VANTs que tenham como restrição o planejamento da rota de menor comprimento.

Palavras-chave: Planejamento de rota. RRT. Grades de Sukharev. Vértices Convexos. Amostragem não uniforme/informada. VANT. Revisão Sistemática da Literatura.

STUDY OF NON-UNIFORM/INFORMED SAMPLING PROCESSES IN RAPIDLY-EXPLORING RANDOM TREES FOR THE AUTOMATIC PATH PLANNING FOR UNMANNED AERIAL VEHICLES

ABSTRACT

The path planning step in the navigation of an Unmanned Aerial Vehicle (UAV) ensures that a kinodynamically viable, collision-free path between a start point and an end point in a navigation environment can be planned. One of the most commonly used algorithms for path planning is the Rapidly-exploring Random Tree (RRT), which iteratively constructs a tree data structure where each of its nodes is randomly collected as a sample of the navigation environment until the start and end navigation points are connected through them. The RRT* algorithm (read RRT star) is a variation of the RRT algorithm that ensures the planning of the path with the shortest length for the UAV based on edge relocation of neighboring node edges. However, due to this new operation requires, it require a high computational cost. Some authors affirm that by biasing the collection of new samples to specific regions in the navigation environment, it would be possible to improve the planning time of this algorithm. Processes using this type of approach are known as non-uniform/informed sampling. In this thesis, a study of the use of two non-uniform/informed sampling processes in the RRT* algorithm, one based on Sukharev grid (one type of grid that generates optimally dispersed samples) and the other on the convex vertices of safety envelopes of the obstacles, is carried out together with the RRT* algorithm. The influence of each of the considered processes on the planning time and length of the path planned by the RRT* algorithm is studied. Based on these verifications, a new RRT* based algorithm called RRT* Sukharev Vertices (RRT*-SV) is presented, in which both processes studied in this work are used. Computational tests were performed to verify if the RRT*-SV algorithm has shorter planning time than the RRT* algorithm and other literature algorithms such as RRT*-Smart, Informed-RRT* and BIT* (Batch Informed Trees Star) algorithms, which also uses non-uniform/informed sampling processes. The computational tests were performed considering several navigation environments with continuous and two-dimensional representation and with different quantities and spatial distributions of polygonal static obstacles. The results indicate that the proposed sampling process accelerates the RRT* planning time and shows that the RRT*-SV algorithm has better performance in various types of navigation environments when compared to the other algorithms considered. Due to its performance, the RRT*-SV algorithm is suitable for use in real-time UAV navigation applications that have as constraint the planning of the path with the shortest length.

Keywords: Path Planning. RRT. Sukharev Grids. Convex Vertices. Non uniform/informed sampling. UAV. Systematic Literature Review.

LISTA DE FIGURAS

Pág.

1.1 Etapas de planejamento de rota. 3
1.2 Crescimento da árvore do algoritmo RRT em planejamento de rota executado em ambiente de navegação com obstáculos estáticos e com representação poligonal. 6
1.3 Hodógrafo da velocidade de um VANT. 12

2.1 Esquema de comunicação remota entre estação de controle e o VANT. . . 16
2.2 Tendência de crescimento de autonomia estimada por ano pelos sistemas de navegação de VANTs. 18
2.3 Arquitetura típica aplicada para a navegação de VANTs. 20
2.4 Tipos de representação de VANT em ambiente de navegação. 22
2.5 Divisão do ambiente de navegação Q nas regiões Q_{livre} e $Q_{obstáculos}$. . . . 23
2.6 Tipos de representações de ambientes de navegação e obstáculos. 24

3.1 Estrutura usualmente utilizada para compor algoritmos de planejamento de rota baseados em amostragem. 28
3.2 Classificação de amostras. 28
3.3 Descrição da etapa de extensão do RRT. 31
3.4 Planejamento de uma rota pelo algoritmo RRT. 32
3.5 Exemplos de situações identificadas em $Q_{obstáculos}$ no teste de colisão. . . 34
3.6 Exemplo de diagrama de Voronoi dos nós da árvore do algoritmo RRT. . 36
3.7 Efeito da orientação de Voronoi no crescimento da árvore do RRT. 37
3.8 Etapas de extensão da árvore no algoritmo RRT*. 41
3.9 Etapa de reconexão de arestas para redução de comprimento da rota dos nós de uma vizinhança até $q_{início}$. 42
3.10 Exemplo de planejamento de rota pelo algoritmo RRT* em instantes distintos. 44
3.11 Diminuição do tamanho da região de vizinhança (círculo na cor roxa) com o aumento do número de nós da árvore no algoritmo RRT*. 45
3.12 Gráficos ilustrando a) a diminuição do comprimento da rota com o aumento do número de iterações e b) a redução do raio de formação de vizinhança com o aumento da quantidade de nós na árvore no algoritmo RRT*. 46

4.1 Distribuição das células geradas pela grade de Sukharev para $k = 196$. . 86
4.2 Estágios na amostragem pela grade de Sukharev. 89

4.3 Exemplo de uma envoltória de segurança (azul) para um obstáculo (preto). 90
4.4 Exemplo de planejamento por meio de grafo de visibilidade. 92
4.5 Estágios do processo de amostragem baseados em vértices convexos. . . . 96
4.6 Exemplos de menores distâncias Δs para valores de k distintos. 99
4.7 Taxa de sucesso de cada algoritmo em alcançar frações da rota ótima em 100 simulações com sementes distintas para geração de número pseudoaleatório para o ambiente de navegação com 50 obstáculos. 101
4.8 Taxa de sucesso de cada algoritmo em planejar rotas por tempo de planejamento em segundos (em escala de log_{10}) em 100 simulações com sementes distintas para geração de número pseudoaleatório para o ambiente de navegação com 50 obstáculos. 102
4.9 Taxa de sucesso de cada algoritmo em alcançar frações da rota ótima em 100 simulações com sementes distintas para geração de número pseudoaleatório para o ambiente de navegação com 100 obstáculos. 104
4.10 Taxa de sucesso de cada algoritmo em planejar rotas por tempo de planejamento em segundos (em escala de log_{10}) em 100 simulações com sementes distintas para geração de número pseudoaleatório para o ambiente de navegação com 100 obstáculos. 105
4.11 Taxa de sucesso de cada algoritmo em alcançar frações da rota ótima em 100 simulações com sementes distintas para geração de número pseudoaleatório para o ambiente de navegação com obstáculos em zigue-zague. . 107
4.12 Taxa de sucesso de cada algoritmo em planejar rotas por tempo de planejamento em segundos (em escala de log_{10}) em em 100 simulações com sementes distintas para geração de número pseudoaleatório para o ambiente de navegação com obstáculos em zigue-zague. 108
4.13 Taxa de sucesso de cada algoritmo em alcançar frações da rota ótima em 100 simulações com sementes distintas para geração de número pseudoaleatório para o ambiente de navegação com obstáculo com formato em espiral. 110
4.14 Taxa de sucesso de cada algoritmo em planejar rotas por tempo de planejamento em segundos (em escala de log_{10}) em 100 simulações com sementes distintas para geração de número pseudoaleatório para o ambiente de navegação com obstáculo com formato em espiral. 111
4.15 Taxa de sucesso de cada algoritmo em alcançar frações da rota ótima em 100 simulações com sementes distintas para geração de número pseudoaleatório para o ambiente de navegação com passagem estreita. 113

4.16 Taxa de sucesso de cada algoritmo em planejar rotas por tempo de planejamento em segundos (em escala de log_{10}) em 100 simulações com sementes distintas para geração de número pseudoaleatório para o ambiente de navegação com passagem estreita. 114
4.17 Taxa de sucesso dos algoritmos com amostragem baseada exclusivamente na grade de Sukharev (RRT*-Sukharev) com diferentes quantidade de células em alcançar frações da rota ótima em 100 simulações com sementes distintas para geração de número pseudoaleatório para diferentes ambientes de navegação. 116
4.18 Fluxograma do comportamento geral do algoritmo RRT*-SV. 118
4.19 Etapas do processo de amostragem baseada na grade de Sukharev executadas quando não há nenhum vértice convexo disponível para adição à árvore. 121
4.20 Etapas do processo de amostragem baseada em amostragem uniforme executadas quando não há nenhum centroide da grade de Sukharev e nenhum vértice convexo para adição à árvore. 122
4.21 Diagrama de bloco da estrutura de planejamento de rota utilizada pelo algoritmo RRT*-SV. 124
4.22 Primeiro conjunto de testes computacionais: Primeira rota planejada pelos algoritmos. 128
4.23 Primeiro conjunto de testes computacionais: Rotas planejadas após 30 000 iterações pelos algoritmos. 131
4.24 Primeiro conjunto de experimentos: Comprimento médio de rotas planejadas em 100 simulações por iteração. 133
4.25 Primeiro conjunto de testes computacionais: Média da quantidade de iterações para 100 simulações para cada algoritmo encontrar rotas com comprimentos específicos. 135
4.26 Primeiro conjunto de testes computacionais: Média de tempo de planejamento (em segundos) para 100 simulações para cada algoritmo encontrar rotas com comprimentos específicos. 137
4.27 Segundo conjunto de testes computacionais: Rotas planejadas para o ambiente de navegação com cinco obstáculos. 139
4.28 Segundo conjunto de testes computacionais: Rotas planejadas para o ambiente de navegação com 50 obstáculos. 140
4.29 Segundo conjunto de testes computacionais: Rotas planejadas para o ambiente de navegação com 2500 obstáculos. 141
4.30 Segundo conjunto de testes computacionais: Rotas planejadas para o ambiente de navegação com 10 000 obstáculos. 142

4.31 Segundo conjunto de testes computacionais: Tempo médio de planejamento (em segundos) considerando cinco simulações para os algoritmos planejarem rotas com comprimentos específicos no ambiente de navegação com cinco obstáculos. 146
4.32 Segundo conjunto de testes computacionais: Tempo médio de planejamento (em segundos) considerando cinco simulações para os algoritmos planejarem rotas com comprimentos específicos no ambiente de navegação com 50 obstáculos. 147
4.33 Segundo conjunto de testes computacionais: Tempo médio de planejamento (em segundos) considerando cinco simulações para os algoritmos planejarem rotas com comprimentos específicos no ambiente de navegação com 2500 obstáculos. 148
4.34 Segundo conjunto de testes computacionais: Tempo médio de planejamento (em segundos) considerando cinco simulações para os algoritmos planejarem rotas com comprimentos específicos no ambiente de navegação com 10 000 obstáculos. 149
4.35 Terceiro conjunto de testes computacionais: Primeiro conjunto de ambientes de navegação usados nas comparações estatísticas entre o algoritmo RRT*-SV e os algoritmos RRT*-Smart, Informed-RRT* e BIT* 159
4.36 Terceiro conjunto de testes computacionais: Segundo conjunto de ambientes de navegação usados nas comparações estatísticas entre o algoritmo RRT*-SV e os algoritmos RRT*-Smart, Informed-RRT* e BIT*. 160
4.37 Terceiro conjunto de testes computacionais: Terceiro conjunto de ambientes de navegação usados nas comparações estatísticas entre o algoritmo RRT*-SV e os algoritmos RRT*-Smart, Informed-RRT* e BIT*. 161
4.38 Terceiro conjunto de teste computacionais: Taxa de sucesso de cada algoritmo em alcançar frações da rota ótima em 100 simulações com sementes distintas para geração de número pseudoaleatório. 164
4.39 Terceiro conjunto de teste computacionais: Taxa de sucesso de cada algoritmo em planejar uma rota por tempo de planejamento em segundos (log_{10}) em 100 simulações com sementes distintas para geração de número pseudoaleatório. 166

5.1 Curva gerada em situação de colisão devido à uma envoltória de segurança (traço em verde) posicionada à uma distância curta de seu obstáculo. 170
5.2 Forma canônica do problema de suavização de cantos pontiagudos e exemplos de suavização para $\theta = \pi/4$, $\theta = \pi/2$ e $\theta = 3\pi/4$. 175

5.3 Cálculo da posição de vértice convexo (em azul) da envoltória de segurança em relação ao vértice convexo (em verde) do obstáculo (representado pela região cinza). 178
5.4 Rotas ótimas planejadas pelo algoritmo de Dijkstra em uma estrutura formada pelo algoritmo de grafo de visibilidade. 180
5.5 Rota planejada pelo algoritmo RRT*-SV com e sem suavização para $\kappa_{VANT} = 0,009$. 183
5.6 Rota planejada pelo algoritmo RRT*-SV com e sem suavização para $\kappa_{VANT} = 0,015$. 184

A.1 Rotas ótimas planejadas pelo algoritmo de Dijkstra sobre grafos de visibilidade para diferentes ambiente de navegação. 213

B.1 Processo de desenvolvimento da revisão sistemática da literatura. 222

LISTA DE TABELAS

Pág.

3.1 Estudos coletados no processo de revisão sistemática da literatura que se enquadram na classe de amostragem com orientação para a posição de destino. 50
3.2 Estudos coletados no processo de revisão sistemática da literatura que se enquadram na classe de amostragem com orientação para os obstáculos. . 56
3.3 Estudos coletados no processo de revisão sistemática da literatura que se enquadram na classe de amostragem com orientação regional. 59
3.4 Estudos coletados no processo de revisão sistemática da literatura que se enquadram na classe de amostragem com orientação para rota. 69
3.5 Estudos coletados no processo de revisão sistemática da literatura que se enquadram na classe de amostragem com orientação para passagens estreitas. 73
3.6 Estudos coletados no processo de revisão sistemática da literatura que se enquadram na classe de amostragem com orientação por redução do ambiente de navegação. 76
3.7 Estudos coletados no processo de revisão sistemática da literatura que se enquadram na classe de amostragem com orientação por modificação da distribuição das amostras. 79

4.1 Primeiro conjunto de testes computacionais: valores médios para 100 simulações para cada algoritmo planejar a primeira rota. 133
4.2 Segundo conjunto de testes computacionais: Valores médios de comprimento e tempo relacionados à primeira rota planejada em cinco simulações para cada algoritmo em ambientes de navegação com diferentes quantidades de vértices convexos. 143
4.3 Terceiro conjunto de testes computacionais: Resultados e parâmetros do teste t-Student para comparação de valores relativos as rotas planejadas pelos algoritmos para diferentes tipos de ambientes de navegação considerando 100 simulações. 152
4.4 Terceiro conjunto de testes computacionais: Resultados e parâmetros do teste t-Student para comparação de valores relativos as primeiras rotas planejadas pelos algoritmos para diferentes tipos de ambientes de navegação considerando 100 simulações. 155

5.1 Valores de L e d_{Obs} definidos para cada obstáculo do ambiente de navegação, considerando diferentes valores de κ_{VANT}. 181
5.2 Comprimentos de rotas não suavizada e suavizada por curvas HP planejadas pelo algoritmo RRT*-SV para diferentes valores de κ_{VANT}. 182

A.1 Comprimentos das rotas ótimas planejadas pelos algoritmos de grafo de visibilidade e Dijkstra para diferentes ambientes de navegação e suas respectivas posições $q_{início}$ e $q_{destino}$. 212

B.1 Questões de pesquisa desenvolvidas na revisão sistemática da literatura. . 218
B.2 Consultas de pesquisa estruturadas para a revisão sistemática 220
B.3 Lista de mecanismos de busca como fonte de estudos primários para revisão sistemática da literatura 220

LISTA DE ABREVIATURAS E SIGLAS

BIT*	–	*Batch Informed Trees Star*
DoD	–	*U.S. Department of Defense*
I.C.	–	Intervalo de Confiança
Informed-RRT*	–	*Informed Rapidly-exploring Random Tree Star*
RGG	–	*Random Geometric Graph*
RRT	–	*Rapidly-exploring Random Tree*
RRT*	–	*Rapidly-exploring Random Tree Star*
RRT*-Smart	–	*Rapidly-exploring Random Tree Star Smart*
RRT*-SV	–	*Rapidly-exploring Random Tree Star Sukharev Vertices*
RSL	–	Revisão Sistemática da Literatura
SLR	–	*Systematic Literature Review*
UGV	–	*Unmanned Ground Vehicle*
USV	–	*Unmanned Surface Vehicle*
UUV	–	*Unmanned Underwater Vehicle*
VANT	–	Veículo aéreo não tripulado

LISTA DE SÍMBOLOS

Q	–	ambiente de navegação
$Q_{obstáculos}$	–	obstáculos de Q
Q_{livre}	–	região livre de colisão de Q
O_i	–	i-ésimo obstáculos de $Q_{obstáculos}$
n_o	–	quantidade de obstáculos em $Q_{obstáculos}$
A_i	–	arestas de O_i
a_j	–	j-ésima aresta de A_i
V_i	–	vértices de O_i
v_j	–	j-ésimo vértice de V_i
p	–	rota planejada
$q_1, q_2, ..., q_m$	–	sequencia de *waypoints* que define uma rota
m	–	quantidade de *waypoints* em uma rota
$q_{início}$	–	*waypoint* inicial de navegação
$q_{destino}$	–	*waypoint* final de navegação
G	–	árvore de exploração do RRT
V	–	conjunto de vértices de G
E	–	conjunto de arestas de G
q_{novo}	–	novo nó adicionado à G
$q_{próximo}$	–	nó mais próximo de q_{novo}
$q_{aleatório}$	–	amostra gerada aleatoriamente em Q
Δq	–	passo de expansão do algoritmo RRT
n	–	quantidade de iterações
s	–	indicador de rota planejada
l_d	–	distância para conexão de um nó de G com $q_{destino}$
r_n	–	raio de busca de vizinhos do algoritmo RRT*
β	–	constante no cálculo de r_n
Δs	–	passo de expansão por grade de Sukharev
q_{novo_temp}	–	amostra temporária na seleção de célula de grade de Sukharev
k	–	quantidade de células na grade de Sukharev
$q_{vpróximo}$	–	vértice convexo mais próximo de $q_{próximo}$
n_t	–	total de vértices convexos em $Q_{obstáculos}$
n_v	–	quantidade de vértices convexos em $Q_{vértices}$
Q_{beacon}	–	nós da última rota planejada no algoritmo RRT*-Smart
b	–	constante no algoritmo RRT*-Smart
t	–	indicador de uso da amostragem inteligente no algoritmo RRT*-Smart
u	–	incremento para uso da amostragem inteligente no algoritmo RRT*-Smart
$c_{corrente}$	–	comprimento da última rota planejada no algoritmo Informed-RRT*
$c_{ótimo}$	–	menor comprimento de rota no algoritmo Informed-RRT*
Q_{elipse}	–	região de amostragem no algoritmo Informed-RRT*
q_{elipse}	–	nova amostra coletada dentro de Q_{elipse}

$Q_{amostras}$	–	lote (*batch*) de amostras geradas no algoritmo BIT*
$F_{vértices}$	–	fila ordenada de vértices no algoritmo BIT*
$F_{arestas}$	–	fila ordenada de arestas no algoritmo BIT*
ξ	–	parâmetro da curva HP
$\boldsymbol{p_k}$	–	ponto de controle da curva HP
S	–	comprimento do arco
$\boldsymbol{\sigma_k}$	–	coeficiente de Bernstein da curva paramétrica
$\boldsymbol{w_k}$	–	coeficiente de Bernstein na forma complexa
$\bar{\boldsymbol{w}}$	–	conjugado de $\boldsymbol{w}$
$\boldsymbol{w'}$	–	derivada de $\boldsymbol{w}$
θ	–	ângulo de deslocamento
κ_e	–	maior valor de curvatura em uma curva
L	–	distância de suavização
d	–	distância entre um obstáculo e sua envoltória de segurança
κ_{VANT}	–	capacidade de curvatura do VANT
L_{VANT}	–	distância de suavização para VANT com curvatura κ_{VANT}
d_i	–	distância entre o i-ésimo obstáculo e sua envoltória de segurança
d_{Obs}	–	maior distância entre obstáculos e envoltórias de segurança

SUMÁRIO

Pág.

1 INTRODUÇÃO 1
1.1 Hipóteses e contribuições da tese 9
1.2 Estrutura da tese 12

2 PLANEJAMENTO DE ROTA DE VEÍCULOS AÉREOS NÃO TRIPULADOS 15
2.1 Autonomia na navegação de VANTs 15
2.1.1 Arquitetura de navegação de VANTs 18
2.2 Modelagem do ambiente de navegação 21
2.2.1 Modelo de ambiente de navegação adotado 25

3 MÉTODOS DE PLANEJAMENTO DE ROTA BASEADOS EM AMOSTRAGEM 27
3.1 Árvore aleatória de exploração rápida (RRT) 29
3.1.1 Teste de colisão 33
3.1.2 Características e limitações 35
3.2 RRT com otimalidade assintótica (RRT*) 38
3.2.1 Exemplo de aplicação do algoritmo RRT* ao planejamento de rotas . . 43
3.2.2 Características e limitações 46
3.3 Amostragem não uniforme/informada em algoritmos baseados no RRT . 47
3.3.1 Amostragem por importância 49
3.3.1.1 Amostragem com orientação para a posição de destino 49
3.3.1.2 Amostragem com orientação para os obstáculos 55
3.3.1.3 Amostragem com orientação regional 58
3.3.1.4 Amostragem com orientação para rota 68
3.3.1.5 Amostragem com orientação para passagens estreitas 72
3.3.2 Amostragem adaptativa 75
3.3.2.1 Amostragem com orientação por redução do ambiente de navegação . 75
3.3.2.2 Amostragem com orientação por modificação da distribuição das amostras 78

4 ESTUDO DE PROCESSOS DE AMOSTRAGEM BASEADOS EM GRADES DE SUKHAREV E VÉRTICES CONVEXOS . . 85

4.1 Grade de Sukharev 85
4.1.1 RRT*-Sukharev 87
4.2 Vértices convexos de envoltórias de segurança dos obstáculos 90
4.2.1 RRT*-Vértices 93
4.3 Testes computacionais com o algoritmo RRT* usando processos de amostragem baseado em Grades de Sukharev e baseado em Vértices Convexos 97
4.3.1 Análise dos resultados 99
4.4 RRT* com processo de amostragem híbrido baseado em Grades de Sukharev e Vértices Convexos: Algoritmo RRT*-SV 117
4.4.1 Detecção de colisão no algoritmo RRT*-SV 123
4.4.2 Análise assintótica da complexidade de tempo do algoritmo RRT*-SV . 124
4.5 Testes computacionais com o algoritmo RRT*-SV 126
4.5.1 Parâmetros dos algoritmos 126
4.5.2 Primeiro conjunto de testes computacionais 127
4.5.3 Segundo conjunto de testes computacionais 138
4.5.4 Terceiro conjunto de testes computacionais 150
4.5.5 Aspectos técnicos dos experimentos 167
4.6 Considerações 167

5 PROCESSO DE SUAVIZAÇÃO DE ROTAS 169
5.1 Hodógrafo de Pitágoras 171
5.1.1 Representação complexa de curva HP quíntica 173
5.2 Problema de suavização de cantos com curvas HP quínticas com continuidade G^2 175
5.3 Cálculo da distância de geração de envoltória de segurança para valor de curvatura do VANT 177
5.3.1 Teste computacional e resultados 179
5.4 Considerações 185

6 CONCLUSÃO 187
6.1 Trabalhos futuros 190

REFERÊNCIAS BIBLIOGRÁFICAS 191

APÊNDICE A - CÁLCULO DO COMPRIMENTO ÓTIMO DE ROTAS COM DIJKSTRA E GRAFOS DE VISIBILIDADE 211

APÊNDICE B - REVISÃO SISTEMÁTICA DA LITERATURA . . 215
B.1 Objetivo desta Revisão Sistemática da Literatura 215

B.1.1 Definição do escopo das questões de pesquisa 215
B.2 Protocolo de revisão 217
B.2.1 Questões de pesquisa 217
B.2.2 *Strings* de pesquisa 219
B.2.3 Consultas para pesquisa de estudos primários 219
B.2.4 Mecanismos de busca considerados no processo de revisão 220
B.2.5 Critérios para seleção dos estudos primários 220
B.2.6 Extração de dados 221
B.3 Desenvolvimento da Revisão Sistemática da Literatura 222
B.3.1 Descrição das fases da Revisão Sistemática da Literatura 223
B.3.2 Validação da Revisão Sistemática da Literatura 224

ANEXO A - ALGORITMOS DA LITERATURA UTILIZADOS NOS TESTES COMPUTACIONAIS COM O ALGORITMO RRT*-SV 225
A.1 RRT*-Smart 225
A.2 Informed-RRT* 229
A.3 Batch Informed Trees - BIT* 232

1 INTRODUÇÃO

Os Veículos Aéreos Não Tripulados (VANTs) foram inicialmente empregados em larga escala no meio militar, tendo posteriormente, principalmente nas últimas décadas, se popularizado no meio civil (GUNDLACH, 2012; JUNIPER, 2018). A popularização do uso dessas aeronaves pode ser atribuída à redução dos custos de fabricação e miniaturização dos seus componentes (FLOREANO; WOOD, 2015) e ao surgimento de diversas oportunidades de aplicações como: resgate, entrega de produtos e suprimentos, vistoria de estruturas de difícil acesso, sensoriamento remoto, mitigação de desastres, telecomunicações, e até mesmo em competições e lazer. Essa popularização dos VANTs levou ao desenvolvimento de uma crescente área de estudo: o aumento de autonomia desses veículos (GROCHOLSKY et al., 2006; LUDINGTON et al., 2007). Na realidade, o interesse no aumento de autonomia abrange não somente aeronaves, mas todo o contexto de veículos autônomos. Segundo diversos autores (CHOSET et al., 2005; INTERNATIONAL STANDARDIZATION ORGANIZATION (ISO), 2012; LAVALLE, 2006; SIEGWART et al., 2011), ter autonomia significa, não só para VANTs, mas para robôs e veículos em geral, possuir a capacidade de realizar uma tarefa designada sem a intervenção de um operador. Goerzen et al. (2010) e Floreano e Wood (2015) argumentam que o nível de autonomia de aeronaves é percebido pela adição de responsabilidades e capacidades de decisão como: execução de trajetórias, localização de alvos e mapeamento do ambiente, solução de conflitos de informações, planejamento de recarga de bateria, reconhecimento de objetos e pessoas, reação a pertubações de vento e falhas eletromecânicas e de comunicação, retorno para a base de voo, planejamento de rota, etc. Esta tese aborda o estudo de metodologias que possam aumentar a capacidade de autonomia referente ao planejamento de rotas em VANTs.

O planejamento automático de rota faz parte de um esquema mais abrangente empregado na navegação autônoma de uma aeronave. Em aplicações mais tradicionais, com menor nível de autonomia, a responsabilidade de controle do VANT fica a cargo de uma estação de controle remota, operada por humanos, que envia por telemetria/rádio frequência os comandos de navegação. Atualmente, pesquisas vêm sendo desenvolvidas com o intuito de aumentar a autonomia de VANTs, o que consiste em transferir parte dos processos de tomada de decisão da estação de controle para o próprio veículo, inclusive o planejamento de rota. Neste caso, o VANT teria a capacidade de planejamento de suas próprias rotas de navegação.

Uma das vantagens da capacidade de planejamento automático de rota para um

VANT é o aumento da segurança para o próprio veículo. Caso ocorra um problema de comunicação em um sistema de telemetria do VANT, que está sendo remotamente controlado, o mesmo pode detectar essa falha e planejar um rota de segurança para navegar até um local que permita que a comunicação seja restaurada, evitando a perda do veículo. Em uma situação em que há uma perda permanente do sistema de comunicação, como danos na antena de rádio, o VANT pode planejar uma rota para um local seguro em uma tentativa de aterrissagem, evitando a exposição de pessoas aos riscos deste procedimento de pouso. Outra vantagem seria a capacidade do VANT de se adaptar a mudanças em uma tarefa previamente planejada, gerando novas rotas de navegação para realizar novas tarefas. Revisões sobre o desenvolvimento e aplicações do planejamento de rotas para VANTs estão disponíveis na literatura (GOERZEN et al., 2010; NEX; REMONDINO, 2014).

Os processamentos relativos às etapas de planejamento de rota usualmente são encapsulados por um módulo (software, hardware, etc) consultado pelo sistema de navegação. Esse módulo é acoplado à estação de controle, onde a rota é computada *offline* antes que a tarefa inicie e, em seguida, a solução é enviada para o VANT pelo sistema de comunicação. Em um cenário ainda melhor, o módulo de planejamento de rota pode ser embarcado no VANT e a rota de navegação pode ser planejada em tempo real (*online*) por meio de informações obtidas por sensores ou por representações computacionais do ambiente de navegação previamente construídas (MEDEIROS, 2012). Consequentemente, as aeronaves serão capazes de planejar rotas por si próprias durante a execução de uma tarefa, em caso de alterações do estado da mesma ou da detecção de perturbações no ambiente de navegação (PHARPATARA, 2015).

O planejamento automático de rotas é um problema complexo, principalmente quando existem regiões por onde a aeronave é impossibilitada de navegar. Essas regiões podem ser zonas de exclusão aérea (regiões proibidas para a navegação) ou áreas ocupadas por obstáculos que eventualmente o VANT possa vir a colidir. Além disso, outros elementos devem ser considerados durante o planejamento de uma rota para um VANT, como as restrições cinemáticas e dinâmicas da aeronave, entre outros. O problema de planejamento de rota também pode ser observado com outras categorias de veículos, como Veículos Terrestres Não Tripulados (UGV, do inglês *Unmanned Ground Vehicles*) (GAGE, 1995; KAYACAN et al., 2016), Veículos de Superfície Não Tripulados (USV, do inglês *Unmanned Surface Vehicles*) (CACCIA et al., 2008; ŠVEC et al., 2014) e Veículos Submarinos Não Tripulados (UUV, do inglês *Unmanned Underwater Vehicles*) (WANG et al., 2017a; WANG et al., 2017b).

Considerando os tipos de restrições anteriormente descritos, no problema de planejamento de rota, uma rota cinematicamente e dinamicamente viável deve ser planejada entre dois pontos (posições) de um ambiente de navegação, permitindo que o VANT navegue pela mesma em segurança, evitando para isso zonas de exclusão aérea e situações de colisão com obstáculos naquele ambiente. Geralmente, o planejamento de rota consiste em três etapas principais: a modelagem computacional de um ambiente de navegação; o planejamento de uma rota de navegação livre de colisão; e a sua conversão em uma rota dinamicamente viável (GOERZEN et al., 2010; LAVALLE, 2006). Cada uma dessas etapas encontra-se ilustrada na Figura 1.1.

Figura 1.1 - Etapas de planejamento de rota.

a) Modelagem do ambiente de navegação; b) Formação da rota por segmentos de retas que conectam os pontos inicial e final de navegação; c) Suavização da rota, com rota original em tracejado marrom.

Fonte: Produção do Autor.

A modelagem do ambiente de navegação consiste em uma abstração ou modelo do ambiente real que viabilize um referencial para a navegação da aeronave (STURTEVANT, 2013). A eficiência dos métodos aplicados nas próximas etapas de planejamento geralmente está relacionada ao modelo do ambiente de navegação adotado (TOZOUR, 2003). As regiões navegáveis e todos os obstáculos do ambiente podem ser representados por meio de polígonos, grades regulares (BARRAQUAND; LATOMBE, 1991), malhas de navegação (ARKIN, 1987), diagramas de Voronoi (VORONOÏ, 1908), *quadtrees* (FINKEL; BENTLEY, 1974), dentre outras formas de representação.

Na etapa seguinte de planejamento da rota, segmentos de reta livres de colisão com os obstáculos são obtidos de forma a conectar os pontos de início e de fim da navegação do VANT, utilizando a representação computacional do ambiente de navegação definido na etapa anterior. As rotas são definidas por um conjunto de pontos sequenciais denominados *waypoints*, que são interligados pelos segmentos de reta livres de colisão para conectar um *waypoint* inicial (saída do VANT) até um *waypoint* final (chegada/destino do VANT) (TSOURDOS et al., 2010). As rotas podem ser extraídas de estruturas como redes, grafos, árvores ou vetores computacionais, dependendo da concepção e formalização do problema tratado e dos métodos utilizados no planejamento de rota. Alguns exemplos de métodos que podem ser utilizados para o planejamento automático de rotas são: o algoritmo de Dijkstra para busca em grafo (DIJKSTRA, 1959); o algoritmo A* (HART et al., 1968); os algoritmos genéticos (HOLLAND, 1975), o algoritmo *Probabilistic Roadmaps* (PRM) (KAVRAKI et al., 1996), o algoritmo de Grafos de Visibilidade (NILSSON, 1969); e o algoritmo Árvore Aleatória de Exploração Rápida (RRT, do inglês *Rapidly-exploring Random Tree*) (LAVALLE, 1998).

Na terceira e última etapa, denominada suavização, a rota planejada é então convertida em uma trajetória de navegação livre de colisão e dinamicamente viável, com base nas restrições cinemáticas e dinâmicas do VANT: velocidade; aceleração; curvatura; entre outras. Usualmente, são utilizados métodos polinômicos de suavização de curvas que atendem a essas restrições. São exemplos de métodos aplicados na suavização de rotas: trajetórias de Dubins (DUBINS, 1957), k-trajetórias (ANDERSON et al., 2005), clotóides (ALIA et al., 2015), *spline* (MAEKAWA et al., 2010), curvas de Bézier (DO et al., 2011), curva logística (UPADHYAY; RATNOO, 2016), *Support Vector Machine* (SVM) (YANG et al., 2012) e Hodógrafo de Pitágoras (TSOURDOS et al., 2010). É importante mencionar que, neste trabalho, a rota de menor comprimento possível entre dois pontos de um ambiente de navegação é considerada a solução ótima do problema de planejamento de rotas, desde que o VANT tenha

autonomia (alcance obtido pela quantidade de energia/combustível) para realizar uma navegação completa pela mesma.

Em aplicações onde a rota é planejada a bordo da aeronave, o tempo de execução do módulo de planejamento da rota pode influenciar diretamente na segurança do VANT. Consequentemente, os algoritmos de planejamento não podem exigir custos computacionais muito elevados, devido às restrições inerentes à navegação em tempo real e das capacidades computacionais dos módulos embarcados. Dessa forma, é importante que um problema de planejamento de rotas seja solucionado com eficiência de tempo computacional (PHARPATARA, 2015). Em diversos trabalhos da literatura, diferentes soluções para o planejamento de rotas foram propostas considerando o desempenho computacional, dentre elas uma classe de algoritmos conhecida como algoritmos de planejamento baseados em amostragem (LAVALLE, 2006). Nessa classe de algoritmos, o ambiente de navegação não é explicitamente incorporado à estrutura do mesmo, como ocorre em algoritmos mais tradicionais de planejamento. Em vez disso, amostras (pontos) do ambiente de navegação são coletadas para compor a solução de planejamento de rota. A cada coleta de uma amostra, é verificado se ela, antes de ser incorporada à rota que está sendo planejada, gera uma situação de colisão. Em caso afirmativo, ela é descartada da solução e outras amostras são verificadas. Para realizar essa verificação, esses algoritmos utilizam um módulo à parte de sua estrutura para a detecção de colisão. Esse módulo é quem possui acesso ao modelo computacional do ambiente de navegação. Essa classe de algoritmos é popular em grande parte devido à sua eficiência de planejamento e simplicidade computacional (ISLAM et al., 2012; LAVALLE, 2006; THRUN et al., 2005).

O RRT, é um dos algoritmos de amostragem mais aplicados ao problema de planejamento automático de rotas. O algoritmo RRT é probabilisticamente completo, isto é, se existe uma solução de planejamento de rota, a probabilidade do mesmo encontrá-lo aproxima-se de um à medida que o número de amostras coletadas aumenta e tende a infinito. Esse algoritmo consiste em expandir uma árvore, em formato de grafo, a partir de um nó raiz, o qual é o ponto inicial da rota, utilizando um processo de coleta aleatória de amostras de um ambiente de navegação. Cada amostra coletada é adicionada ao nó mais próximo da árvore se o segmento de reta formado por esses nós não estiver em situação de colisão com os obstáculos do ambiente de navegação. A árvore é expandida até que seja possível conectar um nó da árvore ao ponto final da rota. Na Figura 1.2 encontra-se ilustrado um exemplo de planejamento de rota pelo algoritmo RRT.

Figura 1.2 - Crescimento da árvore do algoritmo RRT em planejamento de rota executado em ambiente de navegação com obstáculos estáticos e com representação poligonal.

Estados da expansão da árvore exploratória do RRT em iterações distintas encontram-se representadas nas figuras. Em a), b) e c), os retângulos em azul marinho representam os obstáculos. O ponto verde corresponde ao *waypoint* inicial de navegação. O ponto vermelho corresponde ao *waypoint* final de navegação. Os pontos em amarelo são nós da árvore do algoritmo RRT e os segmentos de reta em verde são as arestas que os conectam. Em c), a rota planejada encontra-se representada pelos segmentos de reta em vermelho conectado pelos *waypoints* na cor roxa.

Fonte: Produção do Autor.

A RRT permite uma rápida exploração em ambientes de navegação multidimensionais. Segundo Lindemann e LaValle (2005), isto ocorre devido à orientação na amostragem acarretada por regiões de Voronoi, ou seja, a cada iteração a árvore tende a crescer a partir dos nós com a maior área de Voronoi. Isso ocorre porque a probabilidade que um nó seja selecionado para a expansão da árvore do RRT é diretamente proporcional ao volume da sua célula de Voronoi. Assim, a árvore é rapidamente induzida a crescer para regiões menos exploradas.

Apesar das vantagens apresentadas pelo algoritmo RRT, não há garantia que o mesmo possa planejar a rota com o menor comprimento/extensão possível para um ambiente de navegação específico, ou rotas com determinados comprimentos limitados pela autonomia do veículo. Algumas derivações do algoritmo RRT foram propostas com o intuito de procurar solucionar essa limitação. Esses algoritmos são assintoticamente ótimos, o que significa que a rota mais curta pode ser obtida por eles como uma solução, desde que o número de amostras coletadas tenda ao infinito. Exemplos desses algoritmos são o Grafo Aleatório de Exploração Rápida (RRG, do inglês *Rapidly-exploring Random Graph*) (KARAMAN; FRAZZOLI, 2009) e o RRT* (lê-se RRT estrela) (KARAMAN; FRAZZOLI, 2010).

No algoritmo RRT*, para cada novo nó adicionado à árvore, uma vizinhança de nós é gerada dentro de uma região determinada por um dado raio de distância. Diferente do algoritmo RRT padrão, o novo nó amostrado não é adicionado ao nó mais próximo. Em vez disso, o novo nó amostrado é conectado ao nó vizinho com o comprimento de rota mais curto até o nó raiz (ponto inicial da rota). Em seguida, verificam-se os outros nós de sua vizinhança, para que possam ser reconectados com esse novo nó e então reduzir também os seus comprimentos de rota até o nó raiz da navegação. Assim, quando uma rota completa é planejada, espera-se obter a rota mais curta entre o nó raiz e o ponto final de navegação. Portanto, há uma probabilidade maior de que a rota de navegação planejada pelo algoritmo RRT* tenha um comprimento menor que as rotas planejadas pelo algoritmo RRT (KARAMAN; FRAZZOLI, 2010).

É importante frisar que, apesar de a rota de menor comprimento possível poder ser planejada, existem algumas desvantagens apresentadas pelo algoritmo RRT*. Nesse algoritmo, o tempo de planejamento é consideravelmente maior do que no algoritmo RRT. Isso se deve à adição do processo de reconexão das arestas durante a expansão da árvore, o que envolve testes adicionais de colisão e operações de busca de nós mais próximos na sua execução. Além disso, o algoritmo apenas garante que a solução

obtida irá convergir para a rota de menor comprimento se o tempo de planejamento tender ao infinito.

Vários trabalhos propuseram melhorias do algoritmo RRT e RRT*, visando a redução do tempo de convergência para uma primeira solução ou para rotas de menor comprimento. Alguns exemplos são citados a seguir: *Adaptative RRT Based on Dynamic Step (DRRT)* (LIN; ZHANG, 2015), *Fast RRT* (MA et al., 2015), *Real-time Closed-loop Rapidly Exploring Random Trees (CL-RRT)* (FRAZZOLI et al., 2002) e sua variante *Closed-loop Random Belief Trees (CL-RBT)* (LI et al., 2014), *Resolution Complete Rapidly-Exploring Random Tree (RC-RRT)* (CHENG; LAVALLE, 2001), *Execution extended Rapidly Exploring Random Tree (ERRT)* (BRUCE; VELOSO, 2006), *Chance-Constraint Rapidly Exploring Random Tree (CC-RRT)* (LUDERS et al., 2010), *CC-RRT*-D* (LIU; ANG, 2014), *RRT-Connect* (KUFFNER; LAVALLE, 2000), entre outros.

Umas das estratégias propostas na literatura para acelerar o tempo de planejamento de rota por algoritmos baseados no RRT é a modificação do seu processo de amostragem. Nesse processo, uma amostra (posição no ambiente de navegação contínuo) é coletada aleatoriamente seguindo uma distribuição espacial uniforme a cada iteração do algoritmo para constituir a árvore exploratória. Dessa forma, qualquer amostra possui a mesma probabilidade de ser coletada. Alguns autores (GAMMELL, 2017; LINDEMANN; LAVALLE, 2006; NASIR et al., 2013) afirmam que reduzir/modificar o uso da distribuição uniforme no processo de amostragem possui potencial para acelerar o planejamento da rota do algoritmo RRT. Em muitos estudos, heurísticas/estratégias são utilizadas para influenciar o processo de amostragem dos algoritmos RRT e RRT*, priorizando a coleta de amostras em regiões do ambiente de navegação com maiores chances de gerarem rotas de forma mais rápida. Esse tipo de estratégia de amostragem é denominada não uniforme (LAVALLE, 2006; LINDEMANN; LAVALLE, 2006) ou informada (GAMMELL, 2017). Entre os estudos que propõem versões do RRT com amostragem não uniforme/informada alguns são mencionados: *Voronoi Bias RRT* (VB-RRT)(LINDEMANN; LAVALLE, 2004), RRT baseado em geração de teste orientado ao destino (AHMADYAN et al., 2012), *RRT*-Smart* (ISLAM et al., 2012), *RRT baseada em retração seletiva* (LEE et al., 2012), *Poisson RRT* (PARK et al., 2014), *Potential Guided Directional-RRT* (P-RRT*)* (QURESHI; AYAZ, 2016), *Informed-RRT** (GAMMELL et al., 2014), BIT* (GAMMELL et al., 2015), RRT baseado em probabilidades de amostragem e lógica *fuzzy* (ABBADI; MATOUSEK, 2018). Uma Revisão Sistemática da Literatura (RSL) (KITCHENHAM; CHARTERS, 2007) sobre os tipos de amostragem não uniforme/informada utilizados

em algoritmos do tipo RRT e RRT* é discutida no Capítulo 3 e no Apêndice B desta tese.

1.1 Hipóteses e contribuições da tese

Considerando que o crescimento da árvore do algoritmo RRT e, portanto, do algoritmo RRT*, pode ser direcionado para regiões mais promissoras do ambiente de navegação que levem ao planejamento mais rápido de rotas, esta tese tem como questionamento científico verificar se é possível obter rotas de menor comprimento com menor tempo de planejamento com o algoritmo RRT* utilizando vértices convexos e estratégias de dispersão ótima de amostras. Portanto, a contribuição deste trabalho é o estudo do uso de duas diferentes estratégias como solução de amostragem não uniforme/informada no algoritmo RRT*: as distribuições espaciais de amostras baseadas em grades de Sukharev (SUKHAREV, 1971); e a aplicação de amostras definidas pelos vértices convexos das envoltórias de segurança dos obstáculos do ambiente de navegação.

Uma grade de Sukharev alcança a distribuição mais uniforme possível sobre o ambiente de navegação. O objetivo dessa estrutura é permitir a cobertura/amostragem de todas as regiões de um ambiente de navegação de forma discretizada (LAVALLE, 2006). Os efeitos esperados de seu uso são um número menor de pontos amostrados para planejar uma rota viável para navegação, com o intuito de diminuir o tempo de planejamento de uma rota. O uso de grades de Sukharev em algoritmos baseados em amostragem já foi explorado experimental e teoricamente em estudos anteriores (JANSON et al., 2018; LINDEMANN; LAVALLE, 2003; SÁNCHEZ, 2003), nos quais foi estudada a eficiência da grade de Sukharev em algoritmos baseados em amostragem. Entretanto, nesses estudos apenas a redução do número de nós na rota planejada foi analisada ou o RRT não foi considerado nos experimentos.

Já a aplicação de vértices convexos no planejamento de rotas é mais conhecida através da aplicação de grafos de visibilidade (LATOMBE, 2012). Um grafo de visibilidade permite gerar uma estrutura através da qual pode ser planejada a rota mais curta entre duas posições em um ambiente de navegação por meio de algoritmo ótimo de busca como, por exemplo, o algoritmo de Dijkstra (DIJKSTRA, 1959). Desta forma, uma rota é planejada próxima dos limites dos obstáculos (ou da envoltória de segurança do mesmo), com o objetivo de obter a rota com o comprimento mais curto possível, isto é, a solução ótima.

Considerando a segurança do VANT durante a navegação, os vértices convexos uti-

lizados no planejamento de rota no estudo conduzido pertencem às envoltórias de segurança dos obstáculos. A envoltória de segurança é uma região que confina todo o corpo de um obstáculo, porém, mantém o seu formato original. Os vértices da envoltória são definidos a uma dada distância de segurança dos vértices de seu obstáculo, gerando as arestas da envoltória, as quais são deslocadas em relação as do obstáculo associado à mesma. Consequentemente, a rota pode ser planejada próxima das bordas da envoltória (o que miniminiza o comprimento da rota) sem que os limites dos obstáculos sejam alcançados pelo corpo físico da aeronave, evitando possíveis colisões.

Um novo algoritmo é proposto neste trabalho para estudar as estratégias consideradas de amostragem não uniforme/informada. Esse algoritmo, denominado RRT*-Sukharev Vertices (RRT*-SV), foi descrito em Véras et al. (2018a). A finalidade de usar essas duas estratégias no processo de amostragem é planejar a rota mais curta possível em um tempo de planejamento menor do que o apresentado por outras metodologias. A melhora significativa do tempo de planejamento no RRT*-SV é demonstrada por meio de comparação estatística entre os resultados obtidos com os algoritmos RRT*, RRT*-Smart, Informed-RRT* e BIT*, quando aplicados ao planejamento de rotas em ambientes de navegação bidimensionais com diferentes distribuições espaciais e formatos de obstáculos estáticos. Os algoritmos citados foram selecionados devido à sua eficiência reconhecida pela literatura e por utilizarem estratégias de amostragem que também visam o planejamento da rota com menor comprimento para um dado ambiente de navegação em menor tempo. O algoritmo RRT* foi selecionado porque é a estrutura básica de todas as soluções discutidas nas comparações. Desde que o método proposto possa obter soluções ótimas em menor tempo, ele pode ser utilizado na etapa de planejamento automático de rotas em sistemas de navegação de VANTs com o objetivo de aumento de autonomia.

Nesta tese também encontram-se descritos os resultados da elaboração e execução de uma RSL com tema sobre métodos de amostragem não uniforme/informada aplicados em algoritmos baseados no RRT. A RSL é uma metodologia de revisão bastante popular nas áreas de engenharia de software (KITCHENHAM, 2004; BUDGEN; BRERETON, 2006) e medicina (STRAUS et al., 2005). Essa revisão bibliográfica é organizada de forma estruturada, onde motores de busca (portais na *web* onde trabalhos acadêmicos indexados podem ser consultados), *strings* para a pesquisa, critérios usados para incluir e excluir um estudo e todas as informações sobre o processo de revisão são especificados em um documento denominado protocolo de revisão. O protocolo é desenvolvido em conjunto com uma equipe de pesquisadores com experiência no

tema, os quais definem o tipo de evidências acadêmicas de que uma pesquisa dentro do tema foi realizada devem ser pesquisadas. A revisão é então guiada pelo protocolo de pesquisa. Assim, além de permitir uma avaliação do processo de revisão por terceiros, espera-se que possíveis desvios de interpretação causados pela visão pessoal dos pesquisadores envolvidos na revisão sejam reduzidos (KITCHENHAM, 2004). Os resultados da RSL apresentada nesta tese encontram-se publicados em Véras et al. (2019b). O protocolo de revisão desenvolvido encontra-se descrito no Apêndice B.

Em adição às contribuições declaradas anteriormente, uma estratégia que visa garantir que a rota planejada pelos métodos propostos citados seja viável para a navegação do VANT é apresentada. Nessa estratégia, uma restrição de curvatura é aplicada junto ao algoritmo RRT*-SV para que sejam planejadas somente rotas que atendam ao limite de curvatura máxima que um VANT pode executar. Para isso, a rota planejada é suavizada nas junções (nós) de seus segmentos para que a geometria da mesma atenda as restrições de movimento do VANT. A estratégia de suavização apresentada adota como representação da trajetória a curva de Hodógrafo de Pitágoras (FAROUKI; SAKKALIS, 1990). Ela é uma Curva de Bézier (PIEGL, 1993) onde os componentes de seu hodógrafo (vetores velocidade que, unidos por uma origem, formam a curva de aceleração de movimento (NUSSENZVEIG, 2013), como descrito na Figura 1.3) atendem ao teorema de Pitágoras. Essa representação da curva, dentre outras vantagens, permite a computação exata do comprimento da curva (FAROUKI; NITTLER, 2016), dispensando a necessidade de aplicações de métodos de quadratura para o cálculo do comprimento da curva. Dessa forma, é possível converter a rota planejada em movimento através de um processo de estimação da posição do VANT na rota, baseado na relação entre o comprimento da rota calculado e a velocidade do veículo. A estratégia apresentada é baseada no processo de suavização apresentado em Véras et al. (2018a).

Figura 1.3 - Hodógrafo da velocidade de um VANT.

a) Trajetórias realizadas por um VANT em diferentes posições p_i e b) hodógrafo formado pelos vetores velocidade v_i do VANT unidos por uma mesma origem O.

Fonte: Produção do Autor.

1.2 Estrutura da tese

No Capítulo 2, são discutidos as etapas envolvidas no processo de planejamento de rota aplicados na navegação autônoma de VANTs, os diferentes níveis de autonomia e os modelos de representação de ambientes de navegação.

No Capítulo 3, é introduzida a classe de algoritmos de planejamento baseados em amostragem. Os algoritmos RRT e RRT* encontram-se descritos e as suas características que influenciam os seus desempenhos de planejamento são discutidos. Encontram-se ainda descritos no referido capítulo os resultados de uma RSL com tema sobre estratégias de amostragem não uniforme/informada em algoritmos baseados no RRT e RRT*.

No Capítulo 4, o estudo das duas estratégias de amostragem não uniforme/informada baseadas nas grades de Sukharev e vértices convexos das envoltórias de seguranças dos obstáculos é apresentado. É também proposto nesse capítulo o método desenvolvido a partir desse estudo, denominado RRT*-SV. Resultados de testes computacionais do RRT*-SV com outros algoritmos da literatura encontram-se descritos e discutidos.

No Capítulo 5, uma estratégia de suavização de rotas planejadas pelo RRT*-SV baseada no valor do raio de curvatura de VANTs é apresentada. Essa estratégia utiliza o Hodógrafo de Pitágoras.

No Capítulo 6, as conclusões desta tese são sintetizadas. As publicações decorrentes

da pesquisa de doutorado encontram-se listadas e os principais desafios futuros de pesquisa sobre o tema são apontados.

2 PLANEJAMENTO DE ROTA DE VEÍCULOS AÉREOS NÃO TRIPULADOS

Neste capítulo são introduzidos conceitos e ideias básicas sobre navegação autônoma de VANTs e como o problema de planejamento de rota está relacionado com arquiteturas utilizadas para a efetiva navegação desses veículos. Inicialmente, é discutido o contexto no qual está inserida a motivação desta tese, que é o aumento do grau de autonomia de navegação de VANTs através de otimização de algoritmos utilizados no planejamento automático de rotas. Um padrão de arquitetura comum a diversos sistemas de navegação também é discutido. Por fim, tipos de modelos computacionais do ambiente de navegação são apresentadas e o modelo adotado nesta tese é estabelecido.

2.1 Autonomia na navegação de VANTs

O Departamento de Defesa dos Estados Unidos (DoD, do inglês *U.S. Department of Defense*) define um veículo autônomo como

> Um veículo motorizado que não carrega um operador humano e que pode ser operado de forma autônoma ou remota, pode ser descartável ou recuperável, e pode transportar uma carga letal ou não letal. Veículos balísticos ou semi-balísticos, mísseis de cruzeiro, projéteis de artilharia, torpedos, minas, satélites e sensores não assistidos (sem forma de propulsão) não são considerados veículos não tripulados. Veículos não tripulados são o principal componente de sistemas não tripulados (CLAPPER et al., 2007).

A definição anterior abrange qualquer tipo de veículo autônomo, como os terrestres (chamados de UGV), os de superfícies (chamados de USV) ou os submersos (chamados de UUV). Para que a definição atenda também a VANTs, basta que uma aeronave faça parte do sistema autônomo citado (GUNDLACH, 2012). De fato, para a navegação do VANT é necessário muito mais do que uma aeronave não-tripulada. Elementos chave incluem a aeronave não tripulada em si, carga útil, sistemas de comunicação, estações de controle, equipamentos de lançamento e recuperação e equipamentos de apoio (GUNDLACH, 2012).

Nas aplicações mais tradicionais com VANTs (com menor autonomia), a estação de controle possui participação crucial para o sucesso da missão (tarefa a ser cumprida dentro de um contexto de aplicação) para a qual o VANTs foi designado. Ela é o

centro da operação da missão do VANT e realiza a interface entre homem e máquina (AUSTIN, 2011). A estrutura de uma estação de controle pode variar, podendo ser composta por um simples computador até prédios (ou mais de um em diferentes localidades) com diversos operadores e equipamentos atuando em conjunto para controle da navegação da aeronave autônoma. É por meio da estação de controle que comandos de voo são informados ao VANT. Ela ainda pode planejar a rota e trajetória de navegação; monitorar dados da integridade da aeronave; realizar processamento de dados; realizar o planejamento da missão da aeronave; controlar o lançamento; e, recuperação da aeronave e planejar e monitorar a comunicação com o VANT (AUSTIN, 2011; DALAMAGKIDIS et al., 2011; GUNDLACH, 2012), além de outras funcionalidades adicionais que dependem da complexidade de cada aplicação. A comunicação das estações de controle com o VANT pode ser realizada por rádio-frequência de forma direta e, quando o VANT está fora da linha de visada da estação, por satélites (DALAMAGKIDIS et al., 2011). Na Figura 2.1 encontra-se ilustrado um esquema de comunicação entre estação de controle e o VANT em situações dentro e fora de alcance das linhas de visada de comunicação entre eles.

Figura 2.1 - Esquema de comunicação remota entre estação de controle e o VANT.

Fonte: Gundlach (2012).

À medida que algumas das responsabilidades das estações de controle são transferidas ao VANT, este vai ganhando maior autonomia, ou capacidade de realizar

tomadas de decisão e ações com menor, ou com nenhuma intervenção de um elemento externo. Isso passa a ser mais desejável à medida que um único operador necessita gerenciar mais de um VANT ao mesmo tempo (PROTTI; BARZAN, 2007). Aumentar a autonomia do VANT apresenta vantagens principalmente quando há perda de comunicação com a estação de controle, onde, a partir de então, o VANT pode assumir a tomada de decisão referente à navegação. Além disso, uma maior autonomia pode permitir manobras de contingenciamento em situações de risco pelo próprio VANT, as quais dependem a princípio de comandos da estação de controle que podem sofrer problemas de comunicação afetando o tempo de resposta da entrega de controle. Esse fator é crucial, pois os comandos da estação de controle devem ser enviados a tempo, de modo que o VANT possa manter a sua integridade e a dos elementos em torno da região de sobrevoo (pessoas podem ser atingidas no caso de queda do VANT, por exemplo (MEDEIROS, 2012)).

O nível de autonomia é definido pela quantidade de responsabilidades assumidas pelo VANT. Um VANT alcança um nível de autonomia completo quando não há nenhuma comunicação entre ele e a estação de controle (AUSTIN, 2011). Na Figura 2.2, disponibilizada pelo DoD em 2005, encontra-se ilustrado um gráfico que relaciona uma estimativa do nível de autonomia à certas responsabilidades/capacidades adicionadas ao VANT, no período de 1955 até 2015. O menor nível corresponde à dependência de um controlador remoto para guiar o VANT. À medida que o nível aumenta, maior é a autonomia que o VANT possui, até que seja alcançado o nível em que frotas de VANTs possam se autogerenciar.

Figura 2.2 - Tendência de crescimento de autonomia estimada por ano pelos sistemas de navegação de VANTs.

Os pontos brancos indicam o nível de autonomia de alguns modelos de VANT.

Fonte: Adaptada de Roadmap (2005).

2.1.1 Arquitetura de navegação de VANTs

A arquitetura de navegação possui forte dependência da classe do VANT, da missão e da estrutura de implementação (GUNDLACH, 2012). Baseado nas descrições apresentadas em Boskovic et al. (2004), Goerzen et al. (2010) e Medeiros (2012), um arranjo típico de sistema de controle de VANT encontra-se ilustrado na Figura 2.3. Essa arquitetura típica contém os seguintes módulos:

a) **Tomada de decisão**: Módulo responsável por decidir a abordagem a ser tomada pelo VANT, com o objetivo de cumprir uma missão de navegação como, por exemplo, quais alvos perseguir, o modo de voo (vigilância, reconhecimento, ataque, mapeamento), avaliar os riscos para a aeronave, definir as posições de início e destino de navegação, etc, . Os parâmetros de decisão podem mudar dependendo das mudanças de cenários da navegação (mudanças climáticas, quantidade de combustível, aumento de risco, etc) (GUNDLACH, 2012).

b) **Planejamento de rota**: planeja uma rota viável para navegação sem colisão com os obstáculos do ambiente de navegação. A rota é composta

por um conjunto de *waypoints*, incluindo as posições de início e de destino definidas pelo módulo de tomada de decisão. Para o planejamento da rota, deve o módulo possuir acesso a uma representação computacional do ambiente de navegação. Os dados do ambiente de navegação podem ser capturados em tempo real por sensores acoplados ao VANT ou serem previamente disponibilizados para consulta pelo módulo de planejamento de rota.

c) **Geração de trajetória**: Esse módulo é responsável pela conversão de uma rota em uma trajetória que, segundo Goerzen et al. (2010), é a rota com tempo associada ao longo da mesma, onde se busca adequar a geometria resultante dessa rota às restrições de movimento do VANT. Ajustar a rota planejada resultante no módulo anterior, de forma a suaviza-la para atender restrições de cinemáticas e/ou dinâmicas do VANT, é o principal papel do módulo de planejamento de trajetória. Exemplos de restrições de movimento são velocidade, aceleração, curvatura, etc.

d) **Controle:** Esse módulo é o responsável por enviar comandos aos atuadores da aeronave de forma que a mesma siga ou rastreie a trajetória planejada para a navegação. O controlador busca diminuir os erros entre a trajetória planejada e a executada em tempo real.

e) **VANT**: Representa a aeronave de fato, constituída de partes mecânicas e eletrônicas. A este módulo estão associadas ainda os atuadores, responsáveis por executarem a realização física dos comandos enviados pelo controlador, e os sensores, que coletam dados utilizáveis nos mais diversos objetivos, como imageamento, localização, detecção de obstáculos, comunicação, identificação de falhas, etc.

Figura 2.3 - Arquitetura típica aplicada para a navegação de VANTs.

Fonte: Produção do Autor.

Segundo Goerzen et al. (2010), estruturas semelhantes à apresentada são muito utilizadas pela sua relativa simplicidade de implementação, o que permite escalabilidade para incorporar mais VANTs, adição de novos módulos a arquitetura e adaptabilidade para outras categorias de veículos. É importante frisar que a arquitetura ilustrada foi apresentada considerando um alto nível de abstração. Arquiteturas de aplicações reais usualmente apresentam maior nível de detalhamento em cada um dos módulos apresentados, os quais podem ser constituídos por diversos submódulos. Este trabalho atua no contexto do módulo de planejamento de rota por meio de métodos que buscam encontrar rotas de menor comprimento em menor tempo,

o que contribui para o aumento de sua autonomia para navegação em tempo real. Ainda neste trabalho, métodos que podem ser aplicados ao módulo de geração de trajetória também foram estudados para identificar soluções de planejamento que atendam restrições cinemáticas e dinâmicas do VANT.

2.2 Modelagem do ambiente de navegação

Na busca de soluções de problemas de planejamento de rota para a navegação de VANTs, o ambiente de navegação deve ser modelado a partir do meio real, em uma abstração que colete as informações necessárias e suficientes. Em uma analogia a Sturtevant (2013), a representação computacional do ambiente de navegação consiste em uma abstração ou modelo do ambiente real que viabilize a navegação do veículo autônomo. A eficiência dos métodos aplicados nas próximas etapas de planejamento geralmente depende da representação considerada (TOZOUR, 2003). Nesta seção, são apresentados alguns tipos de representação do ambiente de navegação.

Antes de apresentar estas representações, algumas definições do problema de planejamento de rota são descritas. O primeiro elemento a ser considerado é o *ambiente de navegação*, o qual pode ser representado no espaço ($\mathbb{R}^3$) ou no plano ($\mathbb{R}^2$). O ambiente de navegação considerado neste trabalho é constituído por regiões livres para navegação e obstáculos. Esses últimos são elementos que possuem limites geométricos, os quais definem áreas onde há impossibilidade de realização da navegação do VANT devido à colisão. O VANT é o corpo físico que irá realizar uma trajetória pela rota de referência planejada para o ambiente de navegação. Dependendo de como é especificado o problema, podem ser consideradas ou não outros tipos de restrições, como o tamanho da envergadura do VANT, restrições cinemáticas (velocidade, aceleração, curvatura) e dinâmicas (forças externas, como, por exemplo a ação do vento). Dessa forma, o problema de planejamento de rota consiste em planejar a rota p, que é uma sequência de *waypoints* $q_1, q_2, ..., q_m$ com $m \in \mathbf{N}$, que conectam um *waypoint* inicial (denominado $q_{início}$) do ambiente de navegação a um *waypoint* de chegada (denominado $q_{destino}$) sem que o corpo físico do VANT cruze com os limites dos obstáculos do ambiente de navegação. Uma rota que permita a navegação do VANT sem que o mesmo cruze com esses limites é dita livre de colisão.

O veículo pode ser modelado de duas formas no ambiente de navegação. Pode-se utilizar uma geometria (como uma circunferência) em volta do veículo para representar o corpo do mesmo, o qual pela definição do problema de planejamento de rota não pode interceptar os limites dos obstáculos no ambiente de navegação. Um exemplo é apresentado na Figura 2.4(a). Pode-se ainda representá-lo como um ponto, em que

a área ocupada pelo corpo do veículo é considerada nas envoltórias de segurança dos obstáculos. Um exemplo é apresentado na Figura 2.4(b).

Figura 2.4 - Tipos de representação de VANT em ambiente de navegação.

Representação de VANT como a) área e como b) ponto em ambiente de navegação com obstáculos com envoltórias de segurança.

Fonte: Adaptado de Goerzen et al. (2010).

O ambiente de navegação pode ser subdivido para a distinção das regiões navegáveis (livres de colisão) e não navegáveis (com obstáculos). Usualmente o ambiente de navegação é representado por Q. Ele ainda é dividido em dois subconjuntos: $Q_{obstáculos} = \bigcup_{i=1}^{n_o} O_i$, que representa o conjunto de todos os n_o obstáculos que Q pode conter; $Q_{livre} = Q/Q_{obstáculos}$, representando as regiões navegáveis do ambiente de navegação, i. e., as regiões sem obstáculos e, consequentemente, sem riscos de colisão. A Figura 2.5 ilustra a divisão do ambiente de navegação em Q_{livre} e $Q_{obstáculos}$.

Figura 2.5 - Divisão do ambiente de navegação Q nas regiões Q_{livre} e $Q_{obstáculos}$.

Fonte: Produção do Autor.

É considerado um obstáculo qualquer elemento que um veículo possa vir a colidir ou região que esteja impedida de navegar. Exemplos de obstáculos no mundo real podem ser prédios, aves, terrenos elevados, regiões com clima não adequado para a navegação, outros veículos ou qualquer limite definido pelo qual o VANT esteja impedido de navegar. A classificação de um elemento do ambiente de navegação como obstáculo irá depender da instância do problema de planejamento de rota considerada. Em Wang et al. (2007), por exemplo, existem regiões não desejáveis para navegação do VANT que não são obstáculos físicos. Exemplos dessas regiões não desejáveis seriam áreas restritas ao voo do VANT como aquelas destinadas às rotas de aeronaves tripuladas ou áreas onde a integridade da aeronave esteja ameaçada (regiões hostis).

Os obstáculos podem assumir também comportamentos estáticos ou dinâmicos. Aqueles que possuem posição fixa no ambiente de navegação, permanecendo sempre no mesmo lugar independente do que ocorra neste ambiente, são denominados estáticos. Os obstáculos que se deslocam pelo ambiente de navegação, apresentando diferentes posições em instantes distintos, são denominados dinâmicos. Problemas de planejamento de rota envolvendo obstáculos dinâmicos são mais difíceis de se-

rem solucionados, pois, usualmente é necessário estimar as posições futuras desses obstáculos (LAVALLE, 2006).

Existem duas principais abstrações para representar o ambiente de navegação. Em alguns estudos o ambiente de navegação é representado como uma grade celular (LATOMBE, 2012), onde cada célula discretiza uma porção do espaço físico real por onde o VANT navega. A grade consiste em um conjunto de células não sobrepostas (formas geométricas que delimitam uma unidade da grade), geralmente de mesmo tamanho e formato, adjacentes umas às outras, onde cada célula representa uma porção do espaço em uma resolução. Caso um obstáculo não ocupe completamente uma das células, a mesma, por inteira, é considerado não livre de colisão. Outra representação do ambiente de navegação considera-o um espaço contínuo, com os obstáculos representados por formas geométricas. Na Figura 2.6 encontram-se ilustrados exemplos das duas representações citadas.

Figura 2.6 - Tipos de representações de ambientes de navegação e obstáculos.

a) Obstáculos em ambiente de navegação com representação contínua e b) com representação em grade.

Fonte: Produção do Autor.

Em muitos problemas de planejamento, o ambiente de navegação do veículo é desconhecido a priori. Nesses casos, a detecção do ambiente de navegação (identificação dos obstáculos) pode ser realizada em tempo real de navegação por meio de sensores (sonar, imageador, laser, etc) acoplados ao veículo. Revisões sobre o uso de sensores em navegação de veículos autônomos podem ser encontradas na literatura

(DADKHAH; METTLER, 2012; LUMELSKY, 1987). Já em outros problemas de planejamento, os dados do ambiente de navegação são conhecidos, e podem ser previamente disponibilizadas ao controlador remoto ou ao computador de bordo do VANT.

2.2.1 Modelo de ambiente de navegação adotado

Nesta tese, considera-se que os ambientes de navegação são conhecidos. Eles possuem representação contínua (VANT pode assumir qualquer posição livre de colisão, como ilustrado na Figura 2.6(a)) e bidimensional, pois, a altitude de navegação do VANT é considerada constante. Todos os obstáculos considerados são estáticos, e o veículo é representado como um ponto no ambiente de navegação. Dessa forma, todos os obstáculos considerados possuem uma envoltória de segurança, como especificado na Figura 2.4(b). Outra consideração feita é que um VANT pode navegar livre de colisão pelas bordas e pelos vértices de cada envoltória de segurança de obstáculos do ambiente de navegação.

3 MÉTODOS DE PLANEJAMENTO DE ROTA BASEADOS EM AMOSTRAGEM

Neste capítulo é introduzida a classe de algoritmos de planejamento de rota baseados em amostragem. Nesses algoritmos, a rota é construída a partir de amostras coletadas do ambiente de navegação até que a mesma seja planejada. Nesta tese, uma amostra é uma posição espacial no plano cartesiano bidimensional contínuo, sendo definida pelas coordenadas (x, y).

A principal característica de métodos baseados em amostragem é a isenção da representação explicita do ambiente de navegação (KAVRAKI; LATOMBE, 1994; KAVRAKI et al., 1996; LAVALLE et al., 2001). Isso significa que o ambiente de navegação não faz parte da estrutura de dados utilizada pelo algoritmo. Até o desenvolvimento dessa classe de algoritmos, a representação do ambiente de navegação era incorporada à estrutura dos algoritmos de planejamento de rotas, acarretando em alto custo computacional. Algoritmos como decomposição por células (BROOKS; LOZANO-PEREZ, 1985), campos potenciais (KHATIB, 1986) e os apresentados em Canny (1988), Lozano-Pérez e Wesley (1979), Schwartz e Sharir (1983), utilizam a representação explícita do ambiente de navegação. Entretanto, esses algoritmos podem gerar excessivo esforço computacional, principalmente em ambientes de navegação com grande quantidade de obstáculos (KARAMAN; FRAZZOLI, 2011).

Para evitar a representação explícita do ambiente de navegação, algoritmos baseados em amostragem utilizam um esquema de coleta de amostras (LAVALLE, 2006). Esse esquema utiliza um módulo de detecção de colisão com os obstáculos. Dessa forma, o algoritmo baseado em amostragem não mantém relação explícita com o ambiente de navegação, sendo decidido, inicialmente, se uma amostra é válida ou não para uma solução do algoritmo pelo módulo de detecção de colisão. Esses algoritmos também utilizam técnicas de busca discreta para selecionar por onde ocorrerá a expansão da rota, o que geralmente corresponde à operações de busca de vizinho mais próximo. A Figura 3.1 ilustra a estrutura usualmente compartilhada por algoritmos baseados em amostragem que resulta em um planejamento de rota mais eficiente do que aqueles com representação explícita do ambiente de navegação.

Com base em critérios definidos por cada método de planejamento, uma amostra pode ser considerada válida ou inválida para a construção da rota. Amostras válidas são aquelas coletadas na região navegável do ambiente de navegação (região sem obstáculos). As amostras inválidas são coletadas nas regiões não navegáveis (região contendo obstáculos). Exemplos dessas amostras estão ilustrados na Figura 3.2.

Figura 3.1 - Estrutura usualmente utilizada para compor algoritmos de planejamento de rota baseados em amostragem.

O teste de colisão é realizado por um módulo à parte, o qual possui acesso à representação do ambiente de navegação.

Fonte: Adaptado de LaValle (2006).

Figura 3.2 - Classificação de amostras.

Exemplo de amostra válida (em ciano) e amostra inválida (em vermelho) em ambiente de navegação com envoltórias de segurança de obstáculos (em azul).

Fonte: Produção do Autor.

Entretanto, se mais restrições forem adicionadas ao critério de validação de amostras, situações adicionais às que foram citadas podem surgir. Por exemplo, é possível planejar a rota considerando as restrições cinemáticas do veículo durante a coleta de uma amostra. Neste tipo de restrição, se uma amostra coletada gerar uma rota que não atenda a critérios de velocidade, aceleração, curvatura, etc, do veículo, a mesma pode ser rejeitada, mesmo estando posicionada na região navegável do ambiente de navegação (trabalhos que consideram restrição cinemática durante o planejamento podem ser verificados em Frazzoli (2001), Hsu et al. (2002), LaValle et al. (2001)). Cenários em que as restrições cinemáticas de um VANT são consideradas durante o planejamento da rota são discutidos com maiores detalhes na Seção 5. Adicionalmente, um algoritmo de planejamento que considera essas restrições é utilizado e descrito nesta tese na referida seção.

Os principais algoritmos de planejamento de rota baseado em amostragem são *Probabilistic Roadmaps* (KAVRAKI et al., 1996), RRT (do inglês *Rapidly-exploring Random Tree*) (LAVALLE, 1998), *Expansive Space Trees (EST)* (HSU et al., 1997), *Ariadne's clew* (MAZER et al., 1998) e *Fast Marching Tree (FMT)* (JANSON et al., 2015), onde os dois primeiros foram os mais estudados nos últimos anos. Uma revisão geral de métodos baseados em amostragem é dada em (ELBANHAWI; SIMIC, 2014).

3.1 Árvore aleatória de exploração rápida (RRT)

O algoritmo RRT (LAVALLE, 1998) é uma das técnicas baseadas em amostragem que podem ser aplicadas ao planejamento de rota de VANTs. O algoritmo permite o planejamento de rotas por meio da exploração aleatória de ambientes de navegação com obstáculos. O RRT consiste em expandir iterativamente uma estrutura de dados no formato de árvore pelo ambiente de navegação a partir do ponto inicial (nó raiz dessa árvore) de navegação até que um ponto final (destino da navegação) previamente definido seja encontrado. Para expandir a árvore, o algoritmo coleta a cada iteração uma amostra do ambiente de navegação aleatoriamente seguindo uma distribuição uniforme. A cada nova amostra do ambiente de navegação coletada, um novo galho (aresta) da árvore pode ser formado. A amostra coletada se torna uma nova folha (nó) na árvore se ela permitir formar um novo galho livre de colisão (que não intercepta os limites dos obstáculos do ambiente de navegação). A árvore sofre expansões até que um de seus galhos alcancem a posição final de navegação do VANT, uma quantidade de iterações seja alcançada, ou um tempo limite de planejamento seja atingido.

O algoritmo RRT encontra-se descrito no Algoritmo 1. A descrição do algoritmo

segue a notação definida na Seção 2.2 para identificar as regiões navegáveis e não navegáveis do ambiente de navegação. Os principais procedimentos executados pelo algoritmo são definidos a seguir:

a) **AMOSTRAGEM_ALEATÓRIA**: gera uma posição aleatória $q_{aleatório}$ no ambiente de navegação.

b) **NÓ_PRÓXIMO**: busca o nó mais próximo $q_{próximo}$ de $q_{aleatório}$ em G.

c) **NOVO_NÓ**: gera um novo nó q_{novo} sobre o segmento de reta $\overline{q_{próximo}q_{aleatório}}$ a uma distância Δq.

d) **ESTENDE**: expande a árvore, atribuindo $q_{próximo}$ como nó antecessor/pai de q_{novo}, adiciona $\overline{q_{próximo}q_{novo}}$ ao conjunto de aresta E e adiciona q_{novo} ao conjunto de nós V, onde $E \in G$ e $V \in G$.

e) **ROTA**: identifica todos os *waypoints* da rota planejada, armazenando-os em R, que é uma estrutura de dados do tipo pilha, a partir de $q_{destino}$ até $q_{início}$, utilizando a informação do nó antecessor de cada nó da árvore.

Algorithm 1 Algoritmo RRT

```
procedure RRT(Q, q_início, q_destino, Δq, l_d, n)
    G ← {q_início}
    R ← {}
    s ← 0
    i ← 0
    while (s = 0) e (i ≤ n) do
        i ← i + 1
        q_aleatório ← AMOSTRAGEM_ALEATÓRIA(Q)
        q_próximo ← NÓ_PRÓXIMO(q_aleatório, G)
        q_novo ← NOVO_NÓ(q_próximo, q_aleatório, Δ_q)
        if q_próximo q_novo não intercepta Q_obstáculos then
            ESTENDE(G, q_próximo, q_novo)
            if d(q_novo, q_destino) ≤ l_d e (q_novo q_destino) não intercepta Q_obstáculos) then
                ESTENDE(G, q_novo, q_destino)
                s ← 1
                R ← ROTA(q_início, q_destino)
```

Seja uma árvore $G = (V, E)$, onde V é o seu conjunto de nós e E o seu conjunto de arestas. A árvore é inicializada com um nó $q_{início} \in Q_{livre}$, que representa a

posição inicial de navegação. A partir de $q_{início}$, G é expandida iterativamente. A cada iteração, uma nova amostra $q_{aleatório}$ é coletada aleatoriamente do ambiente de navegação por $AMOSTRAGEM_ALEATORIA$ (Algoritmo 1, linha 8). Novas amostras são coletadas até que $i > n$ ou $s = 1$ (indica que uma rota foi planejada).

A amostra $q_{aleatório}$ tem a função de indicar a direção da expansão de G e de determinar por qual nó da árvore essa expansão deve ocorrer. G irá expandir pelo nó de V espacialmente mais próximo de $q_{aleatório}$. Esse nó é encontrado pela operação $NÓ_PRÓXIMO$ (Algoritmo 1, linha 9), o qual passa a ser referenciado por $q_{próximo}$. Em seguida, G é expandida em direção a $q_{aleatório}$ a uma distância constante Δq, definindo assim uma nova amostra q_{novo} a ser adicionada à árvore. A amostra q_{novo} é gerada pela operação $NOVO_NÓ$ (Algoritmo 1, linha 10).

A nova amostra coletada q_{novo} forma junto com o nó $q_{próximo}$ o segmento de reta $\overline{q_{próximo}q_{novo}}$. Para que esse segmento possa expandir G e q_{novo} possa ser adicionada à árvore e se tornar um nó da mesma, ele deve ser livre de colisão. Se $\overline{q_{próximo}q_{novo}}$ for livre de colisão, o mesmo é adicionado a E e q_{novo} é adicionada como novo nó a V pela operação $ESTENDE$ (Algoritmo 1, linha 12). Dessa forma, $q_{próximo}$ torna-se o nó antecessor/pai de q_{novo} e este o nó sucessor/filho de $q_{próximo}$. Essas etapas encontram-se ilustradas na Figura 3.3. Caso $\overline{q_{próximo}q_{novo}}$ estiver em colisão com $Q_{obstáculos}$, uma nova amostra aleatória $q_{aleatório}$ é coletada para indicar uma nova direção de expansão da árvore.

Figura 3.3 - Descrição da etapa de extensão do RRT.

a) Coleta de uma amostra aleatória $q_{aleatório}$ no ambiente de navegação e busca do nó mais próximo (busca discreta) na árvore $q_{próximo}$. b) Geração de uma nova amostra q_{novo} em direção de $q_{aleatório}$. c) Se $\overline{q_{próximo}q_{novo}}$ for livre de colisão, q_{novo} é adicionada a árvore e uma nova aresta é formada.

Fonte: Produção do Autor.

Essa dinâmica entre coletar uma amostra aleatória e expandir G é executada até que eventualmente um nó q_{novo} seja gerado dentro da região associada à posição final de navegação indicada por $q_{destino}$. Se q_{novo} for livre de colisão e estiver no raio de alcance l_d de $q_{destino}$, é verificado se o segmento formado $\overline{q_{novo}q_{destino}}$ é livre de colisão. Caso afirmativo, o segmento estende a rota adicionando-o à árvore (Algoritmo 1, linha 14) formando então uma rota entre $q_{início}$ e $q_{destino}$. Esse processo encontra-se ilustrado na Figura 3.4.

Figura 3.4 - Planejamento de uma rota pelo algoritmo RRT.

a) Um nó q_{novo} encontra-se posicionado dentro do raio de alcance l_d de $q_{destino}$. b) É verificado se o segmento $\overline{q_{novo}q_{destino}}$ é livre de colisão. c) Em caso positivo, o segmento é adicionado à árvore e a rota R que conecta $q_{início}$ à $q_{destino}$ é planejada.

Fonte: Produção do Autor.

Quando uma rota é planejada pelo RRT, os nós que ligam $q_{início}$ à $q_{destino}$ são armazenados na pilha R. Isso é realizado transferindo-se cada nó predecessor de $q_{destino}$ até $q_{início}$ pela operação $ROTA$ (Algoritmo 1, linha 16). Dessa forma, o VANT obtêm a informação sobre os nós que representam os *waypoints* de navegação para o seu deslocamento de $q_{início}$ até $q_{destino}$.

Várias extensões do algoritmo RRT foram publicadas ao longo dos anos: *Adaptative RRT Based on Dynamic Step (DRRT)* (LIN; ZHANG, 2015), *RRT Star (RRT*)* (KARAMAN; FRAZZOLI, 2010), *Fast RRT* (MA et al., 2015), *Real-time Closed-loop Rapidly Exploring Random Trees (CL-RRT)* (FRAZZOLI et al., 2002) e suas variações *Closed-loop Random Belief Trees (CL-RBT)* (LI et al., 2014), *Resolution Complete Rapidly-Exploring Random Tree (RC-RRT)* (CHENG; LAVALLE, 2001), *Execution extended Rapidly Exploring Random Tree (ERRT)* (BRUCE; VELOSO, 2006), *Chance-*

Constraint Rapidly Exploring Random Tree (CC-RRT) (LUDERS et al., 2010), *CC-RRT*-D* (LIU; ANG, 2014), *RRT-Connect* (KUFFNER; LAVALLE, 2000), entre outras. Portanto, a variabilidade de estudos focados em melhorar ou adaptar algum aspecto do algoritmo RRT é notável, o que pode ser considerado um indicador da sua importância para a área.

3.1.1 Teste de colisão

O teste de colisão é um dos processos com maior custo computacional em planejamento de rota (LAVALLE, 2006). Nesta tese, o teste de colisão consiste na detecção da intersecção entre a aresta formada durante a extensão do RRT e as arestas dos obstáculos. Como suporte às operações geométricas necessárias ao teste de colisão, foram utilizadas as funções disponibilizadas pela biblioteca de geometria computacional CGAL (CGAL, 2008). A estratégia de teste de colisão descrita a seguir é executada pelo Algoritmo 1 nas linhas 11 e 13.

Como definido na Seção 2.2, $Q_{obstáculos}$ corresponde ao conjunto de todos os obstáculos do ambiente de navegação Q. Seja o conjunto $Q_{obstáculos} = \{O_1, O_2, ..., O_{n_o}\}$, com $n_o \in \mathbb{N}$. Cada um de seus elementos corresponde a um obstáculo do ambiente de navegação. Para cada O_i, com $i \leq n_o$, existem j arestas que formam o conjunto $A_i = \{a_1, a_2, ..., a_j\}$, com $j \in \mathbb{N}$, e t vértices que formam o conjunto $V_i = \{v_1, v_2, ..., v_t\}$, com $t \in \mathbb{N}$. Com base nas definições dadas, o teste de colisão é realizado em duas etapas, descritas a seguir e ilustradas na Figura 3.5.

Figura 3.5 - Exemplos de situações identificadas em $Q_{obstáculos}$ no teste de colisão.

As aresta em vermelho são resultados positivos para o teste de colisão e não poderão ser utilizadas para expandir a árvore. As arestas e pontos na cor verde escuro são livres de colisão pois se encontram sobre uma aresta ou vértice de uma envoltória convexa de obstáculo. Pontos em vermelho são amostras inválidas, pois, estão localizadas em $Q_{obstáculos}$. O ponto em cíano é uma amostra válida, entretanto, gera uma aresta com colisão em $Q_{obstáculos}$.

Fonte: Produção do Autor.

Na primeira etapa, é verificado se uma nova amostra q_{novo} encontra-se localizada em Q_{livre}. Para isso, cada obstáculo $O_i \subset Q_{obstáculos}$ é consultado. Se $q_{novo} \in O_i$, então a amostra é considerada inválida. Caso contrário, q_{novo} é considerado válido e é dado prosseguimento para a segunda etapa. É importante mencionar que, neste trabalho, são consideradas as envoltórias de segurança de obstáculos. Assim, amostras podem ser coletadas nas bordas e vértices das envoltórias de segurança dos obstáculos.

Na segunda etapa é verificado se a aresta $\overline{q_{novo}q_{próximo}}$ intercepta algum obstáculo de $Q_{obstáculo}$. Para cada O_i, se $\overline{q_{novo}q_{próximo}}$ estiver em intersecção com qualquer a_j de A_i, então a aresta não é livre de colisão e não poderá ser adicionada à árvore. Caso contrário, q_{novo} e $\overline{q_{novo}q_{próximo}}$ são adicionadas a árvore (operação $ESTENDE$ da linha 12 do Algoritmo 1) finalizando o seu processo de extensão.

O teste de colisão descrito anteriormente também é executado quando q_{novo} está a uma distância l_d de $q_{destino}$ (linha 13 do Algoritmo 1). Essa operação é um pré-

requisito para conectar a árvore ao destino do planejamento e finalizar a rota para navegação. A operação da linha 13 é executada exatamente como descrito na segunda etapa do teste de colisão, apenas trocando $q_{próximo}$ por q_{novo}. A primeira etapa não é executada, pois é pré requisito para o início de planejamento pelo algoritmo RRT que $q_{destino} \in Q_{livre}$, caso contrário não seria possível planejar uma rota.

3.1.2 Características e limitações

No algoritmo RRT, a amostragem aleatória segue uma distribuição uniforme, o que implica que cada posição do ambiente de navegação possui a mesma probabilidade de ser coletada. Assim, quanto maior o número de amostras coletadas no ambiente de navegação, maior a chance do RRT encontrar uma rota livre de colisão, caso ela exista (LINDEMANN; LAVALLE, 2005). Devido à essa característica, o RRT é um algoritmo *probabilisticamente completo* (KUFFNER; LAVALLE, 2000).

Outra característica do RRT é que as amostras tendem a ser coletadas próximas aos nós da árvore com maiores regiões de Voronoi associadas. Uma região de Voronoi representa a área em torno de um nó em que todos os seus pontos possuem distância menor a esse nó do que aos demais. A união de todas as regiões de Voronoi forma a estrutura conhecida como Diagrama de Voronoi (AURENHAMMER, 1991). Na Figura 3.6 encontra-se ilustrado um diagrama de Voronoi.

Figura 3.6 - Exemplo de diagrama de Voronoi dos nós da árvore do algoritmo RRT.

O nó em amarelo possui maior chance de ser selecionado para expansão por estar associado a maior região de Voronoi.

Fonte: Produção do Autor.

Considerando a distribuição uniforme do processo aleatório de amostragem, os nós com maiores regiões de Voronoi associadas possuem uma chance maior de serem selecionados para a expansão do RRT. A essa caraterística é dada o nome de *orientação de Voronoi* (do inglês *Voronoi Bias*) (LINDEMANN; LAVALLE, 2004). Em cenários com obstáculos que formam regiões sem saída, a orientação de Voronoi tende a atrasar a expansão do RRT devido a uma maior frequência em gerar arestas em situação de colisão, como ilustrado no exemplo da Figura 3.7. Muitos trabalhos acadêmicos se dedicam a mitigar esse problema como, por exemplo, em Lindemann e LaValle (2004), Park et al. (2014) e Qureshi e Ayaz (2016).

Figura 3.7 - Efeito da orientação de Voronoi no crescimento da árvore do RRT.

A área em vermelho representa a região de Voronoi associada ao nó com maior chance de expandir a árvore do algoritmo RRT. Entretanto, o nó encontra-se cercado por um obstáculo, impedindo uma rápida exploração do ambiente de navegação por ele. Portanto, ele pode não ser um bom candidato para a expansão da árvore.

Fonte: Adaptado de Lindemann e LaValle (2004).

A velocidade com que o algoritmo RRT planeja uma rota também é dependente do tamanho do passo de expansão Δq. Dessa forma, quanto maior for o tamanho de Δq, mais acelerada será a expansão da árvore. Entretanto, a qualidade da exploração do ambiente de navegação fica prejudicada, dependendo da distribuição espacial dos obstáculos. Isso fica mais evidente quando um ambiente de navegação contém muitos obstáculos, onde valores de Δq muito altos podem levar a um grande número de

colisões até que uma amostra válida seja coletada, o que pode aumentar o tempo de planejamento pelo RRT. Caso seja utilizado um tamanho menor de Δq, o ambiente de navegação passa a ser explorado de forma mais eficiente, porém uma quantidade maior de iterações será necessário para planejar a rota. Os efeitos da escolha do tamanho de Δq no desempenho do RRT são investigados em Elbanhawi e Simic (2014). Exemplos de estudos que buscam definir o melhor valor para Δq podem ser verificados em Kuffner e LaValle (2000), McCourt et al. (2016), Wang e Meng (2016) e An et al. (2018). Neste trabalho, o valor de Δq é definido por um valor de porcentagem do maior lado de um ambiente de navegação considerado em cada problema de planejamento de rota, porém, não faz parte do escopo do mesmo o estudo da estimativa do valor de Δq mais adequado para cada ambiente de navegação.

Em Karaman e Frazzoli (2011) uma análise de complexidade do algoritmo RRT é dada. Segundo esse trabalho, o RRT possui complexidade assintótica igual a $O(n\,log\,n)$, com n correspondendo ao número de nós da árvore. O valor de $log\,n$ corresponde à complexidade do procedimento de busca de nó mais próximo executado em $NÓ_PRÓXIMO$. Essa operação pode ser obtida com algoritmos de busca de vizinho mais próximo como *k-d-tree* (que possui complexidade $O(log\,n)$ no caso médio) ou técnicas de busca aproximada como a apresentada em Arya et al. (1998). Por simplicidade, um algoritmo, do tipo força bruta para a busca do nó mais próximo é adotado por todos os algoritmos de planejamento de rota considerados neste trabalho. Assim, a complexidade da versão do algoritmo RRT adotada neste trabalho é $O(n^2)$.

O trabalho desenvolvido em Karaman e Frazzoli (2010) ainda aponta outra deficiência apresentada pelo RRT. Apesar de rotas serem rapidamente planejadas pelo algoritmo, mesmo que infinitas amostras sejam coletadas do ambiente de navegação, não existe garantia que a melhor rota, em termos do seu comprimento, seja encontrada. Em Karaman e Frazzoli (2010) foi proposto então o algoritmo RRT*, uma versão do RRT que possui *otimalidade assintótica*. Isso significa que à medida que o número de nós da árvore tende ao infinito, a probabilidade de que a rota ótima, para um dado ambiente de navegação, seja encontrada se aproxima de um. O algoritmo RRT* foi utilizado como base para o desenvolvimento do algoritmo proposto nesta tese. Assim, ele é descrito na Seção 3.2.

3.2 RRT com otimalidade assintótica (RRT*)

O algoritmo RRT* (lê-se RRT Estrela) é uma variante do RRT introduzida em Karaman e Frazzoli (2010). Neste algoritmo, a rota é otimizada baseada em um

valor de custo de navegação associada a cada par de nós da árvore. Quanto maior um custo associado a uma aresta, menos indicada é a navegação de um VANT por ela. No RRT*, as arestas entre pares de nós são recombinadas cada vez que um novo nó é adicionado à árvore, tendo como critério a minimização do custo de navegação de cada nó da árvore até a sua raiz $q_{início}$. Neste trabalho, o valor de custo de uma aresta é o comprimento da distância entre os dois nós que a definem. O custo total relacionado a um nó qualquer da árvore é igual à soma dos comprimentos das arestas que o ligam até a raiz da árvore $q_{início}$. O custo total de uma rota planejada é igual à soma dos comprimentos das arestas que ligam $q_{destino}$ até $q_{início}$.

A principal diferença entre o algoritmo RRT* e o algoritmo RRT está no modo como se dá a expansão da árvore. A principal modificação apresentada pelo RRT* é a substituição da operação $ESTENDE$ do RRT pela operação $RRT_ESTRELA_ESTENDE$. Essa operação e as demais introduzidas no algoritmo RRT* são descritas a seguir:

a) **VIZINHANÇA**: retorna o conjunto $Q_{vizinhança} \in G$ dado por todos os nós que estão à uma distância menor que $\beta(log(n)/n)^{1/2}$ de q_{novo}, em que n é a quantidade atual de nós em G;

b) **RRT_ESTRELA_ESTENDE**: estende a árvore selecionando o nó vizinho $q_v \in Q_{vizinhança}$, que permita formar a rota com o menor comprimento (custo), em comparação com os nós de $Q_{vizinhança}$, de q_{novo} a $q_{início}$, passando por q_v;

c) **RECONECTA**: q_{novo} é conectado como antecessor/pai dos nós vizinhos de $Q_{vizinhança}$ caso formem com ele uma rota de menor comprimento até o nó $q_{início}$.

O custo de uma rota entre dois nós q_a e q_b de G é definido por:

$$CUSTO(q_a, q_b) = \sum_{i=1}^{m-1} c(q_i, q_{i+1}) \tag{3.1}$$

onde: m é a quantidade de nós da rota definida pelos nós q_a e q_b; q_i é o i-ésimo nó da rota entre q_a e q_b; q_{i+1} é o nó predecessor de q_i; q_1 é o último nó q_b da rota; q_m é o primeiro nó q_a da rota; e $c(q_i, q_{i+1})$ é o custo/comprimento de cada par de nós consecutivos (q_i, q_{i+1}).

Algorithm 2 Procedimento de extensão do algoritmo RRT*.

1: **procedure** RRT_ESTRELA_ESTENDE($G, q_{próximo}, q_{novo}$)
2: $q_{min} \leftarrow q_{próximo}$
3: $Q_{vizinhança} \leftarrow VIZINHANÇA(G, q_{novo})$
4: **for** $q_v \in Q_{vizinhança}$ **do**
5: **if** $\overline{q_v q_{novo}}$ não intercepta $Q_{obstáculo}$ **then**
6: $c' \leftarrow CUSTO(q_{início}, q_{novo}) + c(q_{novo}, q_v)$
7: **if** $c' < CUSTO(q_{início}, q_{min}) + c(q_{novo}, q_{min})$ **then**
8: $q_{min} \leftarrow q_v$
9: $ESTENDE(G, q_{min}, q_{novo})$
10: $RECONECTA(Q_{vizinhança}, q_{novo})$
11: **retorna**

Para cada q_{novo}, o algoritmo RRT* verifica se não há obstáculos entre ele e o nó $q_{próximo}$ selecionado. Se isso não ocorrer, os nós vizinhos de q_{novo}, dentro do intervalo do raio r_n, definido por:

$$r_n = \beta(log(n)/n)^{1/2} \tag{3.2}$$

serão selecionados e atribuídos a $Q_{vizinhos}$. Se $CUSTO(q_{início}, q_v) + c(q_{novo}, q_v) < CUSTO(q_{início}, q_{próximo}) + c(q_{novo}, q_{próximo})$, então este $q_v \in Q_{vizinhos}$ garante o planejamento de rota livre de colisão para $q_{início}$ menor que a rota que conecta $q_{próximo}$ com q_{novo}. Assim, cada nó q_v de $Q_{vizinhos}$ é analisado para encontrar aquele (q_{min}) que permite a rota mais curta entre $q_{início}$ e q_{novo} . Este nó q_{min} é então conectado a q_{novo}, de modo que q_{min} seja o predecessor de q_{novo}. Esse processo é definido no procedimento $RRT_ESTRELA_ESTENDE$, conforme descrito no Algoritmo 2. Na Figura 3.8 encontram-se ilustradas as etapas do processo de seleção de q_{min}.

Algorithm 3 Procedimento de reconexão de arestas do algoritmo RRT*

1: **procedure** RECONECTA($Q_{vizinhança}, q_{novo}$)
2: **for** $q_v \in Q_{vizinhança}$ **do**
3: **if** $\overline{q_v q_{novo}}$ não intercepta $Q_{obstáculos}$ **then**
4: $c' \leftarrow CUSTO(q_{início}, q_{novo}) + c(q_v, q_{novo})$
5: **if** $c' < CUSTO(q_{início}, q_v)$ **then**
6: $q_{antecessor} \leftarrow ANTECESSOR(q_v)$
7: $G.REMOVE_ANTECESSOR(q_v, q_{antecessor})$
8: $G.ADICIONA_ANTECESSOR(q_v, q_{novo})$

Figura 3.8 - Etapas de extensão da árvore no algoritmo RRT*.

Fonte: Produção do Autor.

Depois de definir a rota mais curta de $q_{início}$ para q_{novo}, passando por q_{min}, verifica-se a possibilidade de reduzir o comprimento de rota dos vizinhos restantes de $Q_{vizinhos}$. Para fazer isso, é analisado o tamanho de rota entre $q_{início}$ para cada q_v, passando por q_{novo}. Se $CUSTO(q_{início}, q_{novo}) + c(q_v, q_{novo}) < CUSTO(q_{início}, q_v)$, então o predecessor de q_v é alterado para q_{novo}. Desta forma, se possível, a rota é otimizada para todos os nós pertencentes à $Q_{vizinhos}$. Este processo é executado pelo procedimento $RECONECTA$, descrito no Algoritmo 3. Um exemplo de reconexão dos nós de $Q_{vizinhos}$ encontra-se ilustrado na Figura 3.9.

Figura 3.9 - Etapa de reconexão de arestas para redução de comprimento da rota dos nós de uma vizinhança até $q_{início}$.

Fonte: Produção do Autor.

3.2.1 Exemplo de aplicação do algoritmo RRT* ao planejamento de rotas

Na Figura 3.10 encontra-se ilustrada a expansão da árvore do algoritmo RRT*. Nas figuras, a árvore resultante do planejamento de rota pelo algoritmo RRT* apresenta um padrão em que as arestas tendem a convergir para o ponto inicial de planejamento $q_{início}$. Dessa forma, entende-se que a tendência das arestas da árvore é a de se posicionarem uma em relação as outras de maneira a formarem retas a partir de qualquer nó para a posição inicial de navegação $q_{início}$. Sabendo que a menor distância entre dois pontos quaisquer é um segmento de reta entre ambos, e que o algoritmo RRT* reconecta as arestas dos nós de uma vizinhança baseada na formação da menor rota possível até $q_{início}$, a obtenção dessa estrutura de árvore seria de fato a esperada. Esse padrão só é interrompido devido à existência de obstáculos.

Na Figura 3.11, está ilustrado um exemplo de árvore do RRT* com diferentes quantidades de nós e a região de vizinhança correspondente à $r(n)$, definida na Equação 3.2. Nota-se que a vizinhança (representada como um circunferência de cor roxa), à medida que a quantidade de nós da árvore aumenta, diminui correspondentemente. Como especificado em Karaman e Frazzoli (2010), essa redução busca diminuir a quantidade de nós a serem verificados pelo teste de colisão em $RRT_ESTRELA_ESTENDE$ (Algoritmo 2). Quando a quantidade de nós da árvore se tornar muito grande, caso o raio de vizinhança forme uma região muito vasta, uma grande quantidade de nós deverá ser submetida ao teste de colisão, o que aumentará o custo computacional do RRT*. Por outro lado, se o tamanho do raio da vizinhança for reduzido desde o início do planejamento, uma menor quantidade de nós será selecionada para formação de vizinhança, e assim, a otimização do comprimento da rota será prejudicada. Dessa forma, o valor da constante na Equação 3.2 deve ser definido buscando-se um equilíbrio entre eficiência na redução do comprimento da rota e desempenho computacional (KARAMAN; FRAZZOLI, 2010).

A Figura 3.12 apresenta gráficos que refletem em números o comportamento dos elementos associados ao algoritmo RRT*. Na Figura 3.12(a), encontra-se ilustrado o decaimento do comprimento da rota a medida que o número de iterações (e consequentemente o número de nós) no algoritmo RRT* aumenta. Na Figura 3.12(b) encontra-se ilustrado um gráfico com o decaimento do tamanho do raio $r(n)$ para formação de vizinhança à medida que a quantidade de nós da árvore aumenta para valor de $\beta = 650$. Os gráficos e figuras apresentados nesta seção, portanto, caracterizam o comportamento esperado de um algoritmo baseado no RRT*.

Figura 3.10 - Exemplo de planejamento de rota pelo algoritmo RRT* em instantes distintos.

Nota-se que devido ao processo de reconexão das arestas em uma vizinhança, a árvore adquire uma estrutura em que as conexões entre as arestas tendem a formar retas que convergem em direção a $q_{início}$ (representado pelo ponto na cor verde).

Fonte: Produção do Autor.

Figura 3.11 - Diminuição do tamanho da região de vizinhança (círculo na cor roxa) com o aumento do número de nós da árvore no algoritmo RRT*.

Fonte: Produção do Autor.

Figura 3.12 - Gráficos ilustrando a) a diminuição do comprimento da rota com o aumento do número de iterações e b) a redução do raio de formação de vizinhança com o aumento da quantidade de nós na árvore no algoritmo RRT*.

Fonte: Produção do Autor.

3.2.2 Características e limitações

A principal característica do algoritmo RRT* é a adição de otimalidade assintótica ao planejamento de rota. Isso significa que existe garantia teórica, demonstrada em Karaman e Frazzoli (2011), de que a rota ótima (aquela de menor comprimento possível) será encontrada à medida que o número de amostras coletadas tendem ao infinito. Entretanto, na prática, devido à algumas desvantagens computacionais apresentadas pelo algoritmo, deve haver um tempo de planejamento, em muitos casos, impraticável, para que essa rota ótima possa ser obtida. Essas desvantagens serão detalhadas a seguir.

Como mencionado em Ma et al. (2015), apesar de o algoritmo RRT* retornar como resultado uma rota de menor custo/comprimento do que o algoritmo RRT e ser capaz de obter a rota ótima para um dado ambiente de navegação, o algoritmo não consegue convergir a uma taxa de processamento rápida o suficiente para a rota de menor comprimento em alguns casos. Como no algoritmo há uma alta quantidade de pesquisas de vizinho mais próximo envolvidas que realizam teste de colisão, dependendo da complexidade do ambiente de navegação (por exemplo, com muitos obstáculos), o tempo de planejamento pode ser elevado consideravelmente. Portanto, o algoritmo RRT* pode não ser adequado para aplicações reais pela lenta convergência para a solução ótima que pode apresentar (NASIR et al., 2013).

Esse problema já foi apontado por diversos estudos na literatura, como, por exemplo

em Akgun e Stilman (2011), Nasir et al. (2013), Elbanhawi e Simic (2014) e Qureshi e Ayaz (2016). Essa ineficiência do algoritmo motivou o desenvolvimento de diversos estudos que buscam reduzir o seu tempo de planejamento por meio da otimização de suas operações. Em Akgun e Stilman (2011), por exemplo, uma estratégia de rejeição de amostras baseada em uma heurística é introduzida ao RRT* para reduzir o tempo de amostragem do algoritmo. Já em Karaman et al. (2011), a operação de reconexão das arestas do RRT* é otimizada para reduzir o tempo computacional do planejamento. Em Gammell et al. (2014), o processo de amostragem do RRT* é modificado para acelerar o tempo de planejamento por meio da redução do ambiente de navegação para a coleta de amostras com maiores chances de pertencerem à rota de menor comprimento. A contribuição desta tese encontra-se inserida nesta última categoria de otimização do RRT*, onde o processo de amostragem é modificado para que a rota ótima seja obtida mais rapidamente. Portanto, abordagens de otimização do tempo de planejamento por meio da modificação do processo de amostragem não só do RRT*, mas também do RRT, serão exploradas com maiores detalhes na Seção 3.3.

3.3 Amostragem não uniforme/informada em algoritmos baseados no RRT

Nos algoritmos RRT e RRT*, a amostragem aleatória segue uma distribuição espacial uniforme para coletar novas amostras. Assim, qualquer posição do ambiente de navegação possui a mesma probabilidade de ser coletada. No entanto, alguns autores argumentam que a inserção de um comportamento não uniforme/informado no processo de amostragem aumentaria a velocidade de convergência desses algoritmos para uma solução viável (rota livre de colisão) (LINDEMANN; LAVALLE, 2005). Em uma amostragem não uniforme/informada, algumas posições do ambiente de navegação têm maior probabilidade de serem coletadas do que outras. A principal diferença da amostragem não uniforme/informada em relação à uniforme, é o direcionamento/orientação da expansão da árvore de busca para regiões mais promissoras do ambiente de navegação, i. e., regiões onde uma rota tem maior chance de ser planejada, expansão orientada para a posição de destino.

Segundo Lindemann e LaValle (2006), normalmente duas abordagens são usadas para implementar a amostragem não uniforme/informada no algoritmo RRT. Na primeira, a expansão da árvore pode ser regionalmente direcionada, induzindo à coleta de amostras próximas às principais regiões de interesse. Essa abordagem é chamada de *amostragem por importância*. Na segunda, são empregadas estratégias

nas quais a distribuição espacial da coleta das amostras é alterada durante o planejamento, dependendo de restrições e novas informações que podem ser encontradas durante o planejamento. Esta abordagem é chamada de *amostragem adaptativa*. Ao adotar uma dessas abordagens, o processo de amostragem passa a sofrer um direcionamento na posição de novas amostras coletadas, o que reduz a quantidade de amostras necessárias para encontrar uma rota viável. No entanto, é necessário que a amostragem não uniforme/informada seja utilizada junto com a amostragem uniforme para garantir o comportamento probabilístico completo do algoritmo RRT (como descrito na Seção 3.1.2). Portanto, o uso de estratégias eficientes de estratégias não uniformes/informadas junto do uso da distribuição uniforme é primordial para o desempenho bem-sucedido de métodos baseados em amostragem não uniformes (LINDEMANN; LAVALLE, 2005).

Embora alguns autores tenham chamado a atenção para o tema há algum tempo, como em Lindemann e LaValle (2006), o estudo dos efeitos da amostragem não uniforme/informada no RRT é um tópico recente, como apontado em Noreen et al. (2016). Portanto, novos estudos são necessários para entender seus efeitos em comparação com a abordagem uniforme. No entanto, a comunidade científica já desenvolveu uma quantidade considerável de métodos e aplicou-os ao processo de amostragem de algoritmos baseados no RRT. A busca por trabalhos acadêmicos relacionados ao tema, principalmente envolvendo o algoritmo RRT, é uma tarefa complexa e exaustante devido à grande quantidade de estudos propostos na literatura. Algumas metodologias de revisão bibliográfica podem padronizar o método para pesquisar um tema específico na literatura para lidar com o enorme volume de trabalhos que podem ser encontrados em uma época como a de hoje de disseminação rápida e ampla de informações científicas. Uma técnica cada vez mais popular que lida com essas questões em diversas áreas de pesquisa é a Revisão Sistemática da Literatura (RSL). Um guia para realizar esse tipo de revisão é descrito em Kitchenham e Charters (2007).

Nesta tese, foram investigadas as melhorias propostas na literatura para a amostragem do RRT baseadas em estratégias não uniforme/informada. Essa investigação foi conduzida por meio da RSL descrita no Apêndice B. Os resultados da execução da RSL encontram-se publicados em Véras et al. (2019b). Cada um dos estudos selecionados foi classificado baseado nas características apresentadas por suas estratégias de amostragem propostas e agrupadas por semelhança de seus mecanismos, comportamentos e efeitos no crescimento da árvore do algoritmo RRT. Dessa forma, os processos de amostragem identificados na RSL foram divididos por classes, que

podem ser associadas a um dos tipos de amostragem definidos previamente: “por importância” e “adaptativa”. Portanto, dois processos de amostragem podem ser do mesmo tipo, porém, devido às diferenças nos elementos utilizados para causar o crescimento direcionado da árvore do RRT, de classes diferentes. Nas Seções 3.3.1 e 3.3.2 estão listados os algoritmos propostos pelos estudos identificados durante a condução da RSL do tipo “por importância” e “adaptativa” com suas respectivas classes.

3.3.1 Amostragem por importância

Nesta seção, as soluções de amostragem não uniforme/informada aplicadas ao problema de planejamento de rota com algoritmos baseados no RRT, que foram identificadas na execução da RSL e que são do tipo “por importância”, estão descritas. Diferentes classes de amostragem são classificadas como sendo do tipo “por importância”. As subseções a seguir apresentam as soluções de amostragem identificadas na literatura em cada uma das classes de amostragem do tipo “por importância”, que são: amostragem com orientação para o destino; amostragem com orientação para os obstáculos; amostragem com orientação regional; amostragem com orientação para a rota; e amostragem com orientação para passagens estreitas.

3.3.1.1 Amostragem com orientação para a posição de destino

Nessa classe de amostragem não uniforme/informada, o crescimento do algoritmo RRT é direcionado para a região próxima da posição de destino do planejamento da rota. Geralmente, as estratégias dessa classe utilizam explicitamente a posição de destino para estender a árvore com uma frequência definida por uma variável aleatória. Em contrapartida, outras estratégias aplicam métodos que utilizam os obstáculos e a posição de destino para implicitamente direcionar a coleta de novas amostras para a região de destino, como é o caso do método de campos potenciais artificiais (KHATIB, 1986). Os estudos selecionados durante a RSL que se enquadram nessa classe estão listados na Tabela 3.1.

Tabela 3.1 - Estudos coletados no processo de revisão sistemática da literatura que se enquadram na classe de amostragem com orientação para a posição de destino.

Referência	Tipo RRT	Amostragem	Solução	Completude	Otimalidade	Comparações
(ZUCKER et al., 2007)	ERRT, DRRT	*Goal bias sampling and forest bias*	MP-RRT	Probabilisticamente completo	Não ótimo	RRT, ERRT, DRRT
(XIA et al., 2009)	Multi-RRT	*Goal-bias sampling*	*Multi-RRT-GoalBias*	Probabilisticamente completo	Não ótimo	Multi-RRT
(HUAZHONG et al., 2012)	RRT	*Goal oriented sampling*	ERRT	Probabilisticamente completo	Não ótimo	RRT
(AHMADYAN et al., 2012)	RRT	*Learning approach*	*Goal-oriented test generation*	Probabilisticamente completo	Não ótimo	RRT
(KONG et al., 2013)	RRT	*Scent pervasion based sampling*	*Two stage RRT*	Probabilisticamente completo	Não ótimo	RRT, RRT-Connect
(QURESHI; AYAZ, 2016)	RRT*	*Artificial Potential Field*	P-RRT*	Probabilisticamente completo	Ótimo	RRT*
(KANG et al., 2016)	RRRT	*Goal-biased sampling*	*Goal oriented* RRT (GO RRT)	Probabilisticamente completo	Não ótimo	RRT, RRT-Connect and RRT*
(MOSES et al., 2017)	RRT	*Goal-bias sampling*	GRRT	Probabilisticamente completo	Não ótimo	RRT

(Continua)

Tabela 3.1 - Continuação

Referência	Tipo RRT	Amostragem	Solução	Completude	Otimalidade	Comparações
(GAO et al., 2018)	RRT	*Potential fields based sampling*	*RRT with potential fields-based sampling*	Probabilisticamente completo	Não ótimo	RRT

Fonte: Véras et al. (2019b).

Em Zucker et al. (2007), o algoritmo *Multipartite RRT (MP-RRT)* é proposto sobre a base conceitual dos algoritmos ERRT (HUAZHONG et al., 2012) e DRRT (FERGUSON et al., 2006) e aplicado a ambientes de navegação com obstáculos estáticos e dinâmicos (que mudam de posição com a variação de tempo). O algoritmo mantém uma floresta de RRTs desconectada da árvore que contém a posição inicial de planejamento (árvore principal). Conforme a árvore cresce, as novas amostras são coletadas entre as raízes das árvores da floresta dado um valor de probabilidade. Se os obstáculos no ambiente de navegação mudarem de posição, a árvore principal será reinicializada e seus nós são armazenados na floresta, a qual será usada em futuras iterações do algoritmo MP-RRT. Além disso, a cada iteração existe uma probabilidade de ocorrer orientação na amostragem na direção da posição de destino. Se o valor de probabilidade gerado aleatoriamente p for menor que um limite predefinido, a amostragem será influenciada pela região de destino. Caso contrário, se p for menor que a probabilidade de ocorrer orientação para o destino, então uma nova amostra é coletada usando a floresta de RRTs. Finalmente, se todas as tentativas de amostragem anteriores falharem, uma amostra aleatória com distribuição uniforme será coletada.

O algoritmo *Multi-RRT-GoalBias* é proposto em Xia et al. (2009), o qual é uma versão guiada para o destino do algoritmo Multi-RRT. Sua estratégia de amostragem utiliza um valor de probabilidade para executar a amostragem com orientação para o destino. Quando utilizada, a árvore é expandida na direção da posição de destino. Os autores também oferecem uma abordagem baseada em rede neural para definir o valor limite ideal de probabilidade para um determinado ambiente de navegação.

O algoritmo proposto ERRT (HUAZHONG et al., 2012) utiliza um valor de probabilidade para decidir se o processo de amostragem é definido pela posição de destino ou pela distribuição uniforme. Se um valor p for maior que um determinado parâmetro, a posição de destino é definida como a amostra para influenciar o crescimento do RRT.

O método teste orientado por meta (do inglês *Goal-oriented test*) é proposto em Ahmadyan et al. (2012). O método planeja uma rota em um espaço de estados bidimensional que representa estados de um circuito elétrico não linear analógico. Neste método, o RRT tende a crescer na direção das regiões onde a maioria das amostras dos estados do circuito ocorre. Para isso, duas fases são adicionadas ao RRT: a) uma fase de aprendizado, onde as amostras de estado são coletadas para identificar as regiões de interesse e b) uma fase de exploração, onde o RRT tem sua

expansão orientada para essas regiões de maior interesse. Na fase de aprendizado, k amostras são coletadas do espaço de estados do circuito analógico. Posteriormente, esse espaço é particionado em uma rede com grades de tamanhos iguais. Essas grades são então agrupadas pela quantidade de amostras pertencentes a cada uma delas. Esses agrupamentos recebem o nome de *cluster*. O *cluster* com o maior número de grades é definido como a região para onde a expansão da árvore do RRT deve ser enviesada/orientada. Na fase de exploração, as amostras do *cluster* são usadas como orientação para expandir a árvore do RRT. Em cada iteração, uma amostra é coletada de um dos *clusters* selecionados. Assim, o RRT converge mais rapidamente para essas regiões, as quais poderiam estar associadas a estados de falha do circuito. Finalmente, as rotas entre os estados inicial e final do circuito podem ser geradas de maneira mais rápida. As rotas planejadas pelo RRT são usadas para executar testes de circuito, o que manualmente poderia ser exaustivo.

O método baseado em RRT introduzido em Kong et al. (2013), chamado *Two Stage RRT*, aplica uma decomposição inicial do ambiente de navegação, e então, baseado nos resultados dessa decomposição, a árvore é construída. Na primeira etapa, o método proposto utiliza um algoritmo de impregnação de aromas (*scent pervasion algorithm*) para construir uma representação discreta do ambiente de navegação. Uma informação de aroma é difundida através das células de vizinhança resultantes da decomposição a partir da região de destino, o qual é a fonte desse aroma. Assim, etiquetas (*tags*) são atribuídas a cada célula do ambiente de navegação discretizado, representando a intensidade do aroma naquela posição, a qual fornece uma informação de proximidade entre a célula e a posição de destino. Após o estágio de discretização, o RRT é construído com base nas informações de aroma. As células que forem menos selecionadas durante o planejamento e possuírem menor valor de etiqueta terão maior preferência em serem selecionadas durante a expansão da árvore. É afirmado pelos autores que esse processo acelera o crescimento da árvore do RRT na direção da região de destino.

Uma solução para reduzir o efeito da aleatoriedade no RRT usando a técnica de Campos Potenciais Artificiais (do inglês *Artificial Potential Fields - APF*) é proposta em Qureshi e Ayaz (2016) para criar o método *Potential Guided Directional-RRT* (P-RRT*)*. Nesse algoritmo, a posição de destino, as amostras e os obstáculos emitem forças de polaridades opostas ou não para impulsionar a expansão da árvore. Cada nó da árvore tem a mesma polaridade dos obstáculos, o que cria um efeito de repulsão entre as amostras coletadas na amostragem, os nós da árvore existentes e os obstáculos. Em contraste, a iteração entre as novas amostras e a posição destino

resulta em uma força de atração que converge o RRT para a solução. Assim, o P-RRT* mantém o equilíbrio entre a exploração de amostras coletadas aleatoriamente e o uso de exploração não uniforme/informada. Segundo os autores, o novo método tem melhor convergência na busca de uma solução ótima de planejamento de rota do que o RRT*.

O algoritmo *The Goal Oriented RRT (GO RRT)* é proposto em Kang et al. (2016). O processo de amostragem desse algoritmo é influenciado pela região de destino por uma estratégia chamada amostragem Pre-GO. Esse processo de amostragem define o tamanho da região de coleta de novas amostras com base na distância entre o destino e o nó mais distante na árvore. Esta distância é usada como raio de uma região circular centrada na posição de destino. Novas amostras são coletadas nessa região dado um valor constante que determina a concentração de amostras a partir da posição de destino. A coleta irá ocorrer em maior concentração próximo da posição de destino, diminuindo gradativamente em direção aos limites da região circular definida pelo raio calculado.

Uma amostragem orientada para a posição de destino é proposta em Moses et al. (2017). O esquema de amostragem é aplicado ao algoritmo RRT. A solução proposta coleta duas amostras aleatórias simultaneamente a cada iteração. A amostra mais próxima da posição de destino é então selecionada para expandir a árvore. Segundo os autores, essa estratégia evita colisões locais causadas por estratégias tradicionais com orientação para a posição de destino (quando o destino dá explicitamente a direção de avanço da árvore), ao mesmo tempo em que otimiza o comprimento da rota e o tempo de planejamento.

Uma estratégia de amostragem baseada em campos potenciais artificiais é proposta em Gao et al. (2018). O método de campos potenciais é explorado de duas maneiras distintas pelo algoritmo proposto: de maneira estocástica e de maneira determinística. Na estratégia de amostragem estocástica, um grupo de amostras é coletado aleatoriamente a cada iteração e aquele com o menor valor potencial é selecionado para estender a árvore. Na amostragem determinística, cada nó da árvore tem uma classificação, com base na sua posição relativa a um alvo e a região dos obstáculos. Esse nó com a classificação mais alta é selecionado e uma amostra é coletada na direção da diferença de potencial com a posição de destino. Cada estratégia é selecionada de acordo com um valor probabilístico para equilibrar o uso de ambas. A estratégia de amostragem proposta foi desenvolvida para aplicação em um manipulador robótico. Segundo os autores, com o método proposto, o manipulador apresenta

melhor capacidade de responder a situações repentinas durante o seu processo de movimentação.

3.3.1.2 Amostragem com orientação para os obstáculos

Muitos estudos geralmente orientam o processo de amostragem por informações dos obstáculos ou regiões não desejáveis para navegação do ambiente de navegação para reduzir o comprimento da rota durante o planejamento e melhorar a eficácia da amostragem. Nesta seção, são relatados os trabalhos acadêmicos que apresentam soluções baseadas em amostragem com orientação para os obstáculos. Os dados extraídos desses trabalhos estão listados na Tabela 3.2.

Tabela 3.2 - Estudos coletados no processo de revisão sistemática da literatura que se enquadram na classe de amostragem com orientação para os obstáculos.

Referência	Tipo RRT	Amostragem	Solução	Completude	Otimalidade	Comparações
(PENG; ZHAO, 2010)	RRT-ConCon	*Random Triangle State*	Dangerzone RRT	Probabilisticamente completo	Não ótimo	DPRM
(FEI; YAO, 2012)	RRT	*Danger zones sampling*	Dangerzones RRT (DRRT)	Probabilisticamente completo	Não ótimo	DPRM
(WANG et al., 2016)	RRT	*Obstacle boundary information*	GeneteRRT	Probabilisticamente completo	Não ótimo	RRT
(MENG et al., 2017)	BIT*	*Obstacle-guided sampling*	BIT*-H	Probabilisticamente completo	Ótimo	BIT*, BIT*-G (com distribuição gaussiana)

Fonte: Véras et al. (2019b).

Em Peng e Zhao (2010) um algoritmo baseado no RRT é proposto, onde novas amostras são coletadas próximas de zonas de perigo no ambiente de navegação. Este algoritmo é chamado *Dangerzone RRT (DRRT)* e apresenta um novo processo de amostragem e o adiciona ao algoritmo RRT-ConCon. Em um ambiente de navegação constituído de obstáculos e zonas de perigo (regiões livres de colisões, mas indesejáveis à navegação), o processo de amostragem proposto utiliza superfícies triangulares obtidas a partir dessas zonas. Essas superfícies são utilizadas pelo processo de amostragem após o planejamento da primeira rota pelo algoritmo. Em seguida, nas execuções de amostragem subsequentes, um dos triângulos que formam as superfícies das zonas de perigo é selecionado aleatoriamente e uma amostra aleatória é gerada dentro dos seus limites. Essa amostra é então utilizada para a expansão do RRT, orientando o crescimento da árvore em direção dos limites das zonas de perigo.

O algoritmo proposto em Fei e Yao (2012), apresenta uma versão diferente do algoritmo *Dangerzone RRT (DRRT)*. Nessa versão o RRT têm o processo de amostragem ampliado por meio de técnicas existentes para amostragem em zonas de perigo e planejamento local. Duas estratégias de amostragem são aplicadas no DRRT. Na primeiro, uma medida de densidade é calculada para cada nó da árvore sempre que um novo nó for gerado. Assim, esses nós que possuem menor densidade são mais preferíveis de serem selecionados para expandir a árvore. Por fim, uma nova amostra é gerada na vizinhança do nó (definida como uma região circular com esse nó como centro) com o menor valor de densidade. O segundo utiliza zonas de perigo, regiões do ambiente de navegação nas quais é preferível que não se colete novas amostras. Para estender a árvore, uma amostra aleatória é coletada de triângulos que modelam essas zonas de perigo. Um triângulo de uma zona de perigo é selecionado aleatoriamente, definindo a direção para onde o RRT deve crescer na zona de perigo. Por fim, o nó da árvore mais próximo da amostra da zona de perigo é selecionado para estender o RRT. Segundo os autores, o efeito de aplicar essas estratégias de amostragem resulta em um planejamento mais efetivo (com menor quantidade total de nós da árvore e menor tempo total de planejamento) em cenários com zonas de perigo.

O algoritmo baseado em RRT proposto em Wang et al. (2016) utiliza as informações sobre os obstáculos que são tocados pelo segmento formado pela posição inicial e posição final do planejamento. No processo de amostragem proposto, cada um dos obstáculos tocados é classificado como valioso para o processo de planejamento. Depois, novas amostras são geradas nos pontos médios das arestas que definem os limites desses obstáculos. Os pontos médios gerados são utilizados pelo processo

de amostragem a partir de então. A cada execução de amostragem, o obstáculo mais próximo da posição inicial com pontos livres para conexão é selecionado para estendê-la. Por fim, um dos pontos médios do obstáculo é selecionado aleatoriamente para ser adicionado como uma nova amostra à árvore. Dessa forma, a expansão da árvore é orientada em direção da posição de destino e uma rota com comprimento reduzido pode ser obtida rapidamente.

Em Meng et al. (2017) um processo de amostragem com orientação para os obstáculos que gera uma distribuição de amostras mais densa próxima deles é proposta para o algoritmo BIT*, introduzindo o algoritmo denominado BIT*-H. A estratégia de amostragem introduzida é híbrida, com duas possibilidades de processos a serem executados. No primeiro, a amostragem baseada em uma região elipsoidal do algoritmo BIT* é utilizada. No entanto, quando uma amostra é gerada uniformemente dentro da elipse, uma amostragem baseada em vizinhança é executada para explorar informações locais em torno de amostras coletadas uniformemente. A média da vizinhança é definida como a nova amostra que irá finalmente estender a árvore. Se esta amostra média não for livre de colisão, uma nova amostra é gerada em torno da amostra coletada uniformemente seguindo uma distribuição gaussiana. Esses nós coletados pela distribuição gaussiana recebem o rótulo de "navegadores", caso alguns critérios sejam atendidos. No segundo processo, os nós navegadores são usados para influenciar o crescimento da árvore. O uso de cada processo de amostragem é definido por uma relação dinâmica entre o comprimento ótimo esperado e o comprimento atual da rota planejada. Essa relação define a proporção de uso de cada processo de amostragem durante o planejamento pelo BIT*-H.

3.3.1.3 Amostragem com orientação regional

Na amostragem com orientação regional, as amostras possuem a tendência de serem coletadas em direção de regiões consideradas mais importantes para o planejamento, baseadas em critérios como, por exemplo, o valor de custo da rota ou a maior probabilidade de gerar amostras livres de colisão. Algumas dessas estratégias são: informar previamente ao algoritmo de planejamento a classificação das regiões mais promissoras no ambiente de navegação; utilizar técnicas de aprendizado por amostragem para que essas regiões sejam descobertas em tempo de planejamento; e, utilizar hipóteses para inferir quais são as regiões mais promissoras a serem exploradas do ambiente de navegação. A seguir, as estratégias introduzidas pelos estudos nesta classe de métodos de amostragem retornados pela RSL são brevemente descritas. Os dados extraídos desses estudos estão listados na Tabela 3.3.

Tabela 3.3 - Estudos coletados no processo de revisão sistemática da literatura que se enquadram na classe de amostragem com orientação regional.

Referência	Tipo RRT	Amostragem	Solução	Completude	Otimalidade	Comparações
(LINDEMANN; LAVALLE, 2004)	RRT	*Voronoi diagram*	VB-RRT and DB-RRT	Probabilisticamente completo	Não ótimo	RRT-Connect
(LINDEMANN; LAVALLE, 2006)	RRT	*Multiple Sampling*	MS-RRT (has *a* and *b* versions)	Probabilisticamente completo	Não ótimo	RRTConCon
(MARTIN et al., 2007)	Bidirectional RRT	*Offline EA based sampling and online EA based sampling*	RET	Probabilisticamente completo	Não ótimo	Bidirectional RRT
(KHANMOHAMMADI; MAHDIZADEH, 2008)	RRT	*Sampling around the tree node with the smallest densisty*	DAS-RRT	Probabilisticamente completo	Não ótimo	RRT, Adaptative-DDRRT
(SAKAHARA et al., 2008)	StRRT	*Voronoi based bias*	Voronoi Based StRRT	Probabilisticamente completo	Não ótimo	StRRT
(FRAGKOPOULOS; GRAESER, 2010)	Bi-RRT	*N-dimensional Cuboids*	E-RRT	Probabilisticamente completo	Não ótimo	RRT, Bi-RRT

(Continua)

Tabela 3.3 - Continuação

Referência	Tipo RRT	Amostragem	Solução	Completude	Otimalidade	Comparações
(MOTWANI et al., 2014)	RRT	*Learn of the propagation of search space points of a dynamic system*	LPCA-RRT	Probabilisticamente completo	Não ótimo	RRT, GPCA-RRT
(KIM et al., 2014)	RRT*	*Sphere-based sampling*	Cloud RRT*	Probabilisticamente completo	Ótimo	RRT*-Smart, RRT*
(SUN; CHEN, 2015)	RRT	*Decrease/increase the sampling area*	EB-RRT	Probabilisticamente completo	Não ótimo	RRT, biased-RRT
(YU et al., 2016)	RRT	*Moving-Window Unilateral Gaussian Sampling Strategy*	MW-RRT	Probabilisticamente completo	Não ótimo	Greedy-RRT, Connect-RRT
(PALMIERI et al., 2016)	RRT	*Any angle sampling*	Theta*-RRT	Probabilisticamente completo	Não ótimo	A*-RRT, RRT, RRT*, A*-RRT*
(FANG et al., 2016)	RRT	*General Voronoi Diagram*	VB-RT	Probabilisticamente completo	Não ótimo	RRT

(Continua)

Tabela 3.3 - Continuação

Referência	Tipo RRT	Amostragem	Solução	Completude	Otimalidade	Comparações
(DU et al., 2016)	RRT	*Drivers' visual behavior sampling bias*	DV-RRT	Não completo	Não ótimo	RRT, Bi-RRT
(YUNCHENG; JIE, 2017)	RRT*	*Gaussian distribution sampling*	*RRT* with Gaussian distribution sampling*	Probabilisticamente completo	Ótimo	RRT*
(VONÁSEK; KOZLÍKOVÁ, 2017)	RRT	*Weighted Voronoi diagrams (WVD)*	*RRT-based search with Voronoi-based*	Probabilisticamente completo	Não ótimo	RRT
(TAHIR et al., 2018)	B-RRT*	*Artificial Potential Fields*	PB-RRT*, PIB-RRT*	Probabilisticamente completo	Ótimo	IB-RRT*, P-RRT*, RRT*
(WANG et al., 2018)	RRT	*Goal-based bias sampling*	RRT-DWA	Probabilisticamente completo	Não ótimo	Nenhum

(Continua)

Tabela 3.3 - Continuação

Referência	Tipo RRT	Amostragem	Solução	Completude	Otimalidade	Comparações
(ABBADI; MATOUSEK, 2018)	RRT	*Sampling probabilities and fuzzy logic*	*Boundaries-bias and fuzzy bias RRT*	Probabilisticamente completo	Não ótimo	RRT, *Biased RRT to expand to goal waypoint, Biased RRT to expand to itself node, Biased RRT by cell decomposition*

Fonte: Véras et al. (2019b).

Duas abordagens de planejamento baseados na orientação decorrente de regiões de Voronoi são propostos por Lindemann e LaValle (2004). A primeira abordagem, chamada *Volume-based RRT (VB-RRT)*, estima as áreas das regiões de Voronoi para decidir por onde realizar a expansão da árvore. Para evitar explicitamente a construção do Diagrama de Voronoi, que é computacionalmente custosa, uma estratégia onde k amostras aleatórias são coletadas simultaneamente do ambiente de navegação é proposta. Assim, a expansão da árvore é orientada na direção do ponto médio dessas k amostras. A segunda abordagem, chamada *Dispersion-based RRT (DB-RRT)*, incorpora um comportamento de expansão que exibe certo grau de dependência das regões de Voronoi com base na dispersão das k amostras coletadas do ambiente de navegação. A hipótese da abordagem é a de que estendendo o RRT para a amostra mais distante de sua árvore entre as amostras aleatórias de k, o algoritmo demonstrará um comportamento exato de orientação causada por regiões de Voronoi. Para implementar esse comportamento, o algoritmo classifica as k amostras em ordem decrescente de distância ao seu nó mais próximo na árvore. A amostra aleatória com maior distância de seu nó mais próximo é selecionada para expandir a árvore do RRT.

O método *Multi-Sample RRT (MS-RRT)* é proposto em Lindemann e LaValle (2006). Nessa abordagem, k amostras são coletadas simultaneamente durante a expansão do RRT, em contraste com o processo de amostragem tradicional do RRT, no qual apenas uma amostra é coletada por iteração. O crescimento da árvore é então direcionado para o ponto médio dessas amostras coletadas. Dois algoritmos são introduzidos com base no esquema de multi-amostragem. No primeiro, chamado MS-RRTa, o parâmetro k define quantas amostras aleatórias são coletadas por iteração do RRT. Para cada uma dessas amostras, seu nó vizinho mais próximo é calculado. O nó da árvore que for mais frequentemente selecionado como o vizinho mais próximo, é definido como o melhor nó para a expansão da árvore. A segunda abordagem introduzida é identificada pelo algoritmo MS-RRTb. Ele usa um conjunto pré-definido de k amostras que são usadas para estimar a orientação causada por regiões de Voronoi para a expansão da árvore. Porém, essas amostras são selecionadas apenas uma vez antes do início do planejamento. Assim, esse conjunto é reutilizado em cada iteração da expansão da árvore, ao contrário do MS-RRTa, que sempre gera um novo conjunto de amostras a cada iteração. A hipótese dos autores é que o uso de um mesmo conjunto de amostras que foram selecionadas seguindo uma distribuição uniforme causa um efeito de orientação causada por regiões de Voronoi similar ao de selecioná-las em cada iteração, porém, com menor custo computacional.

O trabalho em Martin et al. (2007) propõe o algoritmo chamado *Rapidly Exploring Evolutionary Tree (RET)*. Nesse algoritmo, um algoritmo evolutivo (*evolutionary algorithm - EA*) que gera indivíduos (pontos no ambiente de navegação) e valores de aptidão que determinam o nível de eficiência que esses indivíduos causam no crescimento da árvore do algoritmo RRT. A estratégia é aplicada a um RRT bidirecional, onde duas árvores crescem simultaneamente a partir da posição de início e destino cada. No processo de amostragem as novas amostras são coletadas com base nos indivíduos do EA. Uma amostra tem 50% de probabilidade de ser coletada por indivíduos com EA, caso contrário, será coletada pela amostragem aleatória com distribuição uniforme.

Um novo esquema de amostragem é proposto em Khanmohammadi e Mahdizadeh (2008) para superar situações em que o RRT é aprisionado pela orientação causada por regiões de Voronoi em obstáculos com formato de labirinto. No algoritmo proposto, chamado *Density Avoided Sampling RRT (DDAS-RRT)*, a árvore é enviesada para escapar de áreas de alta densidade. Um parâmetro de densidade (quantidade de nós vizinhos) é adicionado a cada nó da árvore, que informa um espaço livre estimado associado a ele. O nó da árvore com o menor valor de densidade associado é selecionado para expandir a árvore. Assim, uma área de vizinhança em torno desses nós será a região onde uma nova amostra será coletada. Um procedimento aleatório coleta uma nova amostra dentro desta região.

O algoritmo *Voronoi StRRT* proposto em Sakahara et al. (2008), uma versão modificada de *Spatiotemporal-RRT (StRRT)*, tem um processo de amostragem baseado no diagrama de Voronoi generalizado. As novas amostras coletadas são influenciadas pelos limites do diagrama local de Voronoi. O objetivo dessa estratégia é reduzir o tempo computacional na presença de obstáculos estáticos e dinâmicos no ambiente de navegação. Para alcançar esses resultados, o Voronoi StRRT executa duas etapas. Primeiro, uma amostra coletada é movida na direção oposta do ponto na região dos obstáculos mais próximos. Em seguida, a quantidade de movimento é estimada com base na distribuição dos obstáculos vizinhos e sua distância do ponto mais próximo na superfície dos obstáculos. Como resultado, uma nova amostra é colocada próxima aos limites das regiões de Voronoi e, como resultado, rotas mais seguras são planejadas.

O algoritmo proposto em Fragkopoulos e Graeser (2010) chamado de *Extended bidirectional RRT (E-RRT)* é uma extensão do algoritmo Bi-RRT. O processo de amostragem do algoritmo é influenciado pela posição de destino de navegação e um

cubóide N-dimensional. Em cada iteração, o algoritmo tenta estender a árvore na direção da posição de destino. Então, se não for possível conectar a árvore com o destino, ela é expandida por cubóides N-dimensionais. Essa estrutura é adicionada pela expansão do RRT, onde cada novo nó adicionado tem um cubóide associado a ele. Esse cubóide é gerado dado uma distância que o desloca em direção oposta ao último nó adicionado na árvore. Assim, na etapa de amostragem do RRT, uma nova amostra aleatória é coletada dentro dos limites cubóides do último nó estendido da árvore.

O algoritmo proposto em Motwani et al. (2014) é chamado de *Local Principal Component Analysis RRT (LPCA-RRT)*. O processo de amostragem desse algoritmo utiliza um modelo de função de custo baseado no espaço de estados de um sistema dinâmico para melhorar o planejamento local por meio de orientação na amostragem em um ambiente de navegação discretizado. A estratégia de amostragem é apresentada contendo duas etapas. Na primeira, aprende-se a direção dos pontos de propagação no ambiente de navegação discretizado quando o sistema é simulado nesses pontos. Na segunda etapa, o processo de amostragem do LPCA-RRT é enviesado para as regiões onde os pontos no primeiro passo se propagaram. Desta forma, a árvore converge para as regiões do espaço de estados do sistema que são mais prováveis de ocorrer.

No algoritmo *Cloud RRT** proposto em Kim et al. (2014), o ambiente de navegação é decomposto em um conjunto de esferas com raio de diferentes tamanhos. Cada uma das esferas possui um valor único de probabilidade de ser selecionada durante o processo de amostragem. A posição das esferas no ambiente de navegação e seus valores de probabilidade são inicialmente definidos por um *Generalized Voronoi Graph (GVG)* baseado na distância de visibilidade da posição inicial de planejamento, nos vértices e borda do grafo de Voronoi e na otimização da sobreposição entre as esferas. Quando uma das esferas é selecionada pelo processo de amostragem, uma nova amostra é aleatoriamente coletada dentro da esfera com distribuição uniforme. Durante o planejamento pelo *Cloud RRT**, novas esferas são criadas com base nas rotas anteriormente planejadas. Desse modo, o comprimento da rota é otimizado pelas amostras obtidas por esferas baseadas em nós anteriores.

Para reduzir o efeito da aleatoriedade no RRT e melhorar sua eficiência na busca de rotas, Sun e Chen (2015) propôs o algoritmo *Exponential Back-off sampling Rapid-exploring Random Tree (EB-RRT)*. O processo de amostragem nesse algoritmo é orientado em direção de regiões do ambiente de navegação denominadas de zonas

intermediárias. Essas zonas são regiões de Voronoi que não são nem mal exploradas (contém poucas amostras coletadas) e nem sobrecarregadamente exploradas (contém muitas amostras coletadas), ou seja, são regiões exploradas de forma mediana. Para induzir a árvore do algoritmo RRT a crescer para elas, uma estratégia chamada de amostragem de recuo é executada quando ocorre uma colisão na operação de extensão. Esse recuo consiste em aumentar e reduzir a área de amostragem, a qual está associada à posição de destino do planejamento. O efeito desta estratégia é que a árvore cresce em torno dos obstáculos, otimizando o comprimento da rota.

O algoritmo proposto em Yu et al. (2016), denominado *Moving-Window Rapidly Exploring Random Tree (WM-RRT)*, usa um gerador de amostras gaussiano unilateral para coletar novas amostras próximas a nós que possuem maior chance de se tornarem novos nós para a árvore. A hipótese dos autores é que existe uma maior probabilidade da árvore se expandir a partir de nós adicionados tardiamente em relação aos que foram adicionados mais no início do processo de planejamento. Além disso, eles afirmam que as amostras coletadas em torno da linha do segmento formado pela posição inicial e de destino otimizam o comprimento das rotas planejadas.

O Theta*-RRT (PALMIERI et al., 2016) é um algoritmo com um processo de amostragem orientado por uma rota pré-planejada do tipo “qualquer ângulo” (do inglês *any-angle*) pelo algoritmo de busca baseado em grade Theta*. Uma faixa de largura W centrada ao redor da rota pré-planejada é usada para gerar novas amostras coletadas aleatoriamente dentro dela. Cada nova amostra tem uma direção de ângulo associada definida aleatoriamente, dado um intervalo angular. Assim, a árvore cresce em torno da rota do tipo "qualquer ângulo"definida pelo algoritmo Theta*. Segundo os autores, essa estratégia permite que uma rota mais curta e mais suave possa ser planejada rapidamente em um ambiente de navegação contínuo.

O algoritmo proposto em Fang et al. (2016), chamado VB-RT, cria um *Generalized Voronoi Diagram (GVD)* 2D para extrair a distribuição espacial dos obstáculos no ambiente de navegação em 3D. Assim, pontos são distribuídos localmente em torno dos nós das bordas do diagrama de Voronoi 2D e selecionados de acordo com uma função de custo no processo de amostragem do RRT, em cada iteração. Os nós de Voronoi com menor custo são amostrados para influenciar o crescimento da árvore do RRT.

Inspirado em estudos sobre o comportamento visual de motoristas, um conceito de que os humanos utilizam dois pontos de vista para guiar um veículo em curvas, um distante e outro próximo, o trabalho em Du et al. (2016) propõe um algo-

ritmo de amostragem orientada denominado *Drivers' Visual Behavior-guided RRT (DV-RRT)*. O algoritmo é aplicado no contexto para carros autônomos. Os pontos distantes e próximos do comportamento visual dos motoristas são utilizados para direcionar o processo de amostragem do RRT para coletar amostras que são mais adequadas aos movimentos do veículo. Assim, um dos pontos é selecionado dados critérios de velocidade e novas amostras são geradas dada uma distribuição gaussiana em torno dele. Segundo os autores, essa estratégia simplifica a rota, evitando a geração de amostras desnecessárias para o movimento do veículo em curvas.

Em Yuncheng e Jie (2017), o processo de amostragem aleatória uniforme do RRT é substituído por um processo de amostragem com distribuição gaussiana. O valor médio da distribuição gaussiana é definido quase próximo da posição de destino de planejamento, fazendo com que o processo de amostragem colete novas amostras na região dessa posição. O desvio padrão da distribuição gaussiana determina a eficácia do algoritmo proposto.

Um algoritmo baseado em RRT para detecção de tuneis em estruturas de proteínas foi proposto em Vonásek e Kozlíková (2017). O método de amostragem do algoritmo mantém uma lista de regiões de vértices de Voronoi. O tamanho da região dos vértices de Voronoi é definido pela distância de seu obstáculo mais próximo (no domínio do problema é um átomo). Assim, novas amostras são coletadas em torno dos vértices de Voronoi que são novos nós na árvore. Quando a árvore já explorou uma região de vértices de Voronoi, ela é desativada do processo de amostragem.

Em Tahir et al. (2018), os autores evoluíram o algoritmo *bidirectional RRT* (B-RRT*)* e *Intelligent Bi-directional RRT* (IB-RRT*)* associando-os à estratégia de campos potenciais artificiais, criando os algoritmos PB-RRT* e o PIB-RRT*, respectivamente. Nesses algoritmos, uma nova amostra aleatória é guiada por uma diferença de potência na expansão do B-RRT*. A estratégia decide se a orientação causada pelos campos potenciais sofrerá influência da posição inicial ou da final de navegação, dependendo se a contagem de iterações da expansão é par ou não.

Uma versão do algoritmo RRT com a estratégia denominada *Dynamic Window Approach (DWA)* é proposto em Wang et al. (2018) para planejamento de rota de veículos autônomos, introduzindo o algoritmo chamado *RRT-DWA*. Um esquema de amostragem baseado na seleção da posição de destino como orientação para o crescimento da árvore é aplicado de acordo com um valor de probabilidade. O valor de probabilidade é definido dada a densidade de obstáculos no ambiente de navegação. Se um valor gerado aleatoriamente for maior que o valor de probabilidade,

a amostragem é direcionada para a posição de destino. Caso contrário, a amostra é coletada aleatoriamente. O DWA é utilizado para o planejamento local da velocidade do veículo, o que leva em consideração suas restrições cinemáticas.

O uso de probabilidades de amostragem e lógica *fuzzy* para obter informações de colisão e orientar a expansão do RRT é considerado em Abbadi e Matousek (2018). O método apresentado nesse estudo, utiliza um ambiente de navegação particionado pelo algoritmo de decomposição de células, onde cada célula é associada com valores de probabilidades. Cada uma é classificada, onde algumas regiões do ambiente de navegação tornam-se mais importante que outras. Com base nisso, duas estratégias de planejamento são empregadas: uma com orientação para fronteira; e outro com orientação baseada em funções *fuzzy*. Na orientação para as fronteiras, as classes das regiões são definidas de acordo com a relação de vizinhança dessas regiões/células do ambiente de navegação com os nós do RRT. Regiões com menor quantidade de nós são mais importantes para a amostragem. Na orientação por funções *fuzzy*, os valores de peso são calculados para cada região/célula por regras *fuzzy* baseadas em informações de colisão. Segundo os autores, essas estratégias reduzem o problema de crescimento do RRT para situações de mínimos locais (quando amostras em ambiente de navegação não permitem possibilidades de exploração) e induzem uma melhor exploração do ambiente de navegação.

3.3.1.4 Amostragem com orientação para rota

Durante o planejamento de rota, aquelas que já foram encontradas em iterações anteriores podem ser usadas para induzir a coleta de amostras com potencial de acelerar a convergência do planejamento para a rota de menor comprimento em um dado ambiente de navegação. Normalmente, nesse tipo de estratégia de amostragem, os nós das rotas anteriores são utilizados para influenciar na posição de novos nós candidatos a serem incluídos na árvore do RRT. As estratégias propostas pelos estudos nesta classe estão descritas a seguir e listadas na Tabela 3.4.

Tabela 3.4 - Estudos coletados no processo de revisão sistemática da literatura que se enquadram na classe de amostragem com orientação para rota.

Referência	Tipo RRT	Amostragem	Solução	Completude	Otimalidade	Comparações
(AKGUN; STILMAN, 2011)	bi-RRT*	*Local bias by middle of two neighbors*	*Bi-RRT* with local sampling*	Probabilisticamente completo	Ótimo	bi-RRT*
(KOBILAROV, 2012)	RRT*	*Cross-entropy based sampling*	SCE-RRT*, TCE-RRT*	Probabilisticamente completo	Ótimo	RRT*
(ISLAM et al., 2012)	RRT*	*Intelligent sampling*	RRT*-Smart	Probabilisticamente completo	Ótimo	RRT*
(ZHANG et al., 2014)	RRT	*Bias by the most selected nodes on a reference path*	GRRT	Probabilisticamente completo	Não ótimo	RRT
(SEEGMILLER et al., 2017)	RRT*	*Directed acyclic graph (DAG) based sampling*	*RRT* using Waypoint Graph*	Probabilisticamente completo	Ótimo	RRT*
(CHAO et al., 2017)	T-RRT*	*Biased path areas sampling*	BT-RRT*	Probabilisticamente completo	Ótimo	T-RRT*
(BOARDMAN et al., 2019)	RRT*	*The Focused-Refinement sampling*	*Goal Tree* RRT*	Probabilisticamente completo	Não ótimo	RRT*, RRT*-Smart

Fonte: Véras et al. (2019b).

Em Akgun e Stilman (2011) uma estratégia de amostragem local é proposta para o algoritmo Bi-RRT*. Essa estratégia utiliza três etapas de amostragem, nas quais cada uma é executada dado um valor de probabilidade. São elas: orientação para a direção da posição do destino; a amostragem aleatória com distribuição uniforme; e a orientação local que é descrita a seguir. Quando uma rota é planejada, a amostragem local é executada para influenciar o crescimento da árvore. Primeiro, um dos nós da última rota planejada é selecionado aleatoriamente. Em seguida, seu nó pai e nó filho são obtidos. Esses dois nós são representados como vetores. Com base nessa representação, o vetor que representa o ponto médio entre os nós pai e filho é calculado. Finalmente, uma nova amostra é gerada a partir do nó selecionado da árvore na direção do ponto médio calculado por uma distância gerada aleatoriamente entre valores máximo e mínimo pré-definidos. Os autores argumentam que essa estratégia gera rotas mais retilíneas do que a amostragem aleatória uniforme, o que significa que as rotas otimizadas são geradas mais rapidamente.

Em Kobilarov (2012), um esquema de amostragem adaptativo de estados de sistemas dinâmicos baseados no método *Cross-Entropy (CE)*, o qual realiza otimização estocástica global, é proposto para o algoritmo RRT*. Na estratégia proposta, inicialmente, a rota ótima é obtida pelo RRT* sem considerações sobre restrições dinâmicas e cinemáticas. Assim, a rota ótima é usada para alimentar o modelo probabilístico, o qual é baseado em um modelo de mistura de gaussianas (do inglês *gaussian mixture model - GMM*), para indicar as regiões com o menor custo no ambiente de navegação que geram rotas ótimas restringidas por parâmeros de um sistema autônomo. Duas versões do algoritmo RRT* com amostragem CE são propostas. A primeira é a *trajectory-cross-entropy (TCE) RRT**, onde uma função de distribuição é atualizada com os melhores parâmetros de trajetórias do sistema (ao invés de amostras que indicam apenas posições) que geram os menores custos e melhores trajetórias. O segundo é o *State-Cross-Entropy (SCE) RRT**, onde as amostras são coletadas do espaço de estados do sistema autônomo diretamente pelas distribuições modificadas pelo método *cross-entropy*.

Uma abordagem baseada em amostragem inteligente e otimização de rotas chamada RRT*-Smart é proposta em Islam et al. (2012). Esse algoritmo é aplicado a cenários em que o planejamento de rota é executado até que a rota ótima (a mais curta possível) de um ambiente de navegação possa ser encontrado. O método funciona de maneira semelhante ao algoritmo baseado em custo RRT*. No entanto, após uma primeira rota ser encontrada entre as posições inicial e final de planejamento, ela é otimizada conectando nós diretamente visíveis entre si no ambiente de navegação.

Para otimizar o comprimento das rotas planejadas, novas amostras são coletadas próximas de nós denominados *beacons*, que são nós da última rota anteriormente planejada. Depois que a primeira rota é encontrada, o processo de amostragem se baseia na utilização desses *beacons*. As novas amostras são geradas aleatoriamente dentro de círculos centradas em um dos *beacons* selecionado aleatoriamente. Segundo os autores, o método acelera o tempo de convergência para encontrar a rota ótima em relação ao algoritmo RRT*, considerando as mesmas posições de início e destino de planejamento.

Em Zhang et al. (2014) é apresentado um algoritmo com uma combinação de planejamento geométrico e planejamento cinemático/dinâmico denominado de *Guided RRT*. O algoritmo possui duas camadas: uma onde o RRT constrói a árvore representando as restrições geométricas e outra que é amostrada com base nos nós da primeira. Os nós da árvore com restrições cinemáticas e dinâmicas são amostrados a partir dos nós da árvore geométrica, dado um valor de probabilidade p, caso contrário, a amostragem é realizada aleatoriamente. Cada vez que um novo nó geométrico é selecionado pela amostra cinemática/dinâmica, sua probabilidade de seleção é reduzida. Dessa forma, a amostragem do algoritmo *Guided RRT* da árvore cinemática/dinâmica é influenciada pelos nós da árvore geométrica que foram menos selecionados anteriormente.

O processo de amostragem proposto em Seegmiller et al. (2017) é baseado em um *Directed Acyclic Graph (DAG)* de *waypoints* construídos por técnicas de decomposição celular e pelo *General Diagram Voronoi (GDV)*. Os *waypoints* do DAG são utilizados para expandir o RRT*. Dois tipos de *waypoints* são mantidos no DAG: "percorridos"e "de fronteira". Quando um dos *waypoints* é adicionado ao RRT*, seus ancestrais são classificados como percorridos. Os descendentes do *waypoint* são classificados como fronteira. Assim, a árvore é expandida em direção aos nós de fronteira com maior probabilidade do que os *waypoints* percorridos. Os autores do algoritmo argumentam que a expansão da árvore, pelos nós da fronteira, otimiza o custo da rota planejada.

O estudo apresentado em Chao et al. (2017) introduz uma estratégia de amostragem chamada *amostragem de área de rota enviesada* para o algoritmo *Transition-based -RRT*(T-RRT*)*, criando então o algoritmo *Biased Transition-based-RRT*(BT-RRT*)*. A amostragem parcial inicia depois que a primeira rota é planejada ou quando uma rota com comprimento menor é encontrada. Os nós de um rota planejada são utilizados na orientação da amostragem a partir de então. Para cada um

dos nós da rota, uma amostra é gerada aleatoriamente em torno dele, considerando um raio de distância. Se uma amostra otimizar o comprimento da rota no nó a partir do qual foi gerada (otimização local), considerando a possível conexão com seus vizinhos, ele é adicionado à árvore. O processo otimiza o comprimento da rota cada vez que uma com menor comprimento é encontrada pela árvore.

O algoritmo apresentado em Boardman et al. (2019) introduz o algoritmo de refinamento focalizado para direcionar o crescimento do árvore do algoritmo RRT* após o planejamento de uma primeira rota. Para uma determinada rota planejada, o algoritmo usa o valor máximo e mínimo do k-ésimo componente dos nós da rota. Esses valores definem o intervalo a partir do qual o k-ésimo componente de uma amostra aleatória é coletada (por exemplo, o componente x da amostra). Em seguida, após a definição do primeiro componente, outro k-ésimo componente (por exemplo, o componente y da amostra) é definido com base no k-ésimo componente do nó mais próximo do componente anteriormente definido (no caso, o componente x). Então, dentro de um intervalo definido pelo último k-ésimo somado a um valor de deslocamento, o último componente é gerado aleatoriamente e finalmente uma nova amostra é coletada, definida por todos os k componentes. Esse esquema de amostragem é executado com base em um valor constante que define a proporção de uso junto da amostragem aleatória com distribuição uniforme. O estudo proposto ainda apresenta duas melhorias adicionais ao RRT*: a árvore de destino e as modificações de conexão de nós. A primeira é uma árvore com raiz na posição de destino que realiza replanejamento quando em cenários com obstáculos dinâmicos e a segunda é um processo de otimização de rota que tenta conectar um nó ao avô, e não ao pai, para reduzir a quantidade de nós na rota.

3.3.1.5 Amostragem com orientação para passagens estreitas

Esse tipo de orientação de amostragem é utilizado para superar problemas de expansão apresentados pelo algoritmo RRT em áreas de difícil exploração, como passagens estreitas definidas por obstáculos. Acelerando a exploração nessas regiões, pode ser observada a redução no tempo de planejamento total da rota. As estratégias de amostragem com orientação para passagens estreitas identificadas na RSL são descritas a seguir. Informações sobre as estratégias propostas encontram-se listadas na Tabela 3.5.

Tabela 3.5 - Estudos coletados no processo de revisão sistemática da literatura que se enquadram na classe de amostragem com orientação para passagens estreitas.

Referência	Tipo RRT	Amostragem	Solução	Completude	Otimalidade	Comparações
(BLIN et al., 2016)	IRRT, RRT-Connect	*Contact algorithm sampling*	I-RRT-C	Probabilisticamente completo	Não ótimo	RRT-Connect, I-RRT
(ZHANG; MANOCHA, 2008)	RRT	*Retraction-based sampling*	RRRT	Probabilisticamente completo	Não ótimo	RRT
(LEE et al., 2012)	RRT	*Retraction-based techniques*	SR-RRT	Probabilisticamente completo	Não ótimo	RRT, RRRT

Fonte: Véras et al. (2019b).

O algoritmo *Interactive RRT in Contact (I-RRT-C)* é proposto em Blin et al. (2016). Esse algoritmo é uma versão estendida do algoritmo de planejamento autônomo-humano *Iteration RRT (I-RRT)* (FLAVIGNE et al., 2009) para planejamento de rota em ambientes de navegação 3D. Um novo esquema de amostragem é adicionado com base no contato da árvore com os obstáculos. Quando a árvore está próxima de um obstáculo, o algoritmo entra no denominado "modo de contato". Neste modo, um plano local tangente ao obstáculo mais próximo é usado para coletar novas amostras. As amostras são coletadas aleatoriamente sobre a tangente, permitindo que o RRT "deslize"pelos obstáculos.

Em Zhang e Manocha (2008), um algoritmo baseado no RRT, chamado *Retraction-based RRT*, é proposto com otimização de amostragem em passagens estreitas e outras em situações de contato. O algoritmo utiliza uma estratégia baseada em retração de amostras em situação de colisão, transladando-as para os limites dos obstáculos, melhorando então a exploração do RRT em passagens estreitas. Inicialmente, para uma amostra em colisão gerada aleatoriamente, seu nó mais próximo na árvore é selecionado. Este nó será a indicação inicial para onde a amostra em colisão deve ser reposicionada. Na passagem estreita, o nó selecionado mais próximo possui uma alta probabilidade de estar próximo do limite de um obstáculo. Assim, uma nova amostra é gerada pela otimização baseada em retração, que gera uma amostra também próxima ao limite do obstáculo. Finalmente, a amostra é adicionada à árvore como um novo nó. Este novo nó tem provavelmente a maior densidade de Voronoi, então é mais provável que ele seja selecionado nas próximas iterações. Dessa forma, uma amostra criada pela operação de retração é quase sempre selecionada nas iterações subsequentes. Isso faz com que a exploração em passagens estreitas seja acelerada.

Em Lee et al. (2012), a aplicação da técnica de otimização baseada na retração de amostras para influenciar o crescimento do RRT em passagens estreitas foi estudada. O objetivo da abordagem proposta é retrair amostras coletadas na região dos obstáculos para uma posição livre de colisão próxima ou sobre o limite desses obstáculos, o que é útil em passagens estreitas. O algoritmo baseado no conceito de retração é denominado *Selective Retraction-based RRT (SR-RRT)*. Dois testes são propostos para identificar passagens estreitas no SR-RRT: teste de linha de ponte e teste de linha de não-colisão. O teste de linha de ponte consiste em identificar a existência de uma passagem estreita através de uma linha reta quando amostras aleatórias são coletadas sobre a região dos obstáculos. Assim, uma linha gerada a partir do nó mais próximo das amostras é usada para identificar a passagem estreita. Se a passagem for detectada, a operação de retração é realizada para coletar uma nova amostra sobre

os limites dos obstáculos. O teste de linha de não-colisão registra os espaços livres de colisão abertos no ambiente de navegação e descarta amostras geradas dentro deles e gera mais amostras fora dessas áreas, o que potencialmente direciona a árvore do RRT para as passagens estreitas. Segundo os autores, as estratégias empregadas orientam a coleta de novas amostras na direção dos limites dos obstáculos.

3.3.2 Amostragem adaptativa

Nesta seção, as soluções de amostragem não uniforme/informada aplicadas ao problema de planejamento de rota com algoritmos baseados no RRT, que foram identificadas na execução da RSL e que são do tipo "adaptativa", estão descritas. Como ocorre na seção anterior, as subseções a seguir apresentam as soluções de amostragem identificadas na literatura em cada uma das classes de amostragem do tipo "adaptativa", que são: redução do ambiente de navegação e modificação da distribuição das amostras.

3.3.2.1 Amostragem com orientação por redução do ambiente de navegação

Alguns processos propostos de otimização da amostragem do algoritmo RRT adicionam um custo computacional ao planejamento em contrapartida de uma melhor convergência para uma rota de menor comprimento. Entretanto, muitos problemas de planejamento de rota possuem a restrição de obter uma capacidade de exploração melhorada no RRT mantendo o seu desempenho computacional. Para permitir isso, muitos trabalhos propõem a redução da área de amostragem do ambiente de navegação. Dessa forma, novas amostras podem ser coletadas apenas nesta área reduzida, desconsiderando amostras que provavelmente não poderiam melhorar a qualidade da rota planejada. As estratégias baseadas na redução da área de amostragem encontram-se descritas a seguir. Essas mesmas estratégias estão listadas na Tabela 3.6.

Tabela 3.6 - Estudos coletados no processo de revisão sistemática da literatura que se enquadram na classe de amostragem com orientação por redução do ambiente de navegação.

Referência	Tipo RRT	Amostragem	Solução	Completude	Otimalidade	Comparações
(GAMMELL et al., 2014)	RRT*	*Hyperellipsoid-based sampling*	Informed-RRT*	Probabilisticamente completo	Não ótimo	RRT*
(KIM; SONG, 2015)	Informed-RRT*	*Hyperellipsoid-based sampling*	*Wrapping-based* Informed RRT* (WIRRT*)	Probabilisticamente completo	Ótimo	RRT*, Informed RRT*
(GAMMELL et al., 2015)	Informed-RRT*	*RRG batch sampling*	BIT*	Probabilisticamente completo	Ótimo	RRT*, FMT*, Informed-RRT*, RRT-Connect.
(ZHANG et al., 2016)	RRT*	*Self-learning process*	*Self-learning* RRT*	Probabilisticamente completo	Ótimo	RRT, DD-RRT

Fonte: Véras et al. (2019b).

O algoritmo *Informed-RRT** (GAMMELL et al., 2014) reduz o ambiente de navegação com base no custo da rota. Esse custo define o tamanho de um subespaço elipsoide, de onde novas amostras são coletadas. O tamanho da região elipsoidal é reduzido se há redução do custo da rota. Os autores afirmam que o uso do elipsoide faz com as amostras que geram as melhores rotas sejam coletadas com maior probabilidade.

Em Kim e Song (2015) é proposta uma versão modificada do algoritmo Informed-RRT* chamado *Wrapping-based Informed RRT* (WIRRT*)*. Um processo de encapsulamento é aplicado para acelerar a redução da região elipsoidal que limita a área de amostragem. O processo otimiza o planejamento da rota substituindo seus nós intermediários por aqueles que encontram-se envolvendo os limites dos obstáculos. Consequentemente, o custo da rota é otimizado e o tamanho da elipse, que é dependente desse custo, também é reduzido.

Com base na geração da sub-região elipsoidal no *Informed*-RRT*, o algoritmo *Batch Informed Tree (BIT*)* é proposto em Gammell et al. (2015). Inicialmente, um *Random Geometric Graph (RGG)* é definido por amostras aleatórias uniformemente distribuídas na região de não-colisão do ambiente de navegação. Uma árvore que conecta a posição inicial e a posição final de planejamento é heuristicamente pesquisada (por algoritmos como A*, Dijkstra) sobre o RGG. A árvore resultante é usada como um lote para gerar uma região elipsoidal inicial que é uma sub-região do ambiente de navegação que, teoricamente, contém as melhores amostras para otimizar a rota. Assim, o processo é reiniciado com uma sub-região menor formada pelo custo da rota atual. Cada vez que uma rota com melhor custo é planejada, o novo lote é formado. O processo continua conforme o tempo de planejamento permitir. Desta forma, as rotas otimizadas são geradas mais rapidamente do que outros algoritmos de planejamento baseados em RRT.

O algoritmo apresentado em Zhang et al. (2016) introduz uma autoaprendizagem do ambiente de navegação para adaptar o processo de amostragem dependendo se a região que está sendo explorada é densa em obstáculos ou aberta (com poucos obstáculos). O processo de verificação de colisão fornece informações sobre a distribuição das regiões de obstáculos. Para conseguir isso, cada nó da árvore tem um disco contendo vários setores com informações baseadas na visibilidade do ambiente de navegação gerado por um sensor virtual sobre o nó. Cada setor é classificado com base nos obstáculos identificados dentro da faixa de alcance do seu sensor virtual. Os setores que possuem visibilidade de um obstáculo têm seu alcance de visibilidade reduzido. Além disso, o setor em direção à posição de destino é mais importante do

que outros no processo de amostragem. O esquema de amostragem proposto começa por coletar uma amostra aleatória e encontrar o nó mais próximo na árvore. Assim, o disco deste nó é usado para coletar uma amostra parcial. Cada setor tem uma probabilidade de ser selecionado. Quando um setor é selecionado, um nó é selecionado aleatoriamente dentro de sua área. Segundo os autores, espera-se que, com a amostragem baseada em autoaprendizagem, a árvore possa explorar mais amplamente o ambiente de navegação desde o início do planejamento e, em seguida, melhorar os ramos da árvore em fases mais a frente do planejamento.

3.3.2.2 Amostragem com orientação por modificação da distribuição das amostras

Tradicionalmente, o algoritmo RRT seleciona aleatoriamente uma nova amostra do ambiente de navegação dada uma distribuição uniforme. Assim, cada amostra possui igual probabilidade de ser coletada. As estratégias dessa classe modificam a distribuição da geração das amostras para uma que permita coletar amostras que acelerem o processo de planejamento de rota. Normalmente, uma distribuição modificada pode ser definida desde o início do planejamento ou distribuições adaptativas podem ser alteradas com base em amostras coletadas durante o planejamento, aumentando a eficiência do processo de amostragem à medida que o RRT cresce. As estratégias observadas nesta classe de amostragem estão descritas a seguir. Os seus dados estão listados na Tabela 3.7.

Tabela 3.7 - Estudos coletados no processo de revisão sistemática da literatura que se enquadram na classe de amostragem com orientação por modificação da distribuição das amostras.

Referência	Tipo RRT	Amostragem	Solução	Completude	Otimalidade	Comparações
(JOUANDEAU; CHERIF, 2004)	RRT	*Gaussian-based sampling distribution bounded by obstacles visibility*	*Localized random sampling based* RRT	Probabilisticamente completo	Não ótimo	RRT, Distribution-based RRTs
(KIM; ESPOSITO, 2005)	RRT	*Adaptive sampling based on Gaussian distribution*	*RRT with Adaptive Bias*	Probabilisticamente completo	Não ótimo	RRT
(KIM et al., 2005)	RRT	*Adaptive sampling by probabilistic density function changing*	*Enhanced-RRT*	Probabilisticamente completo	Não ótimo	*Uniform RRT, RRT with Medial Bias, RRT with Heavy Bias*
(PARK et al., 2014)	RRT	*Poisson-disk sampling*	Poisson-RRT	Probabilisticamente completo	Não ótimo	RRT-Connect, Lazy-RRT, pRRT (*parallel processing*)

(Continua)

Tabela 3.7 - Continuação

Referência	Tipo RRT	Amostragem	Solução	Completude	Otimalidade	Comparações
(ARSLAN; TSIOTRAS, 2015)	RRT#	*Adaptive Sampling Strategy*	Sem nome	Probabilisticamente completo	Ótimo	Nenhum
(HUH; LEE, 2016)	RRT	*Gaussian Mixture Model (GMM)*	GMM-based RRT	Probabilisticamente completo	Não ótimo	RRT, *Bi-directional* RRT
(VISERAS et al., 2016)	RRT*	*Ant colony optimization*	ACO-RRT*	Probabilisticamente completo	Não ótimo	ACO-RRT, RRT*, RRT
(YI et al., 2018)	Informed-RRT*	*Markov Chain Monte Carlo-based Informed Sampling (MCMC)*	*Informed-RRT* based on MCMC*	Probabilisticamente completo	Ótimo	Nenhum

Fonte: Véras et al. (2019b).

Em Jouandeau e Cherif (2004) é proposto um RRT baseado em visibilidade para avaliar vários espaços abertos em um ambiente de navegação com obstáculos. Uma distribuição gaussiana é considerada para gerar amostras aleatórias em torno dos limites dos obstáculos. A distribuição é afetada pela quantidade de obstáculos visíveis pelos nós da árvore. Com a expansão da árvore do RRT, eventualmente mais obstáculos se tornam "visíveis" pelos nós, alterando a distribuição de amostragem. O algoritmo também usa uma amostragem orientada pela posição do destino.

Um esquema adaptativo é proposto por Kim e Esposito (2005) para alterar a amostragem uniforme do algoritmo RRT. O algoritmo de amostragem adaptativa começa com um esquema de busca direcionada e, em seguida, se a árvore estiver crescendo rapidamente, mantém ou aumenta a influência dessa orientação. Se a taxa de crescimento diminuir, uma estratégia de amostragem tendendo à uniforme será utilizada. Se a velocidade de expansão da árvore for mais rápida, as amostras serão coletadas em direção às regiões dos obstáculos. O nível de uniformidade é ajustado alterando o desvio padrão de uma distribuição gaussiana. Com a estratégia de amostragem adaptativa, as rotas são geradas mais rapidamente do que com o uso de amostragem estritamente uniforme.

Uma versão aprimorada do algoritmo RRT é proposta em Kim et al. (2005). Sua amostragem não uniforme/informada gera um novo algoritmo adaptativo controlado por um fator dependente do número de vezes em que uma amostra coletada é bem sucedida em estender a árvore por arestas de menor custo. A quantidade de iterações executadas no planejamento define quando uma função densidade de probabilidade Gaussiana é modificada por esse fator. Assim, amostras subsequentes no planejamento são coletadas, dada a distribuição modificada recentemente. Segundo os autores, por meio desse esquema de mudança de função de densidade de probabilidade, o algoritmo adaptativo melhora consideravelmente a eficiência do RRT em comparação com outras estratégias não uniforme/informadas não adaptativas.

Uma abordagem baseada na amostragem de discos de Poisson é proposta em Park et al. (2014). Nesta abordagem, um pré-planejamento é considerado, onde amostras aleatórias (centros de discos) são coletadas dada uma distribuição de disco de Poisson. Os discos são amostrados de forma aninhada e compacta, separados por uma distância mínima r entre os centros de disco. Os centros dos discos de Poisson são então utilizados para expandir a árvore do algoritmo RRT. Esta metodologia tem como objetivo obter o máximo de amostragem, o que significa cobrir completamente o ambiente de navegação por discos de Poisson utilizando o mínimo de amostras.

Os autores afirmam que essa nova abordagem reduz o número de nós na árvore e pode ser implementada em processamento paralelo com várias *threads* (linhas de processamento que podem ser executadas de forma independente e simultânea). Além disso, foi proposto no trabalho um esquema para aumentar a proporção de amostras próximas às áreas de difícil exploração do ambiente de navegação, como passagens estreitas.

Em Arslan e Tsiotras (2015), uma abordagem de descoberta do ambiente de navegação é proposta com base em aprendizado de máquina. As informações aprendidas são usadas em uma estratégia de amostragem adaptativa e aplicadas ao algoritmo RRT# (ARSLAN; TSIOTRAS, 2013). O funcionamento dessa estratégia é descrito a seguir. Uma nova amostra é coletada com base na distribuição uniforme. Assim, se a amostra for livre de colisão ou não, funções de densidade de classe são aproximadas por um classificador baseado nessas amostras coletadas. Essas funções de densidade (atualizadas em cada iteração) são usadas para determinar a posição das novas amostras adicionadas à árvore. Como resultado, a probabilidade de gerar amostras livres de colisão é maior que na distribuição aleatória uniforme.

Uma estratégia baseada em *Gaussian Mixture Model (GMM)* é proposta como processo de amostragem para o RRT em Huh e Lee (2016). O GMM aprende sobre as regiões que levam à colisão no ambiente de navegação por uma abordagem de agrupamento baseada nas amostras coletadas anteriormente, a partir das quais uma função de distribuição pode ser estimada. Dois GMMs são utilizados pelo método proposto: um para representar as amostras em colisão; outro para representar amostras na região livre de colisão. Para obter essas informações, cada GMM é atualizado a cada iteração do RRT. A diferença entre os GMMs é utilizada para estimar posições de colisão para se obter uma maior probabilidade de gerar amostras nas regiões navegáveis do ambiente de navegação no processo de amostragem. Segundo os autores, essa estratégia reduz o número de operações de checagens de colisões e, consequentemente, o tempo de planejamento de rotas.

O algoritmo apresentado em Viseras et al. (2016), chamado de *Ant Colony Optimization RRT* (ACO-RRT*)*, determina a distribuição ideal de amostragem por otimização através de colônias de formigas artificiais. A estratégia apresentada funciona como descrito a seguir. Inicialmente, abstrações de formigas são distribuídas de acordo com sua relevância pelo ambiente de navegação. Com base na exploração do RRT* em um dado instante de planejamento, essas formigas são atualizadas, o que resulta na modificação da distribuição de amostragem do algoritmo. Portanto,

com base nessa distribuição estimada, novos nós são amostrados para expandir a árvore. Em seguida, a eficiência dessa distribuição em melhorar a qualidade da rota é avaliada com base na otimização da solução atual e na exploração do ambiente de navegação para encontrar novas e melhores rotas. De posse das informações geradas por essa avaliação, as formigas são atualizadas e uma nova distribuição é estimada. Segundo os autores, este processo segue otimizando a capacidade de exploração do RRT* e reduz o custo da rota mais rápido do que outros algoritmos baseados em RRT.

O algoritmo proposto em Yi et al. (2018) é baseado no algoritmo Informed-RRT*. Nas etapas iniciais do planejamento do algoritmo proposto, inicialmente uma primeira amostra é coletada aleatoriamente com distribuição uniforme sobre a região informada (definida por um hiperelipsoide n-dimensional). A partir daí, o procedimento de amostragem segue sobre a mesma região informada com base em um processo denominado *Markov Chain Monte Carlo (MCMC)*. O MCMC gera amostras de acordo com uma distribuição-alvo que é estimada com base em amostras coletadas anteriormente e no custo da rota que define a região informada (uma característica do algoritmo Informed-RRT*). Os autores mencionam que o uso do MCMC reduz muito mais rapidamente os custos das rotas planejadas em dimensões mais altas do ambiente de navegação do que o Informed-RRT* com distribuição uniforme.

4 ESTUDO DE PROCESSOS DE AMOSTRAGEM BASEADOS EM GRADES DE SUKHAREV E VÉRTICES CONVEXOS

Neste capítulo, o algoritmo introduzido em Véras et al. (2019a), denominado RRT*-SV (RRT* Sukharev Vértices) é descrito, o qual é baseado em dois conceitos para introduzir um novo processo de amostragem, que são: a grade de Sukharev e os vértices convexos das envoltórias de segurança dos obstáculos. Na Seção 4.1 e na Seção 4.2, a grade de Sukharev e o uso dos vértices convexos das envoltórias de segurança dos obstáculos no processo de amostragem são introduzidos, respectivamente. Para verificação do desempenho individual de cada estratégia, cada uma delas é utilizada como processo de amostragem não uniforme/informada, gerando dois algoritmos para comparação, RRT*-Sukharev e o RRT*-Vértices. Ambos os algoritmos, além de uma versão híbrida com os dois processos de amostragem considerados, são comparados ao algoritmo RRT* e as diferenças no planejamento de rota entre eles são discutidas na seção 4.3. Na seção 4.4, é descrito um algoritmo que é resultado do refinamento da versão híbrida anterior, denominado RRT*-SV. Na seção 4.5, testes computacionais considerando ambientes de navegação com diferentes distribuições espaciais de obstáculos são descritos e utilizados na comparação do RRT*-SV com os algoritmos RRT*, RRT*-Smart (NASIR et al., 2013), Informed-RRT* (GAMMELL et al., 2014) e BIT* (GAMMELL et al., 2015). Os resultados desses testes computacionais dão evidências de que o algoritmo RRT*-SV consegue, de forma geral, obter rotas de menor comprimento em menor tempo de planejamento do que os demais algoritmos considerados.

4.1 Grade de Sukharev

A grade de Sukharev faz parte de um grupo de abordagens de dispersão de pontos denominada *baixa amostragem* (LAVALLE, 2006). O objetivo dessas abordagens é permitir cobertura/amostragem de todas as regiões de um ambiente de navegação uniformemente (LAVALLE, 2006). A grade de Sukharev foi inicialmente estudada em Sukharev (1971), e ela permite que um conjunto de pontos possa ser distribuído o mais uniformemente possível sobre um ambiente de navegação.

A grade de Sukharev é gerada para uma dada quantidade k de células para cobrir um plano ($\mathbb{R}^2$) ou um espaço ($\mathbb{R}^3$). Neste trabalho, a grade é aplicada para cobrir o ambiente de navegação $Q \in \mathbb{R}^2$. Portanto, Q é particionado em k células. A quantidade de células de grade por eixo do ambiente de navegação é definida por

$$\lfloor k^{1/N} \rfloor \tag{4.1}$$

onde N é a dimensão do ambiente de navegação. Cada uma das células geradas possui um ponto posicionado no seu centro geométrico denominado centroide. Na Figura 4.1 um exemplo da grade de Sukharev com centroides de $k = 196$ células no $\mathbb{R}^2$ encontra-se ilustrado.

Figura 4.1 - Distribuição das células geradas pela grade de Sukharev para $k = 196$.

Fonte: LaValle (2006).

A amostragem de pontos está diretamente ligada a sua dispersão. A dispersão é representada pelo padrão de amostragem dos pontos em um ambiente de navegação. Entre esses pontos existem subespaços livres, nos quais nenhum ponto foi ainda amostrado, podendo esse subespaço ser representado por um círculo. O maior espaço livre entre os pontos de um conjunto de amostras define a dispersão para esse conjunto. Então, quanto menor a dispersão, menor é o tamanho do maior espaço vazio entre os pontos amostrados. Formalmente, a dispersão δ de um conjunto de pontos P é definido como:

$$\delta(P, \rho) = sup_{q \in Q} min_{p \in P} \rho(q, p) \tag{4.2}$$

onde Q é o ambiente de navegação e ρ é uma métrica em Q. Isso significa que a dispersão pode ser caracterizada pelo maior raio (considerando ρ) de uma região

circular vazia no ambiente de navegação.

O propósito de usar a grade do Sukharev no processo de amostragem é manter a exploração de todas as regiões navegáveis do ambiente de navegação, sem necessariamente amostrar todos os possíveis pontos das mesmas. Assim, sabendo que um ambiente de navegação com representação contínua possui infinitos pontos, todos aqueles pertencentes a uma das células da grade de Sukharev são representados por um único ponto, que é seu centroide. O efeito esperado é que uma quantidade menor de pontos sejam amostrados para que uma rota livre de colisão seja planejada. A grade de Sukharev já foi explorada experimentalmente e teoricamente com algoritmos baseados em amostragem em estudos anteriores como, por exemplo, em Lindemann e LaValle (2003), Sánchez (2003) e Janson et al. (2018).

4.1.1 RRT*-Sukharev

Para verificar a influência do uso da grade de Sukharev no planejamento de rotas pelo algoritmo RRT*, um algoritmo denominado RRT*-Sukharev foi desenvolvido. Nesse algoritmo, o uso dos centroides da grade de Sukharev foi adicionado ao processo de amostragem e encontra-se descrito no Algoritmo 4.

Algorithm 4 Algoritmo RRT*-Sukharev

1: **procedure** RRT*-SUKHAREV$(Q, q_{início}, q_{destino}, \Delta q, l_d, n, k)$
2: $G \leftarrow \{q_{início}\}$
3: $R \leftarrow \{\}$
4: $s \leftarrow 0$
5: $i \leftarrow 0$
6: $S \leftarrow GRADE_SUKHAREV(k)$
7: $\Delta s \leftarrow PASSO_SUKHAREV(S)$
8: **while** $(s = 0)$ e $(i \leq n)$ **do**
9: $i \leftarrow i + 1$
10: $q_{aleatório} \leftarrow AMOSTRAGEM_ALEATÓRIA(Q)$
11: $q_{próximo} \leftarrow NÓ_PRÓXIMO(q_{aleatório}, G)$
12: $q_{novo} \leftarrow AMOSTRAGEM_SUKHAREV(q_{próximo},\ S, \Delta s)$
13: **if** $q_{novo} = \{\}$ **then**
14: $q_{novo} \leftarrow NOVO_NÓ(q_{próximo}, q_{aleatório}, \Delta q)$
15: **if** $\overline{q_{próximo} q_{novo}}$ não intercepta $Q_{obstáculos}$ **then**
16: $RRT_ESTRELA_ESTENDE(G, q_{próximo}, q_{novo})$
17: **if** $d(q_{novo}, q_{destino}) \leq l_d$ e $(\overline{q_{novo} q_{destino}})$ não intercepta $Q_{obstáculos}$) **then**
18: $ESTENDE(G, q_{novo}, q_{destino})$
19: $s \leftarrow 1$
20: $R \leftarrow ROTA(q_{início}, q_{destino})$

Os principais procedimentos do algoritmo RRT*-Sukharev, apresentado no Algoritmo 4, são descritos a seguir:

a) **AMOSTRAGEM_SUKHAREV**: gera uma nova amostra do ambiente de navegação usando a grade de Sukharev S (Algoritmo 5);

b) **PASSO_SUKHAREV**: retorna o valor da menor distância entre os centroides das células da grade de Sukharev para a geração de q_{novo_temp};

c) **CENTROIDE_SUKHAREV**: retorna o centroide da célula que contém q_{novo_temp}, caso esse centroide forme uma aresta livre de colisão com o mesmo e a célula correspondente já não tenha sido utilizada no processo de amostragem.

No Algoritmo 5 encontra-se descrito o procedimento $AMOSTRAGEM_SUKHAREV$ e o uso dos demais procedimentos listados.

Algorithm 5 Processo de amostragem por grade de Sukharev do algoritmo RRT*-Sukharev

1: **procedure** AMOSTRAGEM_SUKHAREV($q_{próximo}, q_{aleatório}, S, \Delta s$)
2: $q_{novo_temp} \leftarrow NOVO_NÓ(q_{próximo}, q_{aleatório}, \Delta s)$
3: $q_{centroide} \leftarrow CENTROIDE_SUKHAREV(S,\ q_{novo_temp})$
4: **retorna** $q_{centroide}$

O processo de amostragem pela grade de Sukharev consiste em três etapas. Na primeira etapa, para uma dada amostra $q_{aleatório}$, uma amostra q_{novo_temp} é gerada no segmento de reta $\overline{q_{próximo} q_{aleatório}}$, considerando uma distância Δs de $q_{próximo}$ (linha 2 do Algoritmo 5). Esta etapa tem um papel equivalente a $NOVO_NÓ$ do Algoritmo 1. O valor de Δs corresponde a menor distância entre as células da grade de Sukharev. Assim, a amostra q_{novo_temp} é sempre obtida na própria célula contendo $q_{próximo}$ ou em uma das células adjacentes a essa célula. Quanto menor o valor k usado para gerar a grade, maior o valor de Δs. A segunda etapa é identificar a célula à qual q_{novo_temp} pertence (linha 3 do Algoritmo 5). Na terceira etapa, é verificado se o segmento formado pelo centroide dessa célula com o nó $q_{próximo}$ intercepta algum limite de uma envoltória de segurança (linha 15 do Algoritmo 4). Se isso não ocorrer, o centroide da célula será armazenado como um novo nó (q_{novo}) da árvore G. As etapas de amostragem pela grade de Sukharev encontram-se ilustradas na Figura 4.2.

Figura 4.2 - Estágios na amostragem pela grade de Sukharev.

a) O nó $q_{próximo}$ mais próximo de $q_{aleatório}$ é selecionado. b) um nó temporário q_{novo_temp} é gerado a uma distância Δs de $q_{próximo}$. c) o centroide da célula contendo q_{novo_temp} é adicionado como o novo nó da árvore q_{novo}. Os quadrados marrons espalhados pelo ambiente de navegação são centroides de células da grade gerada pela distribuição de Sukharev.

Fonte: Produção do Autor.

4.2 Vértices convexos de envoltórias de segurança dos obstáculos

Na representação do ambiente de navegação através de polígonos, os obstáculos são definidos por vértices conectados por arestas que definem seus limites. Dependendo do arranjo espacial entre duas arestas com um mesmo vértice em comum, esse vértice pode ser classificado como convexo ou côncavo. Considerando os vértices de um obstáculo poligonal ordenado no sentido horário, um vértice v_i deste obstáculo é convexo, se a sequência v_{i-1}, v_i, v_{i+1} encontra-se no sentido horário. Caso contrário, o vértice é côncavo.

A aplicação de vértices convexos no planejamento de rotas é mais conhecida através do algoritmo de grafo de visibilidade (LATOMBE, 2012). Um grafo de visibilidade gera uma estrutura em rede que permite que seja planejada a rota mais curta entre duas posições em um ambiente de navegação. A estratégia explorada pelo método é a de conectar os vértices convexos das envoltórias de segurança dos obstáculos uns aos outros, pois, eles formam arestas livres de colisão, gerando então um grafo denominado *roadmap*. Um exemplo de envoltória de segurança está ilustrado na Figura 4.3. Os vértices conectados podem pertencer ao mesmo obstáculo desde que suas posições estejam em uma sequência que defina uma de suas bordas ou podem estar em obstáculos diferentes desde que sejam bitangentes (quando um segmento de linha tangência ambos).

Figura 4.3 - Exemplo de uma envoltória de segurança (azul) para um obstáculo (preto).

Os pontos verdes representam os vértices convexos dessa envoltória de segurança.

Fonte: Produção do Autor.

Após a formação da *roadmap* pelos vértices convexos, os nós $q_{início}$ e $q_{destino}$ são incorporados a ela, conectando-os aos vértices convexos visíveis livre de colisão. Como resultado, um grafo de visibilidade é gerado contendo diferentes possibilidades de rotas entre $q_{início}$ e $q_{destino}$ por meio dos vértices convexos. A rota mais curta pode ser planejada por um método de busca em grafos, como o algoritmo Dijkstra (DIJKSTRA, 1959) ou A* (HART et al., 1968). Na Figura 4.4 encontra-se ilustrado um exemplo de planejamento onde o grafo de visibilidade foi utilizado.

Figura 4.4 - Exemplo de planejamento por meio de grafo de visibilidade.

a) Formação da *roadmap* por meio dos vértices convexos dos obstáculos. b) Adição de $q_{início}$ e $q_{destino}$ a *roadmap* pelos vértices convexos visíveis. c) Rota mais curta (linha vermelha) no grafo de visibilidade.

Fonte: LaValle (2006).

O desenvolvimento do algoritmo proposto neste trabalho é baseado na propriedade do grafo de visibilidade, que utiliza os vértices convexos para planejar uma rota de comprimento mais curto. Neste caso, os vértices convexos das envoltórias de segurança são considerados elementos de Q_{livre}.

4.2.1 RRT*-Vértices

Para verificar a influência do uso dos vértices convexos no planejamento de rotas pelo algoritmo RRT*, um algoritmo denominado RRT*-Vértices é introduzido nesta seção. Nesse algoritmo, o uso dos vértices convexos das envoltórias de segurança dos obstáculos foram adicionadas ao processo de amostragem. Ele encontra-se descrito no Algoritmo 6.

Algorithm 6 Algoritmo RRT*-Vértices

1: **procedure** RRT*-VÉRTICES($Q, q_{início}, q_{destino}, \Delta q, l_d, n$)
2: $G \leftarrow \{q_{início}\}$
3: $R \leftarrow \{\}$
4: $s \leftarrow 0$
5: $i \leftarrow 0$
6: $Q_{vértices} \leftarrow VÉRTICES_CONVEXOS(Q_{obstáculos})$
7: **while** $(s = 0)$ e $(i \leq n)$ **do**
8: $i \leftarrow i + 1$
9: $q_{aleatório} \leftarrow AMOSTRAGEM_ALEATÓRIA(Q)$
10: $q_{próximo} \leftarrow NÓ_PRÓXIMO(q_{aleatório}, G)$
11: $q_{novo} \leftarrow AMOSTRAGEM_VÉRTICES(q_{próximo}, Q_{vértices})$
12: **if** $q_{novo} = \{\}$ **then**
13: $q_{novo} \leftarrow NOVO_NÓ(q_{próximo}, q_{aleatório}, \Delta q)$
14: **if** $\overline{q_{próximo} q_{novo}}$ não intercepta $Q_{obstáculos}$ **then**
15: $RRT_ESTRELA_ESTENDE(G, q_{próximo}, q_{novo})$
16: **if** $d(q_{novo}, q_{destino}) \leq l_d$ e $(\overline{q_{novo} q_{destino}})$ não intercepta $Q_{obstáculos}$) **then**
17: $ESTENDE(G, q_{novo}, q_{destino})$
18: $s \leftarrow 1$
19: $R \leftarrow ROTA(q_{início}, q_{destino})$

Os principais procedimentos do Algoritmo RRT*-Vértices, apresentado no Algoritmo 6, são descritos a seguir:

a) **VÉRTICES_CONVEXOS**: obtém todos os vértices convexos do ambiente de navegação e os armazena em $Q_{vértices}$;

b) **AMOSTRAGEM_VÉRTICES**: retorna o vértice convexo que gera

uma aresta livre de colisão com $q_{próximo}$ (Algoritmo 7);

c) **VÉRTICE_PRÓXIMO**: procura o vértice convexo mais próximo de $q_{próximo}$;

d) **DESABILITA_VÉRTICE**: desativa para seleção em $Q_{vértices}$ o vértice selecionado como q_{novo}. Dessa forma, o mesmo não pode ser mais conectado a nenhum outro nó da árvore.

No processo de amostragem por vértices convexos, cada novo nó amostrado do ambiente de navegação é um vértice convexo contido em $Q_{vértices} \subset Q_{livre}$. Esses vértices, quando adicionados à árvore, são denominados *vértices nós*. Desta forma, uma rota é planejada próxima à envoltória de segurança dos obstáculos, com o objetivo de acelerar o planejamento da rota com a menor extensão possível no ambiente de navegação, ou seja, a rota ótima. O processo de amostragem por vértices convexos encontra-se descrito no Algoritmo 7.

Algorithm 7 Processo de amostragem por vértices convexos do algoritmo RRT*-Vértices

1: **procedure** AMOSTRAGEM_VÉRTICES($q_{próximo}, Q_{vértices}$)
2: $\quad v_{próximo} \leftarrow VÉRTICE_PRÓXIMO(Q_{vértices}, q_{próximo})$
3: $\quad$ **if** $\overline{q_{próximo} v_{próximo}}$ não intercepta $Q_{obstáculos}$ **then**
4: $\quad\quad q_{vpróximo} \leftarrow v_{próximo}$
5: $\quad\quad DESABILITA_VÉRTICE(Q_{vértices}, v_{próximo})$
6: $\quad$ **retorna** $q_{vpróximo}$

Nesta estratégia de amostragem, após uma amostra $q_{aleatório}$ ser gerada aleatoriamente, o nó da árvore mais próximo $q_{próximo}$ de $q_{aleatório}$ é selecionado, assim como na RRT padrão. Após isso, o vértice convexo de $Q_{vértices}$ mais próximo de $q_{próximo}$ é identificado. Essas operações são realizadas por $VÉRTICE_PRÓXIMO$ na linha 2 do Algoritmo 7. Por questões de desempenho computacional, é importante considerar uma estratégia otimizada para realizar essa operação. Neste trabalho, $Q_{vertices}$ é representado por uma estrutura k-d-tree (BENTLEY, 1975) para esta finalidade.

O nó $q_{próximo}$ será conectado ao vértice convexo mais próximo $q_{vpróximo}$ encontrado na operação anterior, desde que a aresta formada entre ambos seja livre de colisão. Quando um vértice convexo é adicionado à árvore G, ele se torna um vértice nó e não pode mais ser retornado à $Q_{vertices}$ em expansões posteriores da árvore. Se

$q_{vpróximo}$ não permitir a formação de uma aresta livre de colisão, o algoritmo tentará estender a árvore pela geração de novo nó como no algoritmo RRT padrão. Se todas as tentativas de gerar q_{novo} falharem, a próxima iteração será executada, repetindo todo o processo para um novo $q_{aleatório}$ coletado aleatoriamente por uma distribuição uniforme. O processo de extensão da árvore do RRT*-Vértices encontra-se ilustrado na Figura 4.5.

Figura 4.5 - Estágios do processo de amostragem baseados em vértices convexos.

a) Uma amostra aleatória $q_{aleatório}$ é coletada e o nó $q_{próximo}$ de G mais próximo é obtido. b) O vértice convexo $q_{vpróximo}$ de $Q_{vertices}$ mais próximo de $q_{próximo}$ é pesquisado através de uma estrutura de dados *k-d-tree*. c) Se a aresta que conecta $q_{vpróximo}$ e $q_{próximo}$ é livre de colisão, então $q_{vpróximo}$ é adicionado à árvore como q_{novo}. Uma vez adicionado à árvore, $q_{vpróximo}$ não estará mais disponível para ser coletado em uma amostragem futura.

Fonte: Produção do Autor.

4.3 Testes computacionais com o algoritmo RRT* usando processos de amostragem baseado em Grades de Sukharev e baseado em Vértices Convexos

Para avaliar os efeitos de cada estratégia proposta na aceleração do planejamento de rotas no algoritmo RRT*, foram realizados testes computacionais com diferentes variações do processo de amostragem, onde: amostras são coletadas por distribuições espaciais uniformes, como as utilizadas pelo algoritmo RRT*; amostras são coletadas por meio de distribuições espaciais com dispersão ótima de pontos com grades de Sukharev (RRT*-Sukharev); amostras são coletadas dos vértices convexos das envoltórias de segurança de obstáculos (RRT*-Vértices); amostras são coletadas por vértices convexos das envoltórias de segurança dos obstáculos e, em caso de falha (colisão), são coletados por grades de Sukharev (RRT*-Híbrido).

O desempenho de cada processo de amostragem é verificado pela porcentagem de sucesso em planejar uma rota em ambiente de navegação com diferentes distribuições espaciais de obstáculos estáticos. O comprimento das rotas planejadas também são comparadas em relação ao comprimento ótimo para cada ambiente de navegação considerado. Os ambientes de navegação utilizados e seus respectivos comprimentos ótimos são:

a) com 50 obstáculos com comprimento ótimo igual à 1393,12;

b) com 100 obstáculos com comprimento ótimo igual à 1375,31;

c) em zigue-zague com comprimento ótimo igual à 1006,45;

d) em formato espiral com comprimento ótimo igual à 2315,99;

e) com passagem estreita com comprimento ótimo igual à 1581,25.

As rotas com comprimento ótimo foram planejadas pelo algoritmo Dijkstra em busca realizada sobre um grafo de visibilidade formado para cada ambiente de navegação. Essas rotas ótimas encontram-se ilustradas no Apêndice A. Todos os ambientes de navegação possuem intervalos iguais a $[0, 1000]$ na dimensão x e intervalo $[0, 1000]$ na dimensão y.

Nos testes comparativos, foram realizadas medições das taxas de sucesso de cada algoritmo em alcançar um determinado valor de fração entre o comprimento da rota

planejada pelo mesmo e o comprimento ótimo. O valor dessa fração é determinada pela fórmula

$$f_{rrt} = c_{rrt}/c_{ótimo} \tag{4.3}$$

onde c_{rrt} é o comprimento da rota planejada por um dos algoritmos baseados no RRT e $c_{ótimo}$ é o comprimento ótimo de rota para um dado ambiente de navegação. Essas medições são comparadas a valores de frações f_i pré-definidos. Os valores de frações utilizados neste teste computacional são iguais à $1,20$, $1,15$, $1,10$, $1,05$, $1,04$, $1,03$, $1,02$ e aos valores do intervalo $[1,02, ..., 1]$ incrementados em $0,0008$.

Para cada ambiente de navegação, foram realizadas 100 simulações de cinco segundos de planejamento por cada algoritmo. A taxa de sucesso foi medida pela contabilização de quantos resultados geram frações que são iguais ou menores do que um valor de fração pré-determinado. Portanto, se os 100 valores de comprimentos de rotas gerados por um algoritmo resultarem em valores de f_{rrt} iguais ou menores do que um certo valor de fração f_i, então o algoritmo obtém 100% de sucesso para este valor.

O valor de Δq, que define o passo da expansão de uma aresta em algoritmos baseados na RRT, foi equiparado à menor distância entre os centroides da grade de Sukharev (representado por Δs) em todos os algoritmos que utilizam distribuição uniforme em sua amostragem, como ilustrado na Figura 4.6. Portanto, para um ambiente de navegação com intervalo igual a $[0, 1000]$ na dimensão x e intervalo $[0, 1000]$ na dimensão y, uma grade terá a seguinte relação número de células/distância (Δs):

a) com 64 células será igual a 125,00;

b) com 128 células será igual a 90,91;

c) com 256 células será igual a 62,50;

d) com 512 células será igual a 45,46;

e) com 1024 células será igual a 31,25.

Figura 4.6 - Exemplos de menores distâncias Δs para valores de k distintos.

Quanto maior o valor de k, menor será o valor de Δs.
Fonte: Produção do Autor.

Para cada um dos valores de Δs, um gráfico com as taxas de sucesso dos algoritmos foi gerado. Cada gráfico é referenciado de acordo com o valor de Δs associado ao número de células da grade de Sukharev. Por exemplo, se o valor de Δs adotado for o correspondente a uma grade com 64 células, então o gráfico é referenciado por d64, onde $\Delta s = 125,00$, pois, esse valor é referente à menor distância entre os centroides da grade de Sukharev com 64 células. Os valores de Δs são calculados utilizando distância euclidiana.

Em cada conjunto de gráficos comparativos, é juntamente introduzido um gráfico comparando o desempenho da grade de Sukharev de diferentes tamanhos. Dessa forma, pode-se observar a quantidade de células da grade que permite o melhor desempenho de planejamento em cada ambiente de navegação.

4.3.1 Análise dos resultados

Na Figura 4.7 e na Figura 4.8, encontram-se ilustrados os gráficos relativos ao ambiente de navegação com 50 obstáculos. Com base nos gráficos da Figura 4.7, nota-se que as rotas planejadas pelos algoritmos RRT*-Vértices e RRT*-Híbrido se aproximam de valores de frações (próximas de 1) que indicam que suas rotas possuem

comprimento mais próximo da rota ótima do que os demais algoritmos. Comparando ambos, nota-se que o desempenho apresentado é similar entre eles para o ambiente de navegação com 50 obstáculos. Também nota-se que o desempenho do RRT* é superior ao do RRT*-Sukharev (gráfico do primeiro sempre acima do gráfico referente ao segundo em todos os tamanhos de grade).

Nos gráficos que relacionam a taxa de sucesso com o tempo de planejamento (Figura 4.8), o uso de vértices convexos sem grade de Sukharev (RRT*-Vértices) apresenta desempenho melhor do que com grade (RRT*-Híbrido), pois o gráfico da primeira abordagem sempre se encontra acima da segunda para todos os valores de Δs. Além disso, o algoritmo RRT* apresenta melhor desempenho de tempo de planejamento do que sua versão com o uso de grade de Sukharev para todos os valores de Δs no ambiente de navegação com 50 obstáculos.

Figura 4.7 - Taxa de sucesso de cada algoritmo em alcançar frações da rota ótima em 100 simulações com sementes distintas para geração de número pseudoaleatório para o ambiente de navegação com 50 obstáculos.

Gráficos para Δs com a) d64, b) d128, c) d256, d) d512 e e) d1024.

Fonte: Produção do Autor.

Figura 4.8 - Taxa de sucesso de cada algoritmo em planejar rotas por tempo de planejamento em segundos (em escala de log_{10}) em 100 simulações com sementes distintas para geração de número pseudoaleatório para o ambiente de navegação com 50 obstáculos.

Gráficos para Δs com a) d64, b) d128, c) d256, d) d512 e e) d1024. Todos os gráficos são referentes a cinco segundos de planejamento.

Fonte: Produção do Autor.

Na Figura 4.9 e na Figura 4.10, encontram-se ilustrados os gráficos relativos ao ambiente de navegação com 100 obstáculos. Na Figura 4.9, observa-se que os algoritmos RRT*-Vértices e RRT*-Híbrido possuem taxas de sucesso similares em alcançar as menores frações da rota ótima. Entretanto, o RRT*-Vértices apresenta ligeira superioridade sobre o RRT*-Híbrido, pois, o gráfico do primeiro encontra-se ligeiramente acima. Já o RRT* obteve melhor taxa de sucesso em relação ao algoritmo RRT*-Sukharev, exceto na Figura 4.10(d) referente ao Δs de d512 e na Figura 4.10(e) referente ao Δs de d1024, onde o gráfico do RRT*-Sukharev, a partir da fração 1,05, se posiciona acima do gráfico do RRT*.

Nos gráficos que relacionam a taxa de sucesso com o tempo de planejamento (Figura 4.10), os resultados foram semelhantes aos apresentados em relação ao ambiente de navegação com 50 obstáculos (Figura 4.8). O uso de vértices convexos sem grade de Sukharev (RRT*-Vértices) continua apresentando desempenho melhor do que com grade (RRT*-Híbrido). Entre os algoritmos RRT* e RRT*-Sukharev, o primeiro continua apresentando melhor desempenho do que o segundo neste ambiente de navegação com 100 obstáculos. Dessa forma, percebe-se que mesmo dobrando o número de obstáculos e considerando diferentes valores de Δs, a grade de Sukharev continua não demonstrando melhoria quando há obstáculos espalhados pelo ambiente de navegação.

Figura 4.9 - Taxa de sucesso de cada algoritmo em alcançar frações da rota ótima em 100 simulações com sementes distintas para geração de número pseudoaleatório para o ambiente de navegação com 100 obstáculos.

Gráficos para Δs com a) d64, b) d128, c) d256, d) d512 e e) d1024. Todos os gráficos são referentes a cinco segundos de planejamento.

Fonte: Produção do Autor.

Figura 4.10 - Taxa de sucesso de cada algoritmo em planejar rotas por tempo de planejamento em segundos (em escala de log_{10}) em 100 simulações com sementes distintas para geração de número pseudoaleatório para o ambiente de navegação com 100 obstáculos.

Gráficos para Δs com a) d64, b) d128, c) d256, d) d512 e e) d1024. Todos os gráficos são referentes a cinco segundos de planejamento.

Fonte: Produção do Autor.

Na Figura 4.11 e na Figura 4.12, encontram-se ilustrados os gráficos relativos ao ambiente de navegação com obstáculos em zigue-zague. Nos gráficos da Figura 4.11, o RRT*-Vértices e o RRT*-Híbrido mantém taxa de sucesso acima de 80% até a fração de rota ótima com valor igual a $1,05$, portanto, muito superior do que as demais, que ficam abaixo de 20% a partir do valor de fração igual a $1,1$. O desempenho do RRT* em relação ao RRT*-Sukharev inverte dependendo do tamanho da grade, porém, apresentam desempenho equivalente.

Na Figura 4.12, o RRT*-Vértices apresentam o mesmo desempenho comparado ao RRT*-Híbrido. Já os algoritmos RRT* e RRT*-Sukharev apresentam desempenho levemente inferior comparados a suas versões com vértices convexos. Entretanto, com a diminuição de Δs (aumento da quantidade de células na grade de Sukharev) essa diferença aumenta. Adicionalmente, para o ambiente de navegação em zigue-zague, o algoritmo RRT*-Sukharev apresenta desempenho superior ao algoritmo RRT*.

Figura 4.11 - Taxa de sucesso de cada algoritmo em alcançar frações da rota ótima em 100 simulações com sementes distintas para geração de número pseudoaleatório para o ambiente de navegação com obstáculos em zigue-zague.

Gráficos para Δs com a) d64, b) d128, c) d256, d) d512 e e) d1024.

Fonte: Produção do Autor.

Figura 4.12 - Taxa de sucesso de cada algoritmo em planejar rotas por tempo de planejamento em segundos (em escala de log_{10}) em em 100 simulações com sementes distintas para geração de número pseudoaleatório para o ambiente de navegação com obstáculos em zigue-zague.

Gráficos para Δs com a) d64, b) d128, c) d256, d) d512 e e) d1024. Todos os gráficos são referentes a cinco segundos de planejamento.

Fonte: Produção do Autor.

Na Figura 4.13 e na Figura 4.14, encontram-se ilustrados os gráficos relativos ao ambiente de navegação com formato em espiral. Na Figura 4.13, os gráficos RRT*-Híbrido se mantém acima do gráfico do algoritmo RRT*-Vértices. Os algoritmos RRT* e RRT*-Sukharev apresentam desempenho equivalente em todos os valores de Δs considerados.

Em relação ao sucesso em planejar alguma rota por tempo de planejamento, ilustrado na Figura 4.14, o algoritmo RRT*-Híbrido obteve o melhor desempenho comparado aos demais algoritmos, por exceção do gráfico que representa o Δs de d1024, ilustrado na Figura 4.14(e), onde há momentos que o gráfico do RRT*-Vértice se sobrepõe aos demais. Adicionalmente, o RRT*-Sukharev apresenta maior sucesso em planejar rotas nos primeiros milésimos de planejamento em relação ao RRT*, porém, com valores de tempo de planejamento de rota maiores, o RRT* planeja rotas mais vezes do que o RRT*-Sukharev.

Figura 4.13 - Taxa de sucesso de cada algoritmo em alcançar frações da rota ótima em 100 simulações com sementes distintas para geração de número pseudoaleatório para o ambiente de navegação com obstáculo com formato em espiral.

Gráficos para Δs com a) d64, b) d128, c) d256, d) d512 e e) d1024.

Fonte: Produção do Autor.

Figura 4.14 - Taxa de sucesso de cada algoritmo em planejar rotas por tempo de planejamento em segundos (em escala de log_{10}) em 100 simulações com sementes distintas para geração de número pseudoaleatório para o ambiente de navegação com obstáculo com formato em espiral.

Gráficos para Δs com a) d64, b) d128, c) d256, d) d512 e e) d1024. Todos os gráficos são referentes à cinco segundos de planejamento.

Fonte: Produção do Autor.

Na Figura 4.15 e na Figura 4.16, encontram-se ilustrados os gráficos relativos ao ambiente de navegação com passagem estreita. Na Figura 4.15, o algoritmo RRT*-Vértices e RRT*-Híbrido mantém alta taxa de sucesso, porém, não atingem as menores frações da rota ótima, com taxa de sucesso decaindo até 0 quando o valor de fração aproxima-se de 1. O desempenho do algoritmo RRT* é superior em relação ao algoritmo RRT*-Sukharev até valores de frações de rota igual a $1,1$ para os gráficos de Δs referente à d64 (Figura 4.15(c)) e d128 (Figura 4.15(b)). Após esse valor a RRT*-Sukharev apresenta maior taxa de sucesso para planejar rotas de menor comprimento.

Com base nos gráficos da Figura 4.16, nota-se que os algoritmos RRT*-Vértices e RRT*-Híbrido apresentam 100% de sucesso em planejar rotas durante todo o tempo de planejamento considerado. Entretanto, o algoritmo RRT*-Híbrido apresenta uma taxa de sucesso levemente menor nos valores iniciais de tempo de planejamento. O algoritmo RRT* apresenta melhor desempenho do que o algoritmo RRT*-Sukharev, já que a linha de seu gráfico se apresenta sempre acima em todos os valores de Δs considerados.

Figura 4.15 - Taxa de sucesso de cada algoritmo em alcançar frações da rota ótima em 100 simulações com sementes distintas para geração de número pseudoaleatório para o ambiente de navegação com passagem estreita.

Gráficos para Δs com a) d64, b) d128, c) d256, d) d512 e e) d1024.

Fonte: Produção do Autor.

Figura 4.16 - Taxa de sucesso de cada algoritmo em planejar rotas por tempo de planejamento em segundos (em escala de log_{10}) em 100 simulações com sementes distintas para geração de número pseudoaleatório para o ambiente de navegação com passagem estreita.

Gráficos para Δs com a) d64, b) d128, c) d256, d) d512 e e) d1024. Todos os gráficos são referentes a cinco segundos de planejamento

Fonte: Produção do Autor.

Na Figura 4.17 estão ilustrados os gráficos apresentando as taxas de sucesso por fração de rota de cada quantidade de célula adotada em RRT*-Sukharev por ambiente de navegação. Observa-se que o gráfico relativo a grade com 1024 células encontra-se acima dos demais em três ambientes de navegação: com 50 obstáculos (Figura 4.17(a)), com 100 obstáculos (Figura 4.17(b)) e com passagem estreita (Figura 4.17(e)). No ambiente de navegação com obstáculos em zigue-zague (Figura 4.17(c)), o RRT*-Sukharev com 128 células apresentou o melhor desempenho e no ambiente de navegação em espiral (Figura 4.17(d)), todos os tamanhos de grade obtiveram desempenho equivalente.

Figura 4.17 - Taxa de sucesso dos algoritmos com amostragem baseada exclusivamente na grade de Sukharev (RRT*-Sukharev) com diferentes quantidade de células em alcançar frações da rota ótima em 100 simulações com sementes distintas para geração de número pseudoaleatório para diferentes ambientes de navegação.

Os ambientes de navegação são: a) com 50 obstáculos; b) com 100 obstáculos; c) com obstáculos em zigue-zague; d) com obstáculo com formato em espiral; e) com passagem estreita.

Fonte: Produção do Autor.

Considerando o conjunto de testes computacionais desta seção, podemos inferir as seguintes conclusões:

a) O uso de vértices convexos no RRT* é superior no planejamento de rota em relação a grade de Sukharev e o RRT* com amostragem exclusivamente uniforme;

b) Dependendo do número de células, a grade de Sukharev nem sempre garante melhor convergência para a rota ótima no algoritmo RRT*;

c) Quando há melhoria no desempenho com o uso da grade de Sukharev, essa melhoria não é tão significativa;

d) A distribuição dos obstáculos e o seu formato influencia diretamente no desempenho dos algoritmos, e um determinado valor de Δs (para uma dada quantidade de células) utilizado em um ambiente de navegação não garante bom desempenho em outro. Esses fatores também alteram a eficiência da grade de Sukharev em melhorar o desempenho de planejamento do RRT*;

e) O RRT*-Vértices aparentemente não necessita de calibração de Δq para obter uma melhor performance pois a mesma é constante nos diferentes tamanhos de seu valor, sendo então um algoritmo mais robusto do que os demais com um desempenho computacional muito superior.

4.4 RRT* com processo de amostragem híbrido baseado em Grades de Sukharev e Vértices Convexos: Algoritmo RRT*-SV

Nesta seção é introduzido o algoritmo proposto nesta tese denominado RRT*-Sukharev Vértices (RRT*-SV). Esse algoritmo encontra-se publicado em Véras et al. (2019a). O algoritmo RRT*-SV utiliza a grade de Sukharev e os vértices convexos das envoltórias de segurança do ambiente de navegação para acelerar o planejamento de uma rota ótima em ambientes de navegação com obstáculos estáticos. O algoritmo RRT*-SV é a versão consolidada do algoritmo RRT*-Híbrido abordado nos experimentos da Seção 4.3, onde um processo de otimização de rota é adicionado para acelerar a redução do comprimento da rota. O processo de otimização de rota é o mesmo utilizado pelo algoritmo RRT*-Smart e encontra-se descrito na Seção A.1. Um fluxograma do comportamento geral do algoritmo RRT*-SV está ilustrado na Figura 4.18.

Figura 4.18 - Fluxograma do comportamento geral do algoritmo RRT*-SV.

Fonte: Véras et al. (2019a).

Os principais procedimentos do Algoritmo RRT*-SV, apresentado no Algoritmo 8, são descritos a seguir.

a) **VÉRTICES_CONVEXOS**: obtém todos os vértices convexos do ambiente de navegação e os armazena em $Q_{vértices}$;

b) **AMOSTRAGEM_VÉRTICES**: retorna o vértice convexo que gera uma aresta livre de colisão com $q_{próximo}$ (Algoritmo 7);

c) **VÉRTICE_PRÓXIMO**: procura o vértice convexo mais próximo de $q_{próximo}$;

d) **DESABILITA_VÉRTICE**: desabilita para seleção em $Q_{vértices}$ o vértice selecionado como q_{novo}, o que não o permite mais ser conectado a nenhum outro nó;

e) **AMOSTRAGEM_SUKHAREV**: gera uma nova amostra do ambiente de navegação usando a Grade de Sukharev (Algoritmo 5);

f) **PASSO_SUKHAREV**: retorna o valor da menor distância entre os centroides das células da grade de Sukharev para a geração de q_{novo_temp};

g) **CENTROIDE_SUKHAREV**: retorna o centroide da célula que contém q_{novo_temp}, caso forme uma aresta livre de colisão com o mesmo.

No processo de amostragem do Algoritmo 8 (RRT*-SV), três tentativas estão previstas para a coleta de uma nova amostra a ser inserida na árvore. Na primeira, tenta-se gerar q_{novo} a partir dos vértices convexos das envoltórias de segurança dos obstáculos do ambiente de navegação (linha 16). Se isto não for possível, uma nova tentativa é realizada com os centroides da grade de Sukharev (linha 20). Este cenário encontra-se ilustrado na Figura 4.19. Finalmente, se não for possível gerar um novo nó para a árvore após as duas tentativas anteriores, uma amostra aleatória com distribuição uniforme é gerada, como no algoritmo padrão RRT (linha 22). Este cenário encontra-se ilustrado na Figura 4.20. Se após três tentativas um novo nó não for encontrado, um novo $q_{aleatório}$ é coletado do ambiente de navegação e uma nova iteração é executada.

Mais detalhes sobre o uso de cada uma das estratégias propostas no algoritmo RRT*-SV são fornecidos nas subseções a seguir. As melhorias esperadas pelo uso de cada uma no processo de amostragem também são apresentadas.

Algorithm 8 Algoritmo RRT*-SV

```
1: procedure RRT*-SV(Q, q_início, q_destino, Δq, l_d, n, k)
2:     G ← {q_início}
3:     R ← {}
4:     S ← GRADE_SUKHAREV(k)
5:     i ← 0
6:     s ← 0
7:     comprimento_antigo ← ∞
8:     Δs ← PASSO_SUKHAREV(S)
9:     Q_vértices ← VÉRTICES_CONVEXOS(Q_obstáculos)
10:    while (i ≤ n) do
11:        i ← i + 1
12:        por_vertice ← 0
13:        q_novo ← {}
14:        q_aleatório ← AMOSTRAGEM_ALEATÓRIA(Q)
15:        q_próximo ← NÓ_PRÓXIMO(q_aleatório, G)
16:        q_novo ← AMOSTRAGEM_VÉRTICES(q_próximo, Q_vértices)
17:        if q_novo ≠ {} then
18:            por_vertice ← 1
19:        if q_novo = {} then
20:            q_novo ← AMOSTRAGEM_SUKHAREV(q_próximo, S, Δs)
21:        if q_novo = {} then
22:            q_novo ← NOVO_NÓ(q_próximo, q_aleatório, Δq)
23:        if q_próximo q_novo não intercepta Q_obstáculos then
24:            RRT_ESTRELA_ESTENDE(G, q_próximo, q_novo)
25:            if por_vertice = 1 ou d(q_novo, q_destino) ≤ l_d then
26:                if (q_novo q_destino não intercepta Q_obstáculos) then
27:                    ESTENDE(G, q_novo, q_destino)
28:                    rota_encontrada ← verdade
29:            if rota_encontrada then
30:                comprimento_novo ← OTIMIZAÇÃO_DE_ROTA(q_início, q_destino)
31:                if comprimento_novo < comprimento_antigo then
32:                    R ← ROTA(q_início, q_destino)
33:                    comprimento_antigo ← comprimento_novo
34:            rota_encontrada ← false
```

Figura 4.19 - Etapas do processo de amostragem baseada na grade de Sukharev executadas quando não há nenhum vértice convexo disponível para adição à árvore.

a) $q_{aleatório}$ é gerado aleatoriamente a cada iteração; b) o nó $q_{próximo}$ de G mais próximo de $q_{aleatório}$ é selecionado; c) um nó temporário q_{novo_temp} é gerado a uma distância Δs de $q_{próximo}$; ao centroide da célula contendo q_{novo_temp} é adicionado como novo nó q_{novo} da árvore, se o segmento $\overline{(q_{próximo} q_{novo})}$ é livre de colisão. Os pontos na cor preta são vértice convexos. Os quadrados na cor marrom são centroides gerados pela grade de Sukharev.

Fonte: Véras et al. (2019a).

Figura 4.20 - Etapas do processo de amostragem baseada em amostragem uniforme executadas quando não há nenhum centroide da grade de Sukharev e nenhum vértice convexo para adição à árvore.

a) Em cenários onde os centroides da grade de Sukharev e os vértices convexos não estão disponíveis para estender a árvore, a amostragem uniforme do algoritmo RRT padrão é executada no RRT*-SV. b) Similar ao que ocorre no RRT padrão, inicialmente uma amostra $q_{aleatório}$ é gerado e então uma nova amostra q_{novo} é gerada dada a distância Δq do seu nó mais próximo na árvore $q_{próximo}$. c) Se o segmento $\overline{q_{próximo} q_{novo}}$ for livre de colisão, então q_{novo} é adicionado à árvore.

Fonte: Véras et al. (2019a).

4.4.1 Detecção de colisão no algoritmo RRT*-SV

Certas situações de colisão em abordagens tradicionais de planejamento de rota são desconsideradas pelo algoritmo RRT*-SV. Essa estratégia é necessária para que seja possível planejar rotas que coincidam com os limites que definem uma envoltória de segurança de um obstáculo, o que significa que o comprimento da rota será minimizado.

No algoritmo RRT*-SV, uma aresta formada por vértices convexos sequenciais da mesma envoltória de segurança de um obstáculo coincide espacialmente com uma de suas bordas. Normalmente, essa situação seria considerada uma situação de colisão, porém o RRT*-SV a interpreta como sendo livre de colisão. No entanto, quando dois vértices convexos da envoltória de segurança não são sequenciais, a aresta por esses dois vértices pode atravessar/interceptar a própria envoltória de segurança do obstáculo ou outra envoltória do ambiente de navegação, e possivelmente também o próprio obstáculo a partir do qual uma envoltória foi gerada. Esta situação é interpretada como uma colisão. Assim, de forma concisa, duas situações geralmente interpretadas como sendo colisão em métodos de planejamento de rota não são pelo RRT*-SV: quando um dos vértices convexos da envoltória de segurança de um obstáculo coincide espacialmente com um dos vértices nós da árvore; e quando uma das bordas da envoltória de segurança formada por dois vértices consecutivos coincidem espacialmente com uma aresta da árvore. A estrutura e as relações entre o ambiente de navegação, o teste de colisão e as estratégias de amostragem no algoritmo RRT*-SV estão definidos no diagrama de bloco, ilustrado na Figura 4.21.

Figura 4.21 - Diagrama de bloco da estrutura de planejamento de rota utilizada pelo algoritmo RRT*-SV.

Fonte: Véras et al. (2019a).

4.4.2 Análise assintótica da complexidade de tempo do algoritmo RRT*-SV

O cálculo dessa complexidade foi baseado nas propriedades do domínio assintótico descritas em Ziviani (2004). A complexidade do tempo computacional do algoritmo RRT*-SV baseia-se na complexidade da linha 9 e na complexidade do laço da linha 10 do Algoritmo 8, pois a geração da grade de Sukharev, realizada pela operação $GRADE_SUKHAREV$ na linha 4, é $O(1)$, pois o que é realmente gerado é o valor de Δs da grade. Usando os tamanhos do eixo do ambiente de navegação, as células da grade de Sukharev podem ser calculadas diretamente através da sua quantidade de células. A complexidade da linha 9 é $O(n_t)$, onde n_t é o número total de vértices considerando todas as envoltórias de segurança do ambiente de navegação. No pior dos casos, a complexidade do laço da linha 10 corresponde à multiplicação de n (número total de nós da árvore) pela soma da complexidade de tempo das operações $NÓ_PRÓXIMO$, $AMOSTRAGEM_VÉRTICES$, $RRT_ESTRELA_ESTENDE$ e $OTIMIZAÇÃO_DE_ROTA$. Os outros componentes do laço têm complexidade $O(1)$. O primeiro processo com tempo computacional significativo dentro deste laço é a busca pelo nó mais próximo de $q_{aleatório}$ (Linha 15 do Algoritmo 8). Pode ser encontrado em ordem logarítmica (ARYA; MOUNT,

2000), mas neste trabalho foi utilizada a estratégia de força bruta para a busca do nó mais próximo (RRT*, RRT*-Smart, Informed-RRT, BIT*, RRT*-SV). Portanto, esse processo é $O(n)$.

A complexidade de tempo de $AMOSTRAGEM_VÉRTICES$ (Linha 16 do Algoritmo 8) é igual à soma da complexidade de $VÉRTICE_PRÓXIMO$ com a complexidade do teste de colisão. Como n_v é a quantidade de vértices convexos em $Q_{vértices}$, o procedimento $VÉRTICE_PRÓXIMO$ toma $O(log\, n_v)$ em média e $O(n_v)$ no pior caso utilizando uma estrutura k-d-tree. O procedimento $DESABILITA_VÉRTICE$ não remove o vértice de $Q_{vértices}$, mas altera as suas coordenadas para valores fora do intervalo que definem o ambiente de navegação, então é $O(1)$, evitando o custo de removê-lo da árvore k-d-tree. Portanto, $AMOSTRAGEM_VÉRTICES$ custaria $O(n_v) + O_{colisão}$ em cada iteração do algoritmo RRT*-SV.

O procedimento $RRT_ESTRELA_ESTENDE$ (Linha 24 do Algoritmo 8) é baseado na estimativa da vizinhança de q_{novo}. A quantidade de vizinhos de q_{novo} é uma parte da quantidade de nós na árvore. Assim, a complexidade de tempo de $RRT_ESTRELA_ESTENDE$ é $O(n\, O_{colisão})$, no pior caso.

O procedimento $OTIMIZAÇÃO_DE_ROTA$ (Linha 30 do Algoritmo 8) consiste em simplificar uma rota composta por uma porção dos nós da árvore do algoritmo RRT, assim como no algoritmo RRT*-Smart, descrito na Seção A.1. Então, a complexidade de tempo de $OTIMIZAÇÂO_DE_ROTA$ pode ser calculada para ser igual a $O(n\, O_{colisão})$.

Finalmente, a complexidade de tempo total do algoritmo RRT*-SV é $O(O(n_t) + n\,(O(n) + (O(n_v) + O_{colisão}) + O(n\, O_{colisão}) + O(n\, O_{colisão}))) = O(O(n_t) + n\,(O(n\, O_{colisão}) + O(n_v) + O_{colisão})) = O(max(O(n_t), n\,(max(O(n\, O_{colisão}), O(n_v)))))$, no pior caso.

Em comparação com os algoritmos RRT* e RRT*-Smart que possuem complexidade computacional igual a $O(n^2)$ (KARAMAN; FRAZZOLI, 2010; ISLAM et al., 2012) (originalmente $O(n\, logn)$ devido ao uso de algoritmo otimizado de busca de nó mais próximo), o algoritmo RRT*-SV pode demonstrar uma complexidade de tempo diferente dependendo do valor de seu parâmetro adicional n_v. Se $O(n_v)$ for menor que $O(n)$, o RRT*-SV deve mostrar um desempenho melhor que os demais algoritmos. Caso contrário, RRT* e RRT*-Smart devem apresentar um desempenho melhor. Os testes computacionais nas próximas seções mostram esse comportamento esperado

do tempo de planejamento. Os trabalhos originais dos algoritmos Informed-RRT* e BIT* em Gammell et al. (2014) e (GAMMELL et al., 2015), respectivamente, não apresentam suas respectivas análises de complexidade de tempo. Entretanto, pelo menos o algoritmo Informed-RRT*, possui estrutura similar ao algoritmo RRT*. Dessa forma, parece razoável assumir que o mesmo apresenta complexidade próxima de $O(n\,logn)$, o que permite que a análise desenvolvida neste parágrafo possa também ser aplicada a esse algoritmo. O algoritmo BIT* segue uma estrutura diferente do RRT*, portanto não é apresentada nenhuma comparação do ponto de vista de análise assintótica com o RRT*-SV.

4.5 Testes computacionais com o algoritmo RRT*-SV

Três conjuntos de experimentos foram realizados visando a comparação entre os algoritmos RRT*, RRT*-Smart, Informed-RRT*, BIT* e RRT*-SV. Os algoritmos RRT*-Smart, Informed-RRT* e BIT* encontram-se descritos no Anexo A. Os cinco algoritmos foram aplicados a conjuntos de testes computacionais e comparados considerando o tempo de planejamento e o comprimento das rotas planejadas. Cada conjunto de teste computacional foi elaborado considerando ambientes de navegação bidimensionais com obstáculos estáticos. Entretanto, os algoritmos podem ser aplicados a ambientes de navegação em três dimensões com poucas adaptações baseadas na altitude de navegação do VANT. Os principais parâmetros dos algoritmos estão descritos na Subseção 4.5.1. Os três conjuntos de experimentos estão descritos nas Subseções 4.5.2, 4.5.3 e 4.5.4. Em todas as ilustrações de experimentos, os polígonos azuis representam as envoltórias de segurança dos obstáculos dos respectivos ambientes de navegação.

4.5.1 Parâmetros dos algoritmos

Esta subseção descreve os valores dos parâmetros dos algoritmos usados em todos os experimentos. O parâmetro Δq foi definido como 3% do maior valor entre o comprimento e a altura/largura do ambiente de navegação. A distância l_d que define a distância para a verificação de conexão de q_{novo} a $q_{destino}$ foi definida como 50. O valor seis foi definido para o parâmetro b que define a frequência de uso dos *beacons* gerados pela amostragem inteligente do algoritmo RRT*-Smart. O raio r_{beacon} para a geração de novos nós ao redor de um *beacon* é igual a 15. O parâmetro m que define a quantidade de amostras do *batch* de amostragem no algoritmo BIT* é igual a 100. Ainda no algoritmo BIT*, a sua operação $PODA$ é executada toda vez que uma nova rota com comprimento de diferença maior que 1% do que a anterior é planejada, de acordo com os parâmetros utilizados no algoritmo em Gammell et al.

(2015). O parâmetro k que define o número de células da grade de Sukharev usadas pelo algoritmo RRT*-SV é igual a 100. Essa quantidade de células foi selecionada por resultar em um valor de Δs próximo do valor Δq considerado nos testes. O valor β, que é usado para calcular o raio de busca de vizinhança de $q_{próximo}$ nos algoritmos RRT*, RRT*-Smart, Informed-RRT* e RRT*-SV, foi definido como 650.

Os parâmetros foram definidos baseados em experimentações e nos valores utilizados nos trabalhos originais de cada algoritmo. O parâmetro β do algoritmo RRT* (parâmetro o qual é utilizado por todos os algoritmos de planejamento abordados nesta tese), foi definido de forma empírica. Diversos valores de β foram verificados e aquele considerado não acelerar nem desacelerar demais o planejamento do RRT* foi utilizado nos experimentos. No caso do algoritmo RRT*-SV, foram experimentados diversos valores de b para identificar aquele que mais acelera o planejamento do algoritmo. Entretanto, o desempenho do algoritmo configurado com um dado valor de b varia de acordo com o ambiente de navegação. Os valores indicados como ideais em Islam et al. (2012) também foram utilizados como referência para a definição de b. O Informed-RRT* não possui configurações adicionais em comparação ao RRT*. Os valores dos parâmetros do BIT* foram obtidos de Gammell et al. (2015).

4.5.2 Primeiro conjunto de testes computacionais

Algumas considerações foram feitas no primeiro conjunto de testes computacionais. Um ambiente de navegação de um único obstáculo é utilizado neste experimento, como ilustrado na Figura 4.22. O ambiente de navegação foi representado em um plano cartesiano bidimensional, onde: o comprimento do mesmo corresponde ao intervalo $[0, 0, 1000, 0]$ no eixo horizontal (eixo x); e a largura corresponde ao intervalo $[0, 0, 1000, 0]$ no eixo vertical (eixo y). Todas as rotas foram planejadas entre a posição inicial $(500, 500)$ e a posição final $(925, 925)$, que correspondem a $q_{início}$ e $q_{destino}$ respectivamente. Nos três algoritmos, foi usada uma condição de parada de 30 000 iterações, e a mesma semente para gerar números pseudo-aleatórios foi usada na função aleatória para gerar as amostras $q_{aleatório}$ em cada simulação. Um total de 100 simulações foram executadas para este experimento. O valor médio do comprimento de rota planejada nas simulações por cada algoritmo foi comparado à rota de comprimento ótimo do ambiente de navegação deste experimento, o qual corresponde ao valor 1246. Esse valor foi obtido a partir de rota planejada pelo algoritmo Dijkstra em uma estrutura de mapa de rotas (*roadmap*) construída pelo algoritmo de grafo de visibilidade. Uma ilustração da rota ótima do ambiente de navegação considerado neste primeiro conjunto de testes computacionais encontra-se no Apêndice A.

Figura 4.22 - Primeiro conjunto de testes computacionais: Primeira rota planejada pelos algoritmos.

Primeira rota planejada pelo a) RRT* na iteração 1371, b) RRT*-Smart na iteração 1371, com versões otimizada (em vermelho) e não otimizada (em preto), c) Informed-RRT* na iteração 1371, d) BIT* na iteração 646 e e) RRT*-SV na iteração 10, com versão otimizada (em vermelho).

Fonte: Produção do Autor.

Na Figura 4.22 encontram-se ilustradas as primeiras rotas planejadas por cada algoritmo para uma das sementes utilizadas nas simulações. Observando a figura citada e considerando essa única semente, é possível notar que a rota planejada pelo algoritmo RRT*-SV possui um comprimento menor do que as rotas planejadas pelos outros algoritmos. Os algoritmos RRT*, RRT*-Smart e Informed-RRT* planejaram a primeira rota apenas na iteração 1371. O comprimento da rota para o RRT* foi 1601, para o RRT*-Smart foi 1463 e para o Informed RRT* foi 1601.

Esses três algoritmos planejaram a primeira rota na mesma iteração porque usam o mesmo mecanismo de amostragem até o encontro da primeira rota. No entanto, após o planejamento da primeira rota, o algoritmo RRT*-Smart usa uma estratégia diferente, baseada na otimização da rota com o uso de *beacons*, conforme descrito no Anexo A.1. O algoritmo Informed-RRT* também utiliza outro processo de amostragem após o planejamento da primeira rota, o qual é baseado em uma estratégia de amostragem informada, onde novas amostras devem ser coletadas dentro de uma região em formato de elipse definida pelo comprimento da rota corrente na simulação (verificar Anexo A.2 para mais detalhes).

O algoritmo BIT*, que planejou uma rota em menos iterações do que os algoritmos anteriores (a primeira rota foi encontrada na iteração 646 com comprimento igual a 1471), utiliza também uma amostragem informada, como o algoritmo Informed-RRT*. Porém, utiliza uma estrutura baseada na RGG (*Random Geometrical Graph*) e uma heurística de busca de menor rota baseada na estimativa entre os pontos dessa RGG, retardando a verificação de colisão apenas para as situações em que se conhece as arestas mais promissoras em formar a rota de menor comprimento. Para mais detalhes sobre o algoritmo BIT* o Anexo A.3 deve ser consultado. Já o algoritmo RRT*-SV planejou uma rota com comprimento igual a 1246 na iteração 10. Essa rota corresponde à solução ótima, como definido no parágrafo anterior.

Na Figura 4.23 encontram-se ilustradas as rotas planejadas por cada algoritmo ao final de 30 000 iterações para uma das sementes utilizadas nas simulações. Nota-se que as rotas planejadas por cada algoritmo possuem formatos semelhantes, indicando que todos os algoritmos de fato convergem para a rota de menor comprimento. Entretanto, nota-se também que algumas rotas planejadas possuem uma quantidade de nós muito maior do que as demais, como é o caso dos algoritmos RRT* e Informed-RRT*. Nos algoritmos RRT*-Smart e RRT*-SV a quantidade de nós das rotas planejadas é reduzida devido ao procedimento de otimização de rota utilizado por ambos. A rota planejada pelo algoritmo BIT* é a que menos se aproxima da

rota ótima, pois encontra-se demasiadamente afastada dos limites das envoltórias de segurança. Entretanto, esse algoritmo foi o que executou as 30 000 iterações em menor tempo de planejamento (o seu tempo de planejamento foi de $0,221s$ enquanto os dos demais foi em média $30s$), e provavelmente se aproximaria rapidamente dela caso uma maior quantidade de iterações estivesse disponível.

Figura 4.23 - Primeiro conjunto de testes computacionais: Rotas planejadas após 30 000 iterações pelos algoritmos.

Rotas planejadas após 30 000 iterações pelos algoritmos a) RRT*, b) RRT*-Smart, com versões otimizada (em vermelho) e não otimizada (em preto), c) Informed-RRT*, d) BIT* e e) RRT*-SV, com versão otimizada (em vermelho).

Fonte: Produção do Autor.

A Tabela 4.1 apresenta os valores médios gerados por cada algoritmo para planejar a primeira rota considerando 100 simulações com diferentes sementes. O tempo de planejamento do RRT* é semelhante ao do RRT*-Smart. O primeiro precisou em média de 1251 iterações e $0,054s$ para planejar a primeira rota e o segundo também planejou com uma média de 1251 iterações e $0,059s$. A pequena diferença entre esses tempos de planejamento pode ser atribuída ao procedimento de otimização de rota adicional no algoritmo RRT*-Smart. A média do comprimento das rotas planejadas pelo RRT* foi 1384 e pelo RRT*-Smart foi menor com valor igual a 1320. Isso é esperado, já que o RRT*-Smart reduz o tamanho da rota por meio do processo de otimização de rota.

O algoritmo Informed-RRT* planejou suas primeiras rotas em média em 1251 iterações e tempo de planejamento igual à $0,037s$. A média do comprimento das rotas planejadas pelo Informed-RRT* é 1384, igual ao do RRT*. O algoritmo BIT* planejou uma primeira rota em médio em 658 iterações após $0,007s$, em média, com comprimento médio igual a 1429. O comprimento médio das rotas planejadas pelo BIT* é maior que os das rotas planejadas pelos algoritmos citados anteriormente, porém, em média, elas foram planejadas em um tempo consideravelmente menor.

O algoritmo RRT*-SV planejou a primeira rota, em média, na iteração 14 com tempo de planejamento igual a $0,001s$ e com comprimento médio igual a 1254. Comparando os tempos de planejamento e os comprimentos de rota obtidos com cada algoritmo, o RRT*-SV foi mais rápido em 98% comparado ao RRT*, 99% ao RRT*-Smart, 97% ao Informed-RRT* e 87% ao BIT*, e ainda obteve uma rota 5% menor do que o algoritmo que apresentou a menor média de comprimento na primeira rota planejada, que foi o algoritmo RRT*-Smart. Nota-se então que, para este conjunto de testes computacionais, o algoritmo RRT*-SV proporciona um desempenho muito melhor para o planejamento de rota. Além disso, o algoritmo RRT*-SV é o único algoritmo que mantém a média do comprimento das rotas planejadas próxima do comprimento ótimo.

Na Figura 4.24 um gráfico do comprimento médio das rotas planejadas por cada algoritmo ao longo de 30 000 iterações está ilustrado. Pode-se observar que, na média, o RRT*, RRT*-Smart, Informed-RRT* e BIT* não puderam planejar a rota de menor comprimento, isto é, a solução ótima (linha da grade horizontal na posição 1246 no eixo Y), em 30 000 iterações. Por outro lado, o algoritmo RRT*-SV planejou uma rota com comprimento próximo do comprimento ótimo por volta da iteração 2000, na média.

Tabela 4.1 - Primeiro conjunto de testes computacionais: valores médios para 100 simulações para cada algoritmo planejar a primeira rota.

Algoritmo	**#Iterações**	**Comprimento**	**Tempo de planejamento (s)**
RRT*	1251	1384	$0,054$
RRT*-Smart	1251	1320	$0,059$
Informed-RRT*	1251	1384	$0,037$
BIT*	658	1429	$0,007$
RRT*-SV	14	1254	$0,001$

Fonte: Produção do Autor.

Figura 4.24 - Primeiro conjunto de experimentos: Comprimento médio de rotas planejadas em 100 simulações por iteração.

O comprimento da rota ótima corresponde à 1246.

Fonte: Produção do Autor.

Na Figura 4.25 está ilustrado o gráfico que representa as iterações necessárias, em média, para cada algoritmo planejar rotas com comprimentos específicos. Há uma grande diferença no número de iterações necessárias para cada algoritmo atingir diferentes valores de comprimento de rota. Enquanto os algoritmos comparados precisam ser executados com valores acima de 2000 iterações para produzir uma rota de comprimento igual a 1335, o RRT*-SV obteve rotas com esse comprimento em 42 iterações, em média. No entanto, também é necessário verificar o tempo computacional que cada algoritmo precisa para atingir cada valor específico de comprimento de rota, uma vez que esses algoritmos diferem significativamente na geração de uma nova amostra do ambiente de navegação. Isso pode levar a tempos de execução significativamente diferentes entre as iterações desses algoritmos.

Figura 4.25 - Primeiro conjunto de testes computacionais: Média da quantidade de iterações para 100 simulações para cada algoritmo encontrar rotas com comprimentos específicos.

Fonte: Produção do Autor.

Para evidenciar essa relação, é então ilustrado na Figura 4.26 o gráfico com tempos computacionais necessários para cada algoritmo para produzir rotas que alcancem ou excedam certos valores de comprimento específicos. Para comparação, considera-se o comprimento específico igual à 1248. O tempo de planejamento que o algoritmo RRT*-Smart leva para produzir uma rota de comprimento igual a 1248 foi de $26,387s$ contra $0,021s$ do RRT*-SV. Isto representa uma diferença de $99,9\%$ entre os tempos de planejamento. Os algoritmos Informed-RRT* e BIT* não planejaram rotas para o comprimento igual a 1248. Como também pode ser observado na Figura 4.24, nesses casos, o tempo de planejamento é definido como infinito. Pode-se concluir que o algoritmo RRT*-SV demonstra notória superioridade em relação ao algoritmo RRT*-Smart e aos algoritmos RRT*, Informed-RRT* e BIT*, que não foram capazes de planejar rotas com menores comprimentos considerados. Dessa forma, esta seção demonstra evidências da melhor convergência do planejamento pelo algoritmo RRT*-SV para uma rota de menor comprimento em menor tempo de planejamento comparado aos algoritmos considerados neste primeiro conjunto de experimentos.

Figura 4.26 - Primeiro conjunto de testes computacionais: Média de tempo de planejamento (em segundos) para 100 simulações para cada algoritmo encontrar rotas com comprimentos específicos.

Fonte: Produção do Autor.

4.5.3 Segundo conjunto de testes computacionais

O desempenho do algoritmo RRT*-SV depende da quantidade de vértices convexos contidos no ambiente de navegação, como demonstrado na Seção 4.4.2. Quanto maior a quantidade de vértices convexos, maior será o tempo de planejamento do RRT*-SV. Para verificar a influência do número de vértices convexos no desempenho no RRT*-SV comparado aos algoritmos RRT*, RRT*-Smart, Informed-RRT* e BIT*, foram realizadas simulações utilizando ambientes de navegação com cinco, 50, 2500 e 10 000 obstáculos. Cada obstáculo (e também sua envoltória de segurança) possui formato retangular, resultando em quatro vértices convexos por obstáculo. Os ambientes de navegação com obstáculos de cinco, 50, 2500 e 10 000 possuem, respectivamente, uma quantidade de vértices convexos iguais a 20, 200, 10 000 e 40 000. Para cada um deles, cinco sementes diferentes foram usadas para gerar os resultados no planejamento de simulações para 15 000, 40 000, 60 000 e 100 000 iterações, respectivamente a cada quantidade de obstáculos. O valor médio do comprimento da rota planejada e o tempo de planejamento das simulações foram utilizados para a análise do desempenho dos algoritmos. Rota planejadas por cada um dos algoritmos encontram-se ilustradas na Figura 4.27 para o ambiente de navegação com cinco obstáculos, na Figura 4.28 para o ambiente de navegação com 50 obstáculos, na Figura 4.29 para o ambiente de navegação com 2500 obstáculos e na Figura 4.30 para o ambiente de navegação com 10 000 obstáculos.

Conforme descrito na Tabela 4.2, o RRT*-SV foi capaz de planejar a primeira rota mais rápido do que os demais algoritmos em dois ambientes de navegação: com cinco obstáculos e 50 obstáculos. Nos demais ambientes, o RRT*-SV planejou a primeira rota com tempo de planejamento maior do que os demais. Quanto maior o número de vértices convexos, maior a diferença entre o algoritmo RRT*-SV e os outros algoritmos para planejar a primeira rota.

Figura 4.27 - Segundo conjunto de testes computacionais: Rotas planejadas para o ambiente de navegação com cinco obstáculos.

Rotas planejadas após dois segundos de execução dos algoritmos a) RRT*, b) RRT*-Smart, c) Informed-RRT*, d) BIT* e e) RRT*-SV.

Fonte: Produção do Autor.

Figura 4.28 - Segundo conjunto de testes computacionais: Rotas planejadas para o ambiente de navegação com 50 obstáculos.

Rotas planejadas após dois segundos de execução dos algoritmos a) RRT*, b) RRT*-Smart, c) Informed-RRT*, d) BIT* e e) RRT*-SV.

Fonte: Produção do Autor.

Figura 4.29 - Segundo conjunto de testes computacionais: Rotas planejadas para o ambiente de navegação com 2500 obstáculos.

Rotas planejadas após 30 segundos de execução dos algoritmos a) RRT*, b) RRT*-Smart, c) Informed-RRT*, d) BIT* e e) RRT*-SV.

Fonte: Produção do Autor.

Figura 4.30 - Segundo conjunto de testes computacionais: Rotas planejadas para o ambiente de navegação com 10 000 obstáculos.

Rotas planejadas após 400 segundos de execução dos algoritmos a) RRT*, b) RRT*-Smart, c) Informed-RRT*, d) BIT* e e) RRT*-SV.

Fonte: Produção do Autor.

Tabela 4.2 - Segundo conjunto de testes computacionais: Valores médios de comprimento e tempo relacionados à primeira rota planejada em cinco simulações para cada algoritmo em ambientes de navegação com diferentes quantidades de vértices convexos.

RRT*			**RRT*-Smart**			**Informed-RRT***			**BIT***			**RRT*-SV**		
Iteração	**Comprimento**	**Tempo de planejamento (s)**	**Iteração**	**Comprimento**	**Tempo de planejamento (s)**	**Iteração**	**Comprimento**	**Tempo de planejamento (s)**	**Iteração**	**Comprimento**	**Tempo de planejamento (s)**	**Iteração**	**Comprimento**	**Tempo de planejamento (s)**
cinco obstáculos (20 vértices convexos)														
1013	1509	0,023	1013	1461	0,018	1013	1509	0,05	267	1499	0,002	8	1781	0,001
50 obstáculos (200 vértices convexos)														
1859	1576	0,397	1859	1555	0,275	1859	1576	0,272	1496	1760	0,048	149	1522	0,014
2500 obstáculos (10 000 vértices convexos)														
2784	1575	10,264	2784	1572	8,072	2784	1575	7,808	14 625	1868	11,745	3289	1467	19,947
10 000 obstáculos (40 000 vértices convexos)														
11 356	1652	110,755	11 356	1650	111,837	11 356	1652	112,525	58 200	2045	172,765	8444	1873	267,007

Fonte: Produção do Autor.

No ambiente de navegação com cinco obstáculos, o algoritmo RRT*-SV planejou uma primeira rota, em média, a $0,001s$. O algoritmo RRT*-Smart planejou com média de $0,018s$ (diferença de 94% para o tempo do RRT*-SV), o Informed-RRT* com média de $0,050s$ (diferença de 98% para o tempo do RRT*-SV), e o algoritmo BIT* com $0,002s$ (diferença de 50% para o tempo do RRT*-SV).

No ambiente de navegação com 50 obstáculos, o RRT*-SV planeja a primeira rota a $0,014s$. O algoritmo RRT*-Smart planejou com média de $0,275s$ (diferença de 95% para o tempo do RRT*-SV), o Informed-RRT* com média de $0,272s$ (diferença de 95% para o tempo do RRT*-SV), e o algoritmo BIT* com $0,048s$ (diferença de 70% para o tempo do RRT*-SV).

No ambiente de navegação com 2500 obstáculos, o RRT*-SV planejou a primeira rota em $19,947s$. O algoritmo RRT*-Smart planejou com média de $8,072s$ (diferença de -60% para o tempo do RRT*-SV), o Informed-RRT* com média de $7,808s$ (diferença de -61% para o tempo do RRT*-SV), e o algoritmo BIT* com $11,745s$ (diferença de -41% para o tempo do RRT*-SV). Então, desta vez os demais algoritmos obtiveram vantagem de tempo de planejamento sobre o RRT*-SV para encontrar a primeira rota.

A desvantagem de tempo do RRT*-SV continua ainda no ambiente de navegação com 10 000 obstáculos. O mesmo planejou a primeira rota, em média, em $267,007s$. O algoritmo RRT*-Smart planejou com média de $111,837s$ (diferença de -58% para o tempo do RRT*-SV), o Informed-RRT* com média de $112,525s$ (diferença de -58% para o tempo do RRT*-SV), e o algoritmo BIT* com $172,765s$ (diferença de -35% com base no tempo do RRT*-SV).

Nota-se também que o algoritmo BIT* que, de acordo com Gammell et al. (2015), foi desenvolvido para apresentar melhor desempenho do que o algoritmo Informed-RRT*, apresenta maior tempo de planejamento nos ambientes de navegação com 2500 e 10 000 obstáculos. No entanto, apesar do pior desempenho de tempo de planejamento apresentado pelo RRT*-SV em ambientes de navegação com maior quantidade de obstáculos, é importante verificar a média do comprimento da rota planejada por cada algoritmo.

Nas Figuras 4.31, 4.32, 4.33 e 4.34, encontram-se ilustrados gráficos dos tempos de planejamento necessários para cada algoritmo produzir rotas que alcancem ou excedam valores de comprimento específicos para cada ambiente de navegação no segundo conjunto de testes computacionais. Caso algum algoritmo não consiga pla-

nejar rotas com algum dos comprimentos especificados, o tempo de planejamento é definido como infinito (∞). Na Figura 4.31 o RRT*-SV apresenta menor tempo de planejamento que os demais algoritmos para atingir rotas com comprimentos específicos. O mesmo ocorre na Figura 4.32, onde RRT*-SV também demonstra melhor desempenho no tempo de planejamento. Na Figura 4.33 e Figura 4.34, pode ser verificado que o RRT*-SV só demonstra vantagem sobre alguns dos outros algoritmos considerando os penúltimos valores de comprimentos de rota especificados (1458 para o ambiente de 2500 obstáculos e 1400 para o ambiente de 10000 obstáculos). Já para o último comprimento de rota especificado, que é igual à 1360 para o ambiente de 2500 obstáculos e igual à 1375 para o ambiente de 10000 obstáculos, os algoritmos com melhor tempo de planejamento foram o Informed-RRT* e o RRT*-Smart, respectivamente. Isso mostra que, para ambientes de navegação com grandes quantidades de obstáculos, o RRT*-SV não consegue convergir para as rotas com comprimentos mais reduzidos tão quanto as estratégias informadas desses algoritmos. Porém, para o ambiente com 2500 obstáculos, as rotas planejadas pelo Informed-RRT* convergiram para um menor comprimento mais rápido do que as rotas planejadas pelo algoritmo RRT*-Smart. O mesmo não ocorre para ambiente com 10000 obstáculos, onde o algoritmo RRT*-Smart planeja rotas de menor comprimento mais rápido do que o algoritmo Informed-RRT*.

Figura 4.31 - Segundo conjunto de testes computacionais: Tempo médio de planejamento (em segundos) considerando cinco simulações para os algoritmos planejarem rotas com comprimentos específicos no ambiente de navegação com cinco obstáculos.

Fonte: Produção do Autor.

Figura 4.32 - Segundo conjunto de testes computacionais: Tempo médio de planejamento (em segundos) considerando cinco simulações para os algoritmos planejarem rotas com comprimentos específicos no ambiente de navegação com 50 obstáculos.

Fonte: Produção do Autor.

Figura 4.33 - Segundo conjunto de testes computacionais: Tempo médio de planejamento (em segundos) considerando cinco simulações para os algoritmos planejarem rotas com comprimentos específicos no ambiente de navegação com 2500 obstáculos.

Fonte: Produção do Autor.

Figura 4.34 - Segundo conjunto de testes computacionais: Tempo médio de planejamento (em segundos) considerando cinco simulações para os algoritmos planejarem rotas com comprimentos específicos no ambiente de navegação com 10 000 obstáculos.

Fonte: Produção do Autor.

Portanto, para este segundo conjunto de experimentos, conclui-se que o algoritmo RRT*-SV tende a convergir mais rapidamente para uma rota de comprimento menor do que os algoritmos RRT*-Smart, Informed- RRT* e BIT*, apenas em ambientes de navegação com baixa quantidade de obstáculos. Em ambientes de navegação com muitos obstáculos (2500 e 10 000 obstáculos), o algoritmo RRT*-SV tende a apresentar tempo de execução maior para planejar uma primeira rota. É então sugerido usar cada método dependendo da aplicação. Em ambientes de navegação com baixa quantidade de obstáculos, se uma rota tiver que ser planejada no menor tempo possível e com menor comprimento, o RRT*-SV pode ser o mais adequado. Entretanto, se a necessidade é obter uma rota com comprimento minimizado em ambientes de navegação com número elevado de obstáculos, o algoritmo Informed-RRT* oferece vantagem sobre os demais algoritmos analisados neste conjunto de testes computacionais (no geral, pois no ambiente com 10 000 obstáculos o RRT*-Smart apresenta a maior vantagem), pois rotas de menor comprimento podem ser planejadas em menor tempo de planejamento utilizando o mesmo.

4.5.4 Terceiro conjunto de testes computacionais

Nesta seção, serão apresentados resultados de comparações estatísticas entre os algoritmos RRT*-Smart, Informed-RRT*, BIT* e RRT*-SV por meio do método estatístico t-Student (STUDENT, 1908). Foram executadas 100 simulações em ambientes de navegação com diferentes distribuições espaciais e formatos de obstáculos. Os ambientes de navegação usados, as coordenadas de $q_{início}$ e $q_{destino}$ adotadas e o tempo de planejamento utilizado em cada um são

a) **com cinco obstáculos:** $q_{início} = (25, 25), q_{destino} = (975, 975)$ - três segundos;

b) **com 50 obstáculos:** $q_{início} = (25, 25), q_{destino} = (975, 975)$ - três segundos;

c) **com 100 obstáculos:** $q_{início} = (25, 25), q_{destino} = (975, 975)$ - três segundos;

d) **com 200 obstáculos:** $q_{início} = (25, 25), q_{destino} = (975, 975)$ - seis segundos;

e) **obstáculos posicionados em zigue-zague:** $q_{início} = (500, 50), q_{destino} = (500, 950)$ - três segundos;

f) **obstáculo com formato em U:** $q_{início} = (500, 500), q_{destino} = (500, 100)$ - três segundos;

g) **labirinto:** $q_{início} = (25, 25), q_{destino} = (940, 940)$ - cinco segundos;

h) **um labirinto com obstáculo com formato em espiral:** $q_{início} = (500, 500), q_{destino} = (925, 925)$ - cinco segundos;

i) **obstáculos formando passagem estreita:** $q_{início} = (25, 25), q_{destino} = (925, 925)$ - dez segundos.

Para definir a quantidade de simulações, o intervalo de confiança (I.C.) para cada um dos testes computacionais com os ambientes de navegação foi calculado. O I.C. estima um intervalo contendo a média populacional hipotética com determinado nível de confiança a partir das amostras experimentais, assumindo que o processo de geração dessas amostras segue distribuição normal. A estimativa de I.C. foi calculado com nível de confiança igual a 95%, que na distribuição normal corresponde a $z = 1,96$. Quanto menor a amplitude do intervalo, maiores as chances da média real estar contida nesse intervalo para o nível de confiança especificado. O intervalo é afetado pela quantidade de amostras coletadas. Considerou-se que os intervalos calculados permitem inferir que o número de simulações realizadas pelo ambiente de navegação é adequado para a análise estatística. As variáveis estatísticas consideradas nesta análise são o comprimento das rotas planejadas e o tempo de planejamento das mesmas. Os valores de I.C. para cada ambiente de navegação estão descritos na Tabela 4.3 e na Tabela 4.4.

Tabela 4.3 - Terceiro conjunto de testes computacionais: Resultados e parâmetros do teste t-Student para comparação de valores relativos as rotas planejadas pelos algoritmos para diferentes tipos de ambientes de navegação considerando 100 simulações.

Algoritmo	Tempo de planejamento (s)	Comprimento médio	Desvio padrão	I.C.	t-value	z
		cinco obstáculos				
RRT*-SV	3	1413	0,451	1413±0,088	N/V	1,972
RRT*-Smart		1416	6,113	1416±1,198	3,804	
Informed-RRT*		1423	4,124	1423±0,808	22,252	
BIT*		1418	1,863	1418±0,365	24,781	
		50 obstáculos				
RRT*-SV	3	1395	5,667	1395±1,111	N/V	1,972
RRT*-Smart		1425	22,988	1425±4,506	12,786	
Informed-RRT*		1433	16,44	1433±3,222	22,229	
BIT*		1498	26,251	1498±5,145	38,636	
		100 obstáculos				
RRT*-SV	3	1377	3,909	1377±0,766	N/V	1,972
RRT*-Smart		1423	28,715	1423±5,628	15,827	
Informed-RRT*		1431	24,852	1431±4,871	21,28	
BIT*		1486	25,348	1486±4,968	42,275	
		200 obstáculos				

(Continua)

Tabela 4.3 - Continuação

Algoritmo	Tempo de planejamento (s)	Comprimento médio	Desvio padrão	I.C.	t-value	z
RRT*-SV	6	1399	6,095	1399±1,195	N/V	1,972
RRT*-Smart		1468	29,548	1468±5,791	22,818	
Informed-RRT*		1487	24,91	1487±4,882	34,625	
BIT*		1702	105,068	1702±20,593	28,787	
Formato em U						
RRT*-SV	3	1377	0,705	1377±0,138	N/V	1,972
RRT*-Smart		1384	10,913	1384±2,139	6,637	
Informed-RRT*		1437	16,6	1437±3,254	36,354	
BIT*		1434	16,093	1434±3,154	35,721	
Labirinto em espiral						
RRT*-SV	5	2302	0,313	2302±0,061	N/V	1,972
RRT*-Smart		2312	4,561	2312±0,894	22,165	
Informed-RRT*		2465	36,783	2465±7,209	44,253	
BIT*		2530	47,722	2530±9,353	47,791	
Zigue-zague						
RRT*-SV	3	1006	0	1006±0	N/V	1,972
RRT*-Smart		1080	69,991	1080±13,718	10,535	
Informed-RRT*		1055	23,169	1055±4,541	20,907	

(Continua)

Tabela 4.3 - Continuação

Algoritmo	Tempo de planejamento (s)	Comprimento médio	Desvio padrão	I.C.	t-value	z
BIT*		1103	27,258	1103±5,342	35,6	
		Labirinto				
RRT*-SV	5	1790	0,06	1790±0,012	N/V	1,972
RRT*-Smart		1799	4,249	1799±0,833	21,031	
Informed-RRT*		1885	42,118	1885±8,255	22,428	
BIT*		1990	56,814	1990±11,135	35,151	
		Com passagem estreita				
RRT*-SV	10	1581	0	1581±0	N/V	1,972
RRT*-Smart		1587	1,116	1587±0,219	54,021	
Informed-RRT*		1626	10,514	1626±2,061	42,201	
BIT*		1888	327,638	1888±64,216	9,371	

Fonte: Produção do Autor.

Tabela 4.4 - Terceiro conjunto de testes computacionais: Resultados e parâmetros do teste t-Student para comparação de valores relativos as primeiras rotas planejadas pelos algoritmos para diferentes tipos de ambientes de navegação considerando 100 simulações.

	Comprimento da rota					Tempo de planejamento (s)				
Algoritmo	Média	Desvio Padrão	I.C.	t-value	z	Média	Desvio Padrão	I.C.	t-value	z
cinco obstáculos										
RRT*-SV	1703	139,608	1703±27,363	N/V		0	0,001	0±0	N/V	
RRT*-Smart	1466	38,521	1466±7,55	-16,365	1,972	0,061	0,05	0,061±0,01	11,969	1,972
Informed-RRT*	1507	41,852	1507±8,203	-13,407		0,049	0,045	0,049±0,009	10,655	
BIT*	1491	35,428	1491±6,944	-14,714		0,009	0,006	0,009±0,001	13,451	
50 obstáculos										
RRT*-SV	1542	81,25	1542±15,925	N/V		0,023	0,023	0,023±0,005	N/V	
RRT*-Smart	1540	60,162	1540±11,792	-0,185	1,972	0,214	0,203	0,214±0,04	9,352	1,972
Informed-RRT*	1558	62,819	1558±12,312	1,532		0,181	0,186	0,181±0,036	8,435	
BIT*	1871	242,778	1871±47,584	12,864		0,059	0,052	0,059±0,01	6,296	
100 obstáculos										
RRT*-SV	1582	117,704	1582±23,07	N/V		0,086	0,069	0,086±0,014	N/V	
RRT*-Smart	1543	70,256	1543±13,77	-2,831	1,972	0,406	0,344	0,406±0,067	9,112	1,972
Informed-RRT*	1559	70,899	1559±13,896	-1,639		0,388	0,301	0,388±0,059	9,772	
BIT*	1875	207,672	1875±40,703	12,295		0,093	0,105	0,093±0,021	0,586	

(Continua)

Tabela 4.4 - Continuação

Algoritmo	Comprimento da rota					Tempo de planejamento (s)				
	Média	Desvio Padrão	I.C.	t-value	z	Média	Desvio Padrão	I.C.	t-value	z
200 obstáculos										
RRT*-SV	1512	45,991	1512±9,014	N/V	1,972	0,649	0,444	0,649±0,087	N/V	1,972
RRT*-Smart	1615	80,1	1615±15,699	11,123		1,09	0,747	1,09±0,146	5,069	
Informed-RRT*	1628	79,801	1628±15,641	12,569		1,232	0,887	1,232±0,174	5,867	
BIT*	1984	321,31	1984±62,976	14,551		0,776	0,76	0,776±0,149	1,434	
Formato em U										
RRT*-SV	1397	19,016	1397±3,727	N/V	1,972	0,001	0,001	0,001±0	N/V	1,972
RRT*-Smart	1510	44,623	1510±8,746	23,226		0,04	0,03	0,04±0,006	12,892	
Informed-RRT*	1608	61,595	1608±12,072	32,764		0,038	0,034	0,038±0,007	10,837	
BIT*	1593	60,166	1593±11,792	31,135		0,009	0,007	0,009±0,001	10,902	
Labirinto em espiral										
RRT*-SV	2309	21,218	2309±4,159	N/V	1,972	0,034	0,026	0,034±0,005	N/V	1,972
RRT*-Smart	2499	58,154	2499±11,398	30,629		0,385	0,276	0,385±0,054	12,659	
Informed-RRT*	2748	74,898	2748±14,68	56,425		0,389	0,284	0,389±0,056	12,443	
BIT*	2807	141,763	2807±27,785	34,737		0,036	0,029	0,036±0,006	0,39	
Zigue-zague										
RRT*-SV	1139	97,037	1139±19,019	N/V	1,972	0,001	0,001	0,001±0	N/V	1,972
RRT*-Smart	1265	59,078	1265±11,579	11,110		0,028	0,025	0,028±0,005	10,932	

(Continua)

Tabela 4.4 - Continuação

	Comprimento da rota					Tempo de planejamento (s)				
Algoritmo	Média	Desvio Padrão	I.C.	t-value	z	Média	Desvio Padrão	I.C.	t-value	z
Informed-RRT*	1317	73,377	1317±14,382	14,638		0,036	0,03	0,036±0,006	11,859	
BIT*	1291	47,843	1291±9,377	13,996		0,013	0,006	0,013±0,001	18,785	
Labirinto										
RRT*-SV	1796	8,989	1796±1,762	N/V		0,047	0,027	0,047±0,005	N/V	
RRT*-Smart	1884	33,438	1884±6,554	25,364	1,972	0,673	0,607	0,673±0,119	10,31	1,972
Informed-RRT*	1973	61,439	1973±12,042	28,507		0,69	0,66	0,69±0,129	9,73	
BIT*	2124	115,096	2124±22,558	28,41		0,199	0,235	0,199±0,046	6,425	
Com passagem estreita										
RRT*-SV	1599	0	1599±0	N/V		0,004	0,004	0,004±0,001	N/V	
RRT*-Smart	1689	22,591	1689±4,428	39,562	1,972	0,354	0,418	0,354±0,082	8,361	1,972
Informed-RRT*	1775	48,187	1775±9,445	36,594		0,356	0,439	0,356±0,086	8,01	
BIT*	1897	323,745	1897±63,453	9,189		5,383	3,336	5,383±0,654	16,12	

Fonte: Produção do Autor.

Neste terceiro conjunto de testes computacionais, as variáveis estatísticas foram modeladas como amostras aleatórias, sem autocorrelação entre os resultados de cada simulação para cada método e sem dependência entre os resultados dos algoritmos RRT*-Smart, Informed-RRT*, BIT* e RRT*-SV. A comparação foi realizada com pares de conjuntos de amostras, onde um dos conjuntos do par sempre é relativo ao algoritmo RRT*-SV e o outro representa resultados de simulações de um dos algoritmos RRT*-Smart, Informed-RRT* ou BIT*. Para compará-los, aplicou-se o teste t de Student para verificar a hipótese nula, ou seja, para verificar se não há diferença entre os dois conjuntos de amostras (resultados das simulações). Se a hipótese nula for considerada verdadeira, não haverá diferença significativa entre as médias amostrais. Caso contrário, o teste identifica a diferença e mostra o quanto os dois diferem significativamente. O teste é adequado quando as médias e variâncias populacionais relacionadas aos conjuntos amostrais não são conhecidas (BOX et al., 1978). A versão do método de t-Student utilizada foi o teste t de variâncias desiguais de Welch (1947), aplicado quando as variâncias de dois conjuntos de amostras são diferentes. O valor de α, que define o nível de significância do teste, foi definido igual a $0,05$.

Com base nos resultados apresentados na Tabela 4.3 para as estatísticas sobre o comprimento das rotas planejadas no final de cada simulação, nos experimentos com o RRT*-SV o comprimento médio das rotas foi menor para todos os ambientes de navegação considerados. Além disso, os desvios padrão das médias dos resultados do RRT*-SV são menores quando comparados aos desvios padrão obtidos pelos demais algoritmos, ou seja, o valor do comprimento das rotas é menor e varia menos quando planejadas pelo algoritmo RRT*-SV considerando o tempo de planejamento aplicado em cada ambiente de navegação. Os valores do teste t-Student foram maiores que os valores de z relativos ao teste (última coluna da Tabela 4.3), que pelos critérios do teste indicam que a hipótese nula é falsa, e que os métodos apresentam diferença estatística entre seus resultados. Exemplo de rotas planejadas em cada um dos ambientes de navegação considerados encontram-se ilustrados nas Figuras 4.35, 4.36 e 4.37.

Figura 4.35 - Terceiro conjunto de testes computacionais: Primeiro conjunto de ambientes de navegação usados nas comparações estatísticas entre o algoritmo RRT*-SV e os algoritmos RRT*-Smart, Informed-RRT* e BIT*

Listagem dos ambientes de navegação e rotas referentes a cada algoritmo: cinco obstáculos - a) RRT*-Smart, b) Informed-RRT*, c) BIT* e d) RRT*-SV; 50 obstáculos - e) RRT*-Smart, f) Informed-RRT*, g) BIT* e h) RRT*-SV; 100 obstáculos - i) RRT*-Smart, j) Informed-RRT*, k) BIT* e l) RRT*-SV.

Fonte: Produção do Autor.

Figura 4.36 - Terceiro conjunto de testes computacionais: Segundo conjunto de ambientes de navegação usados nas comparações estatísticas entre o algoritmo RRT*-SV e os algoritmos RRT*-Smart, Informed-RRT* e BIT*.

Listagem dos ambientes e rotas referentes a cada algoritmo: 200 obstáculos - a) RRT*-Smart, b) Informed-RRT*, c) BIT* e d) RRT*-SV; zigue-zague - e) RRT*-Smart, f) Informed-RRT*, g) BIT* e h) RRT*-SV; formato em U - i) RRT*-Smart, j) Informed-RRT*, k) BIT* e l) RRT*-SV.

Fonte: Produção do Autor.

Figura 4.37 - Terceiro conjunto de testes computacionais: Terceiro conjunto de ambientes de navegação usados nas comparações estatísticas entre o algoritmo RRT*-SV e os algoritmos RRT*-Smart, Informed-RRT* e BIT*.

Listagem dos ambientes e rotas referentes a cada algoritmo: labirinto - a) RRT*-Smart, b) Informed-RRT*, c) BIT* e d) RRT*-SV; labirinto em espiral - e) RRT*-Smart, f) Informed-RRT*, g) BIT* e h) RRT*-SV; com passagem estreita - i) RRT*-Smart, j) Informed-RRT*, k) BIT* e l) RRT*-SV.

Fonte: Produção do Autor.

A ideia de usar os vértices convexos no processo de amostragem da RRT* é acelerar a convergência do algoritmo para uma solução. Para auxiliar na hipótese de que isso ocorrerá, o teste t de Student também foi aplicado aos dados de simulação relacionados a primeira rota planejada por cada algoritmo.

As estatísticas relacionadas às primeiras rotas planejadas nas simulações estão descritas na Tabela 4.4. No ambiente de navegação com cinco obstáculos, os algoritmos RRT*-Smart, Informed-RRT* e BIT* apresentam um desempenho melhor do que o RRT*-SV em planejar primeiras rotas mais curtas. Entretanto, isso ocorre devido a disposição dos vértices convexos e o comportamento do RRT*-SV ao expandir a árvore pelos mesmos. Como o RRT*-SV sempre seleciona o vértice convexo não conectado mais próximo, a disposição desses vértices neste ambiente de navegação direciona a expansão da árvore para a obtenção de rotas de maior comprimento. Por outro lado, o tempo de planejamento da primeira rota pelo RRT*-SV é menor que os dos demais algoritmos.

Ainda considerando os dados descritos na Tabela 4.4, em dois ambientes de navegação (com 50 obstáculos e 100 obstáculos), os algoritmos RRT*-Smart e Informed-RRT* planejaram rotas de comprimentos estatisticamente iguais com o RRT*-SV. Nos demais ambientes de navegação, o RRT*-SV demonstra melhor desempenho em planejar primeiras rotas mais curtas do que os demais algoritmos. Considerando o tempo de planejamento dessas primeiras rotas, o RRT*-SV possui melhor desempenho do que todos os outros algoritmos em todos os ambientes de navegação (os valores t do teste de t de Student são positivos e possuem módulos maiores do que z). Há exceções, como no caso do algoritmo BIT*, que nos ambientes de navegação com 100 obstáculos, 200 obstáculos e labirinto em espiral não apresenta diferença estatística de tempo de planejamento do algoritmo RRT*-SV. Porém, como o comprimento da primeira rota planejada é menor para o RRT*-SV nesses ambientes de navegação, esse apresenta mais vantagens do que o algoritmo BIT*.

Também são apresentados gráficos com a taxa de sucesso de cada algoritmo em planejar rotas com frações da rota ótima (de menor comprimento). Os gráficos foram gerados utilizando a mesma estratégia apresentada na Seção 4.3. Os gráficos estão ilustrados na Figura 4.38. Nestes gráficos, quanto mais uma linha se manter acima das outras, melhor o seu desempenho. Isso é o que ocorre com a linha do gráfico que representa o algoritmo RRT*-SV (linha na cor verde). Ela sempre se mantém acima do gráfico dos demais algoritmos. Todos os gráficos decaem a medida que o valor da fração se aproxima de 1. Nota-se que o RRT*-SV planeja rotas com o comprimento

ótimo com taxa de sucesso igual a 100% em três ambientes de navegação: labirinto com formato em espiral (Figura 4.38(f)); labirinto (Figura 4.38(h)); e passagem estreita (Figura 4.38(i)).

Figura 4.38 - Terceiro conjunto de teste computacionais: Taxa de sucesso de cada algoritmo em alcançar frações da rota ótima em 100 simulações com sementes distintas para geração de número pseudoaleatório.

Os gráficos são referentes aos seguintes ambientes de navegação: a) com cinco obstáculos; b) com 50 obstáculos; c) com 100 obstáculos; d) com 200 obstáculos; e) com formato em U; f) com formato em espiral; g) com obstáculos em zigue-zague; h) com formato em labirinto; i) e com passagem estreita.

Fonte: Produção do Autor.

Outros gráficos apresentados representam a taxa de sucesso de cada algoritmo em planejar rotas por tempo de planejamento. Esses gráficos estão ilustrados na Figura 4.39. Nestes gráficos, quanto mais acima estiver uma linha sobre a outra, melhor o desempenho do algoritmo que a mesma representa. A linha referente ao algoritmo RRT*-SV (novamente representado pela cor verde), mantém-se acima das demais em todos os gráficos, com exceção de três deles, referentes aos seguintes ambientes de navegação: com 100 obstáculos (Figura 4.39(c)); com 200 obstáculos (Figura 4.39(d)); e labirinto com formato em espiral (Figura 4.39(f)). Esses resultados demonstram a dependência do RRT*-SV da quantidade de vértices convexos no ambiente de navegação pois, naqueles com maior quantidade de obstáculos, é onde o algoritmo RRT*-SV apresenta tempo de planejamento similar ou inferior comparado aos demais algoritmos.

Figura 4.39 - Terceiro conjunto de teste computacionais: Taxa de sucesso de cada algoritmo em planejar uma rota por tempo de planejamento em segundos (log_{10}) em 100 simulações com sementes distintas para geração de número pseudoaleatório.

Os gráficos são referentes aos seguintes ambientes de navegação e tempos de planejamento: a) com cinco obstáculos (três segundos); b) com 50 obstáculos (três segundos); c) com 100 obstáculos (três segundos); d) com 200 obstáculos (seis segundos); e) com formato em U (três segundos); f) com formato em espiral (cinco segundos); g) com obstáculos em zigue-zague (três segundos); h) com formato em labirinto (cinco segundos); i) e com passagem estreita (10 segundos).

Fonte: Produção do Autor.

Considerando todos os resultados apresentados neste terceiro conjunto de testes computacionais, conclui-se que, em geral, o uso do algoritmo RRT*-SV para resolver o problema de planejamento de rota para os ambientes de navegação considerados fornece mais vantagens em relação aos algoritmos comparados RRT*-Smart, Informed-RRT* e BIT*.

4.5.5 Aspectos técnicos dos experimentos

Os algoritmos RRT*, RRT*-Smart, Informed-RRT*, BIT* e RRT*-SV foram implementados na linguagem de programação C++ e compilados com o *Microsoft Visual Studio Compiler* versão 14. O processo de amostragem baseado em região elipsoidal dos algoritmos Informed-RRT* e BIT* foi desenvolvido com base no código disponível da biblioteca *Open Motion Planning Library (OMPL)* (SUCAN et al., 2012). Como nessa biblioteca, as operações de álgebra linear são solucionadas por funcionalidades da biblioteca *Eigen* (GUENNEBAUD et al., 2014). A biblioteca de geometria computacional CGAL (BRÖNNIMANN et al., 2009) foi utilizada para auxiliar na representação geométrica dos ambientes de navegação e na verificação de colisão com os obstáculos. A implementação do algoritmo *k-d-tree* da biblioteca ANN (MOUNT, 2010) foi usada na busca de um vértice convexo mais próximo. Para a representação gráfica dos experimentos, foi utilizada a biblioteca OpenGL. Os experimentos foram executados em um computador com sistema operacional Windows 10 de 64 bits, 16 GB de memória e processador Intel i7-2820QM com *clock* de 2.30Ghz.

4.6 Considerações

O algoritmo proposto neste trabalho não sofre influência da orientação de Voronoi, pois utiliza os vértices convexos do ambiente de navegação e a grade de Sukharev para rapidamente criar arestas para fora destas regiões. Esse fator é importante para que a expansão da árvore não se encontre em um problema de mínimo local. Além disso, o tamanho do passo de crescimento no RRT*-SV será definido pelas distâncias entre os nós da árvore e os vértices convexos das envoltórias de segurança dos obstáculos, diferente do algoritmo RRT, que utiliza o parâmetro Δq para definir o tamanho de uma nova aresta de sua árvore a cada iteração. Neste trabalho, o valor de Δq é definido por uma porcentagem do maior lado do ambiente de navegação, porém, não faz parte do seu escopo o estudo da estimativa do valor de Δq mais adequado para cada ambiente de navegação. Os algoritmos RRT*-Smart e Informed-RRT* sofrem forte influência desse parâmetro. Isso significa que o algoritmo RRT*-SV é mais robusto neste aspecto.

Com base nos testes computacionais apresentados nas Seções 4.5.2, 4.5.3 e 4.5.4, pode-se concluir que quanto menor a quantidade de vértices convexos no ambiente de navegação, mais adequado é o uso do algoritmo RRT*-SV como meio de solucionar problemas de planejamento de rota em ambientes de navegação com obstáculos estáticos. O desempenho do algoritmo é altamente penalizado pela quantidade de vértices convexos do ambiente de navegação, porém, possui alta eficiência no planejamento de uma primeira rota (sem considerar que a mesma tenha menor comprimento do que as rotas planejadas pelos outros algoritmos) e converge para rotas de menor comprimento mais rapidamente que os algoritmos RRT*, RRT*-Smart, Informed-RRT* e BIT*. Entretanto, em ambientes de navegação com um grande número de vértices convexos, a convergência para rotas mais curtas pode ser mais lenta pelo algoritmo RRT*-SV do que por alguns dos outros algoritmos, como apresentado na Seção 4.5.3.

5 PROCESSO DE SUAVIZAÇÃO DE ROTAS

Qualquer VANT possui limitada capacidade de realização de manobras. No problema de planejamento de rotas, a restrição proveniente da capacidade cinemática e dinâmica do VANT deve ser considerada na solução do mesmo. Pode-se alcançar uma solução para o problema de planejamento de rota e torná-la viável para a navegação do VANT em duas etapas: a) encontrando uma rota livre de colisão para um dado ambiente de navegação; e b) suavizando as partes da rota que o VANT não conseguiria realizar devido às suas limitações de manobras. Calculando-se o comprimento da rota suavizada e conhecendo-se o módulo da velocidade de navegação, é possível estimar a posição e a atitude (ângulos de arfagem, de rolamento e de guinada/proa) que o VANT deve descrever nessa rota, a cada instante de tempo.

Rotas não adequadas para a realização de trajetórias pelo VANT são prováveis de serem planejadas por algoritmos baseados em amostragem. Nesses algoritmos, os pontos amostrados são conectados por segmentos de retas que apresentam em suas junções, cantos com formato pontiagudo (*sharper*) que não são adequadas, em geral, para a trajetória de VANTs. Para que fosse possível o VANT realizar a trajetória por essas formas geométricas, ele precisaria desacelerar até que a sua velocidade se aproximasse de zero, para então rotacionar em torno de seu eixo até que estivesse alinhado com o segmento seguinte para continuar a navegação.

Para suavizar os cantos pontiagudos das rotas planejadas por um algoritmo como o RRT*-SV, é necessário utilizar estruturas como as curvas de Bézier e *splines* (REINSCH, 1967). Entretanto, essas estruturas necessitam de métodos de resolução de integrais para obter delas o seu comprimento, o que acarreta custo computacional adicional ao planejamento. Em Farouki e Sakkalis (1990), uma curva denominada de curva de Hodógrafo de Pitágoras (também conhecida como curva HP) é introduzida, onde os componentes do hodógrafo de uma curva paramétrica $\boldsymbol{r}(\xi)$ são definidos por polinômios que combinados satisfazem ao Teorema de Pitágoras. A sua vantagem, dentre outras que podem ser verificadas em Farouki (2008), é que a estrutura obtida permite a resolução do problema do cálculo do comprimento da curva de forma analítica e exata, não sendo necessário o uso de técnicas de resolução de integrais. Essa característica das curvas HP é bastante útil no problema de planejamento que busque a rota de menor comprimento, já que o comprimento da curva pode ser obtido rapidamente por uma equação. Além disso, as curvas podem ser adaptadas para a suavização dos cantos pontiagudos apresentados por rotas planejadas pelos métodos baseados em amostragem, como mostrado em Farouki et al. (2018).

Neste capítulo, é apresentado um esquema de suavização das rotas planejadas pelo algoritmo RRT*-SV, baseado na solução do problema de suavização de cantos pontiagudos proposta em Farouki (2008) e na sua aplicação em Véras et al. (2018a). Para tal, inicialmente é necessário definir a distância em torno dos limites de um obstáculo para a geração de sua envoltória de segurança. Essa distância deve ser calculada de forma que haja garantia de que a curva gerada na suavização dos cantos de uma rota não intercepte os limites do obstáculo para o qual a envoltória foi gerada. Caso contrário, é possível que devido aos nós de uma rota planejada pelo algoritmo RRT*-SV possuírem as mesmas coordenadas dos vértices convexos dessas envoltórias a mesma intercepte um dos limites dos obstáculos ao ser suavizada. A suavização reduz o comprimento de uma porção da rota, onde o seu traço é deslocado na direção da abertura do ângulo interno associado ao vértice convexo utilizado para gerar o nó correspondente na rota. A Figura 5.1 ilustra uma situação onde a distância para a geração da envoltória de segurança não permite gerar uma curva livre de colisão com os limites de seu obstáculo.

Figura 5.1 - Curva gerada em situação de colisão devido à uma envoltória de segurança (traço em verde) posicionada à uma distância curta de seu obstáculo.

Os pontos em vermelho representam a posição na rota necessária para a geração de curva com valor de curvatura que atenda às restrições cinemáticas de um VANT.

Fonte: Produção do Autor.

A partir das estruturas definidas para a construção de curvas HP é possível obter as distâncias adequadas para a criação das envoltórias dada a curvatura do VANT. Para garantir a continuidade da curva com os segmentos que formam os cantos, em Farouki et al. (2018) é proposta uma formulação de curvas HP com continuidade G^2 (continuidade geométrica de grau dois - mudança de curvatura constante) com os segmentos do canto suavizado. Na próxima seção, as formulações da curva HP utilizadas são apresentadas. Na Seção 5.1, as formulações da curva HP para o problema de suavização de cantos pontiagudos são descritas. Na última seção deste capítulo, o esquema de definição da distância para a geração das envoltórias de segurança é apresentada e exemplos de suavização de rotas planejadas pelo algoritmo RRT*-SV são discutidos.

5.1 Hodógrafo de Pitágoras

Proposta em Farouki e Sakkalis (1990), Hodógrafo de Pitágoras é um tipo de curva onde os componentes do seu hodógrafo possuem uma relação com um polinômio de forma que atendam ao Teorema de Pitágoras, ou seja, a soma dos quadrados dos componentes do hodógrafo é igual ao quadrado desse polinômio. Um impacto direto desta relação é que a integral do cálculo do comprimento da curva poder ser obtida em uma forma fechada, permitindo a sua resolução algébrica, evitando assim a necessidade de técnicas de quadraturas para a resolução de integrais. Essa caraterística é utilizada para obter vantagem computacional neste trabalho. As formulações que definem o Hodógrafo de Pitágoras serão definidas a seguir. Demonstrações e outras propriedades podem ser verificadas em Farouki (2008).

Seja uma curva polinomial definida por $\boldsymbol{r}(\xi) = (x(\xi), y(\xi))$ e $\xi \in [0, 1]$ o parâmetro que define cada posição na curva. O comprimento do seu arco é dado por:

$$S = \int_0^1 ||\boldsymbol{r}'(\xi)|| \tag{5.1}$$

onde S é o comprimento total da curva e $\boldsymbol{r}'(\xi)$ é o vetor velocidade da curva (primeira derivada). A Equação 5.1 pode ser reescrita como:

$$S = \int_0^1 \sqrt{x'(\xi)^2 + y'(\xi)^2} \tag{5.2}$$

onde $x'(\xi) = dx/d\xi$ e $y'(\xi) = dy/d\xi$. Relacionando o termo interno da raiz a um polinômio $\sigma(\xi)$, a formulação do comprimento da curva pode ser dada por:

$$S = \int_0^1 \sqrt{\sigma(\xi)^2} = \int_0^1 |\sigma(\xi)| \tag{5.3}$$

Percebe-se então que $\sigma(\xi)$ passa a representar a velocidade paramétrica da curva. É necessário que se defina $\sigma(\xi)$ em termos de polinômios que atendam a Equação (5.3). Dado os polinômios $u(\xi)$, $v(\xi)$, as seguintes definições de $x'(\xi)$ e $y'(\xi)$ satisfazem ao Teorema de Pitágoras na definição de $\sigma(\xi)$:

$$x'(\xi) = u(\xi)^2 - v(\xi)^2, \quad y'(\xi) = 2u(\xi)v(\xi) \tag{5.4}$$

portanto, $\sigma(\xi)$ pode ser dado em termo de $u(\xi)$, $v(\xi)$ por:

$$S = \int_0^1 |\sigma(\xi)| = \int_0^1 u(\xi)^2 + v(\xi)^2, \tag{5.5}$$

Incorporando essas definições em uma curva quíntica (representada por um polinômio de grau cinco) na forma de Bézier, definida por

$$\boldsymbol{r}(\xi) = \sum_{k=0}^{5} \boldsymbol{p_k} \binom{5}{k} (1-\xi)^{5-k} \xi^k, \tag{5.6}$$

os pontos de controle $\boldsymbol{p_k} = (x_k, y_k)$, dados pelos coeficientes de Bernstein, são definidos em termos dos polinômios $u(\xi)$, $v(\xi)$ por:

$$\begin{aligned}
\boldsymbol{p_1} &= \boldsymbol{p_0} + \tfrac{1}{5}(u_0^2 - u_0^2, 2u_0u_0) \\
\boldsymbol{p_2} &= \boldsymbol{p_1} + \tfrac{1}{5}(u_0u_1 - v_0v_1, u_0v_1 + u_1v_0) \\
\boldsymbol{p_3} &= \boldsymbol{p_2} + \tfrac{2}{15}(u_1^2 - u_1^2, 2u_1u_1) + \tfrac{1}{15}(u_0u_2 - v_0v_2, u_0u_2 + u_2v_0) \\
\boldsymbol{p_4} &= \boldsymbol{p_3} + \tfrac{1}{5}(u_1u_2 - v_1v_2, u_1u_2 + u_2v_1) \\
\boldsymbol{p_5} &= \boldsymbol{p_4} + \tfrac{1}{5}(u_2^2 - v_2^2, 2u_2v_2)
\end{aligned} \tag{5.7}$$

onde $\boldsymbol{p_0}$ é arbitrariamente especificado. O polinômio que define a velocidade paramétrica $\sigma(\xi)$ dado por:

$$\sigma(\xi) = \sum_{k=0}^{4} \sigma_k \binom{4}{k} (1-\xi)^{4-k} \xi^k, \tag{5.8}$$

possui seus coeficientes de Bernstein definidos em termos de $u(\xi)$, $v(\xi)$ iguais à:

$$\begin{aligned}
\sigma_0 &= u_0^2 + u_0^2 \\
\sigma_1 &= v_0 u_1 + v_0 v_1 \\
\sigma_2 &= \tfrac{2}{3}(u_1^2 + v_1^2) + \tfrac{1}{3}(u_0 u_2 + v_0 v_2) \\
\sigma_3 &= u_1 u_2 + v_1 v_2 \\
\sigma_4 &= v_2^2 + u_2^2
\end{aligned} \tag{5.9}$$

O comprimento do arco, dado em função dos coeficientes definidos na Equação (5.9), é:

$$S = \frac{\sigma_0 + \sigma_1 + \sigma_2 + \sigma_3 + \sigma_4}{5} \tag{5.10}$$

e o valor de curvatura da curva é dado por:

$$\kappa(\xi) = \frac{2[u(\xi)v'(\xi) - u'(\xi)v(\xi)]}{u(\xi)^2 + v(\xi)^2} \tag{5.11}$$

5.1.1 Representação complexa de curva HP quíntica

A definição da curva HP descrita na Seção 5.1 considera a sua representação no plano cartesiano. Porém, a biblioteca *PHquintic*, introduzida em Dong e Farouki (2015), utilizada no desenvolvimento deste trabalho, utiliza a representação da curva no plano complexo, onde as coordenadas da curva são definidas por um eixo real e imaginário. A vantagem de se definir a curva no eixo imaginário é que há uma significante simplificação nas expressões associadas à curva HP (FAROUKI, 2008). Curvas HP com representação complexa foram introduzidas em Farouki (1994). Por brevidade, esta seção irá apresentar somente as formulações dos pontos de controle, dos coeficientes da velocidade paramétrica e dos coeficientes do comprimento do arco e demais propriedades da curva associadas às variáveis complexas. As relações necessárias para se obter a curva HP no plano complexo podem ser encontradas em Farouki (1994) e Farouki (2008).

Para definir uma curva paramétrica no plano complexo, os polinômios associados ao hodógrafo em (5.4) devem ser definidos na forma complexa tal que:

$$\boldsymbol{w}(\xi) = u(\xi) + iv(\xi) = \sum_{k=0}^{m} \boldsymbol{w_k} \binom{m}{k} (1-\xi)^{4-k} \xi^k; \tag{5.12}$$

com coeficientes de Bernstein $\boldsymbol{w_k} = u_k + iv_k$. Podemos associar então $w(\xi)$ à velocidade paramétrica, obtendo-se então

$$r'(\xi) = \boldsymbol{w}(\xi)^2 \tag{5.13}$$

Os pontos de controles são definidos em termos de $\boldsymbol{w}(\xi)$ como:

$$\begin{aligned}
\boldsymbol{p_1} &= \boldsymbol{p_0} + \tfrac{1}{5}(\boldsymbol{w}_0^2) \\
\boldsymbol{p_2} &= \boldsymbol{p_1} + \tfrac{1}{5}(\boldsymbol{w}_0 \boldsymbol{w}_1) \\
\boldsymbol{p_3} &= \boldsymbol{p_2} + \tfrac{1}{5}(\tfrac{2\boldsymbol{w}_1^2 + \boldsymbol{w}_0 \boldsymbol{w}_2}{3}) \\
\boldsymbol{p_4} &= \boldsymbol{p_3} + \tfrac{1}{5}(\boldsymbol{w}_1 \boldsymbol{w}_2) \\
\boldsymbol{p_5} &= \boldsymbol{p_4} + \tfrac{1}{5}(\boldsymbol{w}_2^2)
\end{aligned} \tag{5.14}$$

Os coeficientes de Bernstein da velocidade paramétrica $\sigma(\xi)$ em (5.8) são definidos como:

$$\begin{aligned}
\sigma_0 &= |\boldsymbol{w}_0|^2 \\
\sigma_1 &= Re(\bar{\boldsymbol{w}}_0 w_1) \\
\sigma_2 &= \tfrac{2|\boldsymbol{w}_1|^2 + Re(\bar{\boldsymbol{w}}_0 \boldsymbol{w}_2)}{3} \\
\sigma_3 &= Re(\bar{\boldsymbol{w}}_1 \boldsymbol{w}_2) \\
\sigma_4 &= |\boldsymbol{w}_2|^2
\end{aligned} \tag{5.15}$$

Em termos de $\boldsymbol{w}(\xi)$, a velocidade paramétrica e a curvatura podem ser formuladas como:

$$\sigma(\xi) = |\boldsymbol{w}(\xi)|^2, \kappa(\xi) = 2\frac{Im(\bar{\boldsymbol{w}}(\xi)\boldsymbol{w'}(\xi))}{\sigma(\xi)^2} \tag{5.16}$$

O comprimento total do arco é dado pela Equação 5.10.

5.2 Problema de suavização de cantos com curvas HP quínticas com continuidade G^2

No problema de arredondamento (suavização) de cantos, a junção entre dois segmentos deve ser transformada em uma curva de tal forma que haja continuidade entre os segmentos e os pontos delimitantes da curva, ou seja, $r(0)$ e $r(1)$ devem pertencer a cada um dos segmentos que formam o canto, respectivamente. Em Farouki et al. (2018) foi proposta uma solução com o uso de curvas HP quínticas para este problema. A curva HP é modelada de tal forma que atenda uma curva de continuidade G^2 - onde posição, tangentes e curvaturas devem ser iguais para ambos os segmentos se encontram após a formação da curva. Nesta seção estão descritas as formulações para a curva HP apresentadas em Farouki et al. (2018) para o problema.

Sejam três pontos representados por p_i, p_c e p_o, onde $\overline{p_i p_c}$ e $\overline{p_c p_o}$ são segmentos distintos interligados por p_c. Definindo a forma canônica de representação desses pontos como:

$$p_i = (0,0), \quad p_c = (L,0), \quad p_0 = ((1 + cos(\theta)L, sin(\theta)L) \tag{5.17}$$

onde $L = |p_c - p_i| = |p_o - p_c|$ e θ é o ângulo de deslocamento do segmento $\overline{p_c p_o}$ em relação à $\overline{p_i p_c}$ (ver Figura 5.2).

Figura 5.2 - Forma canônica do problema de suavização de cantos pontiagudos e exemplos de suavização para $\theta = \pi/4$, $\theta = \pi/2$ e $\theta = 3\pi/4$.

Fonte: Farouki et al. (2018).

A solução para a forma canônica, como definida em Farouki et al. (2018), é dada em termos de L e θ por:

$$\boldsymbol{w}_0 = \lambda\sqrt{L}, \quad \boldsymbol{w}_1 = 0, \quad \boldsymbol{w}_2 = \lambda\sqrt{L}exp(i\frac{1}{2}\theta) \tag{5.18}$$

em que:

$$\lambda = \sqrt{(30cos(\frac{1}{2}\theta))/(6cos(\frac{1}{2}\theta)+1)} \tag{5.19}$$

A velocidade paramétrica na Equação 5.8 passa a ter os coeficientes de Bernstein definidos como:

$$\begin{aligned} \sigma_0 &= \lambda^2 L \\ \sigma_1 &= 0 \\ \sigma_2 &= \tfrac{\lambda^2 L cos(\frac{1}{2}\theta)}{3} \\ \sigma_3 &= 0 \\ \sigma_4 &= \lambda^2 L \end{aligned} \tag{5.20}$$

A curvatura em qualquer ponto ξ da curva é dada por:

$$\kappa(\xi) = 4\lambda^2 L sin(\frac{1}{2}\theta)(\frac{(1-\xi)\xi}{\sigma(\xi)^2}) \tag{5.21}$$

A maior curvatura da curva κ_e, dada em $\xi = 1/2$, é dada por:

$$\kappa_e = \frac{32(6cos(\frac{1}{2}\theta)+1)tan(\frac{1}{2}\theta)}{15L(cos(\frac{1}{2}\theta)+1)^2} \tag{5.22}$$

O comprimento total do arco é dado por:

$$S = \frac{2L(6+cos(\frac{1}{2}\theta))cos(\frac{1}{2}\theta)}{6cos(\frac{1}{2}\theta)+1} \tag{5.23}$$

A curva gerada por essas formulações irão gerar um desvio no ponto médio da curva $r(1/2)$ do ponto p_c igual à:

$$\delta = \frac{(3cos(\frac{1}{2}\theta)+8)|sin(\frac{1}{2}\theta)|L}{8(6cos(\frac{1}{2}\theta)+1)} \tag{5.24}$$

As formulações para a curva HP em função de L e θ serão utilizadas na solução apresentada na Seção 5.3.

5.3 Cálculo da distância de geração de envoltória de segurança para valor de curvatura do VANT

Como mencionado anteriormente, no problema de planejamento de trajetória aplicado a cenários reais, deve-se considerar a utilização de uma envoltória de segurança em torno dos obstáculos, caso sejam planejados trajetos muito próximos aos limites dos obstáculos, evitando assim o risco de colisão do VANT. No entanto, a rota planejada pelo RRT*-SV encontra-se próxima dos obstáculos devido ao uso dos vértices convexos das envoltórias de segurança desses obstáculos. Assim, é importante definir essas envoltórias considerando uma distância segura dos obstáculos, garantindo que a curva que suaviza a rota nas junções entre os seus segmentos de reta não intercepte os limites desses obstáculos. Nesta seção, é apresentada uma estratégia, a qual se encontra publicada em Véras et al. (2018a), para definir essa distância a partir de uma curvatura máxima que o VANT teoricamente pode realizar. Isso é importante para garantir que todos os trechos da rota sejam livres de colisão com os obstáculos do ambiente de navegação. A estratégia utilizada define uma única distância para todas as envoltórias convexas de seus respectivos obstáculos, de forma que qualquer curva suavizada próxima aos vértices convexos esteja livre de colisão.

Considerando que uma envoltória de segurança é gerada a uma distância d de um obstáculo, é possível definir seus vértices convexos a partir dos vértices convexos de seu obstáculo. Um vértice convexo de uma envoltória de segurança estará a uma distância $d\ sec(\frac{1}{2}\theta)$ do vértice convexo de seu obstáculo na direção da geração da envoltória (de acordo com o que está ilustrado na Figura 5.3), onde θ é o ângulo de abertura no sentido anti-horário entre os segmentos que definem o vértice convexo do obstáculo (como ilustrado no caso canônico na Figura 5.2) (FAROUKI et al., 2018). Assim, o seguinte relacionamento pode ser definido para evitar que uma curva intercepte qualquer obstáculo ao ser gerada (FAROUKI et al., 2018):

$$d\ sec(\frac{1}{2}\theta) > \delta. \tag{5.25}$$

Figura 5.3 - Cálculo da posição de vértice convexo (em azul) da envoltória de segurança em relação ao vértice convexo (em verde) do obstáculo (representado pela região cinza).

Fonte: Produção do Autor.

Seja o valor de curvatura de um VANT dado por κ_{VANT}, a distância L_{VANT} de p_c, que gera uma curva $\boldsymbol{r}(\xi)$ com o valor de curvatura correspondente à da posição $\xi = 1/2$ na curva (valor de curvatura máxima associada à curva), pode ser obtida a partir da Equação 5.22, reformulando-a como:

$$L_{VANT} = \frac{32(6cos(\frac{1}{2}\theta) + 1)tan(\frac{1}{2}\theta)}{15\kappa_{VANT}(cos(\frac{1}{2}\theta) + 1)^2} \tag{5.26}$$

Ao associar L_{VANT} com 5.25, a distância dos obstáculos ao VANT com curvatura κ_{VANT} pode ser obtida por:

$$d > \frac{(3cos(\frac{1}{2}\theta) + 8)|sin(\frac{1}{2}\theta)|L_{VANT}}{8(6 + sec(\frac{1}{2}\theta))} \tag{5.27}$$

Seja d_i a distância resultante de 5.27 para o i-ésimo vértice de um obstáculo do ambiente de navegação com a curvatura κ_{VANT}, o valor d_{Obs}, que define a distância entre a borda de cada envoltória de segurança e seu respectivo obstáculo, é dado por:

$$d_{Obs} = max(d_i) \tag{5.28}$$

Como a distância d_{Obs} deve ser maior que o valor de d na formulação 5.27 para garantir a não-colisão do VANT durante a navegação pela curva, a largura d_{VANT} do VANT pode ser adicionada a ele. Desta forma, é garantido que o VANT de curvatura κ_{VANT} será capaz de realizar sua navegação por curvas construídas próximas a qualquer vértice convexo do ambiente de navegação. Uma vantagem direta desse método é que, como a distância é definida algebricamente a partir da restrição 5.25, não é necessário executar uma verificação de colisão para a curva obtida.

5.3.1 Teste computacional e resultados

Simulações foram realizadas para dois valores de curvatura de VANT, iguais a $0,009$ e $0,015$. Também é utilizado um ambiente de navegação com cinco obstáculos estáticos com diferentes formas geométricas. Este ambiente de navegação possui intervalos de coordenadas iguais a $[0, 5000]$ no eixo x e $[0, 5000]$ no eixo y. Além dos recursos de *software* já descritos nos testes computacionais na Seção 4.5.5, para a execução das simulações desta seção, a biblioteca com as estruturas básicas para a geração da curva HP denominada *PHquintic* também foi utilizada.

O planejamento de rota é realizado com o algoritmo RRT*-SV. O algoritmo é configurado para executar o planejamento até que a rota ótima (menor comprimento) seja encontrada para os *waypoints* $q_{início}$ e $q_{destino}$. Nesta simulação, $q_{início} = (1100, 100)$ e $q_{destino} = (2600, 2900)$. O comprimento da rota ótima, planejada pelo algoritmo de Dijkstra aplicado ao grafo de visibilidade gerado pelos vértices convexos das envoltórias de segurança dos obstáculos do ambiente de navegação considerado, é igual a $5214,94$ para $\kappa_{VANT} = 0,009$ e $5415,81$ para $\kappa_{VANT} = 0,015$. Observa-se que, quando o valor de curvatura é menor, maiores valores de d_{Obs} são obtidos. Isso justifica a diferença de comprimento entre as rotas ótimas, já que a área ocupada pelas envoltórias de segurança influência os valores de curvatura. Na Tabela 5.1, os valores de d_{Obs} e L para determinar os pontos de corte para suavização para a geração de envoltórias de segurança são resumidos para cada valor de curvatura de VANT. Na Figura 5.4 encontram-se ilustradas as rotas ótimas planejadas pelo algoritmo de Dijkstra utilizando a estrutura formada por grafo de visibilidade para cada um dos valores de curvatura de VANT considerados.

Figura 5.4 - Rotas ótimas planejadas pelo algoritmo de Dijkstra em uma estrutura formada pelo algoritmo de grafo de visibilidade.

(a)

(b)

Rotas planejadas para a) $\kappa_{VANT} = 0,009$ e b) $\kappa_{VANT} = 0,015$ para os *waypoints* $q_{início} = (1100, 100)$ e $q_{destino} = (2600, 2900)$.

Fonte: Produção do Autor.

Tabela 5.1 - Valores de L e d_{Obs} definidos para cada obstáculo do ambiente de navegação, considerando diferentes valores de κ_{VANT}.

	$\kappa_{VANT} = 0,009$		$\kappa_{VANT} = 0,015$	
Obstáculo	L	d_{Obs}	L	d_{Obs}
1	299,068	93,793	179,441	56,276
2	417,910	102,277	250,746	61,366
3	251,821	90,699	151,093	54,420
4	570,376	113,609	342,225	68,165
5	429,676	103,147	257,805	61,888

Fonte: Produção do Autor.

Após o planejamento da rota, os cantos formados pela junção dos segmentos que conectam $q_{início}$ e $q_{destino}$ da rota são suavizados usando a curva HP aplicando a abordagem descrita na Seção 5.3. Os comprimentos das rotas suavizada e não suavizada são comparados. O comprimento de cada segmento de reta da rota é calculado pela distância euclidiana entre as coordenadas dos *waypoints* que o definem. O comprimento de cada curva da rota é calculado pela Equação (5.10).

Os resultados de duas simulações são descritos a seguir, cada uma para um dos valores de curvatura do VANT especificados no início desta seção. Inicialmente, a distância d é atribuída ao valor de curvatura considerado por meio da estratégia descrita na Seção 5.3.

A Figura 5.5 e a Figura 5.6 apresentam as rotas planejadas para cada valor de curvatura de VANT considerado. Na Figura 5.5(a) e na Figura 5.5(b) estão ilustradas as rotas para $k_{VANT} = 0,009$ com e sem suavização, respectivamente. Na Figura 5.6(a) e Figura 5.6(b) estão ilustradas, respectivamente, as rotas para $\kappa_{VANT} = 0,015$, com e sem suavização. Como indicado na Tabela 5.2, os comprimentos das rotas geradas para cada valor de κ_{VANT} são menores em relação às rotas não suavizados. Para $\kappa_{VANT} = 0,009$ houve uma redução de $3,64\%$ no comprimento da rota após a suavização e para $\kappa_{VANT} = 0,015$ a redução foi de $2,23\%$. Portanto existe um indicativo de que para menores valores de curvatura, maiores valores de d serão necessários para atender a essa restrição (isso pode ser percebido pela diferença das posições das envoltórias de segurança entre a Figura 5.5 e a Figura 5.6). Dessa forma, o processo de suavização resultará em uma redução maior do comprimento

da rota após ela ser suavizada em comparação à aplicação da mesma estratégia com o uso de valores de curvatura maiores (a redução de comprimento na suavização para $\kappa_{VANT} = 0,009$ foi de $3,64\%$ enquanto que para $\kappa_{VANT} = 0,015$ a redução foi de $2,23\%$).

Tabela 5.2 - Comprimentos de rotas não suavizada e suavizada por curvas HP planejadas pelo algoritmo RRT*-SV para diferentes valores de κ_{VANT}.

κ_{VANT}	Não suavizada	Suavizada	Tempo de planejamento (s)
0,009	5415,814	5218,346	0,003
0,015	5214,943	5098,496	0,005

Fonte: Produção do Autor.

Figura 5.5 - Rota planejada pelo algoritmo RRT*-SV com e sem suavização para $\kappa_{VANT} = 0,009$.

A rota planejada pelo RRT*-SV (em vermelho) a) sem suavização b) e com suavização com curvas HP. Em b), as curvas geradas para as bordas pontiagudas são representadas pelos segmentos na cor azul. As envoltórias de segurança são representados pelos segmentos na cor verde.

Fonte: Produção do Autor.

Figura 5.6 - Rota planejada pelo algoritmo RRT*-SV com e sem suavização para $\kappa_{VANT} = 0,015$.

A rota planejada pelo RRT*-SV (em vermelho) a) sem suavização e b) com suavização com curvas HP. Em b), as curvas geradas para as bordas pontiagudas são representadas pelos segmentos na cor azul. As envoltórias de segurança são representados pelos segmentos na cor verde.

Fonte: Produção do Autor.

5.4 Considerações

Nesta seção, foi apresentado um esquema para a suavização de rotas planejadas pelo algoritmo RRT*-SV utilizando curva de Hodógrafo de Pitágoras para navegação segura de VANTs em um ambiente de navegação com obstáculos estáticos, após uma rota ser planejada pelo algoritmo RRT*-SV. Subsequentemente, os cantos pontiagudos formados pela junção dos segmentos de linha reta da rota planejada são suavizados usando curvas HP quínticas com continuidade G^2.

Um esquema de como definir a distância para a geração da curva HP baseada nas envoltórias de segurança dos obstáculos também foi apresentado. A estimativa da distância adequada para a construção das envoltórias de segurança é importante para garantir que a rota não intercepte nenhum obstáculo, evitando eventuais colisões do VANT durante sua navegação. Esta distância é definida a partir de formulações da curva HP para uma dada curvatura do veículo aéreo.

Em trabalhos futuros, a geometria das curvas HP estará associada a um modelo dinâmico de um VANT, visando controlar sua velocidade e aceleração para navegação através de rotas planejadas pela estratégia apresentada neste capítulo.

6 CONCLUSÃO

Nesta tese é apresentado um estudo de métodos de distribuições de amostras aplicados ao processo de amostragem do algoritmo denominado Árvore Aleatória de Exploração Rápida Estrela (do inglês *Rapidly-exploring Random Tree Star* - RRT*). Busca-se entender qual a influência que diferentes distribuições de amostras podem apresentar no tempo e na capacidade de planejamento de rotas de menor comprimento do algoritmo RRT*. Identificando as distribuições que reduzem tanto o tempo de planejamento quanto o comprimento das rotas de navegação, podem ser propostos novos processos de amostragem não-uniforme/informada para serem acoplados ao algoritmo considerado. Além disso, é possível utilizar essas novas estratégias propostas em algoritmos de planejamento de rotas para a obtenção de uma maior capacidade de autonomia em veículos autônomos, especialmente em Veículos Aéreos Não Tripulados (VANTs).

Foram realizados estudos de diferentes estratégias de amostragem nesse algoritmo utilizando os vértices convexos das envoltórias de segurança dos obstáculos e métodos de dispersão ótima de amostras, conhecidas como grades de Sukharev. Ao longo da realização do estudo mencionado, foi proposto o algoritmo RRT*-SV (RRT* Sukharev Vértices), que mescla uma amostragem baseada nos vértices convexos das envoltórias de segurança de obstáculos com grades de Sukharev.

O algoritmo RRT*-SV foi avaliado experimentalmente por testes computacionais considerando ambientes de navegação com diferentes distribuições espaciais de obstáculos estáticos. Eles foram comparados a algoritmos com melhorias do gênero, propostos na literatura, como RRT*-Smart, RRT*-Informed e BIT*. Em comparação com os algoritmos citados foi verificado, através de diferentes conjuntos de testes computacionais, que o algoritmo RRT*-SV, considerando ambientes de navegação com diferentes distribuições espaciais e quantidades de obstáculos, demonstrou:

a) menor tempo necessário para planejar uma primeira rota na grande maioria dos testes realizados;

b) capacidade de planejar rotas de menor comprimento mais rápido na maioria dos testes realizados;

c) maior robustez devido à menor influência sofrida por parâmetros como tamanho do passo Δq do algoritmo RRT;

d) sofrer perda maior de desempenho computacional à medida que a quan-

tidade de obstáculos aumenta em comparação com os demais algoritmos considerados.

O melhor desempenho do algoritmo RRT*-SV em certos cenários, pode ser explicado pela menor influência sofrida pelo mesmo da orientação de Voronoi, pois utiliza os vértices convexos do ambiente de navegação e a grade de Sukharev para rapidamente criar arestas para fora destas regiões. Desta forma, é possível planejar rotas de menor comprimento em um menor tempo de planejamento, como demonstrado nos testes computacionais realizados nesta tese. Entretanto, em ambientes de navegação com uma grande quantidade de obstáculos (por exemplo, 10 000 obstáculos), o RRT*-SV perde desempenho devido à grande quantidade de testes de colisão envolvida em suas operações.

Finalmente, uma estratégia de suavização de rotas baseada em curvas de Hodógrafo de Pitágoras (também denominadas curvas HP) é aplicada nas rotas planejadas pelo algoritmo RRT*-SV. Após o planejamento de uma rota pelo RRT*-SV, os cantos pontiagudos formados pela junção dos segmentos de reta dessa rota são suavizados usando curvas HP. Entretanto, pelo fato de uma rota ser formada por vértices convexos das envoltórias de segurança e da rota ter seu comprimento reduzido ao ser suavizada, é possível que as curvas geradas interceptem os limites dos obstáculos dessas envoltórias. Assim, foi utilizado um esquema para a definição da distância mínima para a geração das envoltórias de segurança dos obstáculos. A estimativa dessa distância é importante para garantir que a rota não intercepte nenhum obstáculo ao ser suavizada, evitando eventuais colisões do VANT durante sua navegação. Essa distância é definida a partir de formulações da curva HP para um dado valor de curvatura do VANT.

Em adição às contribuições declaradas anteriormente, nesta tese também são apresentados os resultados da aplicação de uma Revisão Sistemática da Literatura (RSL) com tema sobre estratégias de amostragem não uniforme/informada em algoritmos baseados em RRT. Bastante popular nas áreas de medicina e engenharia de software, essa metodologia de revisão utiliza um protocolo de pesquisa para guiar o processo de busca de trabalhos acadêmicos que potencialmente contribuam para o estado da arte de uma determinada área de conhecimento. Sua finalidade é reduzir o viés inerente à abordagem tradicional de revisão, onde os estudos da literatura em uma determinada área de conhecimento são selecionados baseados na experiência do(a) pesquisador(a), o(a) qual pode conduzi-la com subjetividade e seleção não objetiva.

Para a realização da RSL, inicialmente duas questões científicas são desenvolvidas, a partir das quais é delineado o escopo da revisão. A partir dessas questões, um protocolo de pesquisa é desenvolvido em conjunto com uma equipe de pesquisadores especialistas. Baseado no protocolo, estudos acadêmicos são pesquisados (processo de busca) em diferentes bases de consulta de trabalhos acadêmicos (motores de busca). Os trabalhos retornados de cada base passam por um processo de seleção onde critérios de inclusão e exclusão definidos no protocolo de pesquisa são aplicados.

Na RSL apresentada nesta tese, foram selecionados 1136 estudos, dos quais 53 foram identificados alegando conter solução dentro do tópico definido. Os resultados dessa revisão resultaram em uma nova classificação das estratégias de amostragem não uniforme/informada, e foi utilizada como instrumento de validação dos estudos apresentados nesta tese. Para a comunidade científica, os dados extraídos dos estudos selecionados podem ajudar no desenvolvimento de novos projetos de pesquisa. Analisando os resultados da RSL, uma equipe de pesquisa pode verificar o que já foi proposto como uma solução de amostragem não uniforme/informada para versões específicas do algoritmo RRT e identificar acoplamentos ainda não propostos. Além disso, uma equipe de projeto tecnológico pode tomar conhecimento de quais estratégias existem através da RSL proposta e analisar quais delas podem preencher alguns requisitos de seu projeto.

Em ordem cronológica, as publicações associadas a esta tese são

a) VÉRAS, L. G.; MEDEIROS, F. L.; GUIMARÃES, L. F. Application of rapidly exploring random tree star-smart and G^2 quintic pythagorean hodograph curves to the UAV path planning problem. **International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering**, World Academy of Science, Engineering and Technology, v. 12, n. 5, p. 522 - 530, 2018.

b) VÉRAS, L. G. D. O.; MEDEIROS, F. L. L.; GUIMARÃES, L. N. F. Rapidly exploring random tree* with a sampling method based on sukharev grids and convex vertices of safety hulls of obstacles. **International Journal of Advanced Robotic Systems**, v. 16, n. 1, 2019.

c) VÉRAS, L. G. D. O.; MEDEIROS, F. L. L.; GUIMARÃES, L. N. F. Systematic literature review of sampling process in rapidly-exploring random trees. **IEEE Access**, v. 7, p. 50933 - 50953, 2019.

É importante ressaltar que o artigo Véras et al. (2018a) (o primeiro da lista anterior) foi primeiramente publicado em evento acadêmico e posteriormente selecionado para publicação em revista. Assim, a primeira publicação do trabalho citado encontra-se em Véras et al. (2018b).

6.1 Trabalhos futuros

Neste trabalho, o estudo proposto sobre estratégias de distribuição de amostras considerou somente cenários em duas dimensões. Futuramente, pode-se realizar estudos considerando problemas de planejamento de rota com maiores quantidades de dimensões (em três dimensões ou mais, dependendo da definição do problema). Além disso, considerando o algoritmo proposto RRT*-SV e o processo de suavização utilizado nesta tese, pretende-se associar a geometria das curvas HP com modelos dinâmicos de VANT. Durante a etapa de expansão de árvore do algoritmo RRT*-SV, a dinâmica do VANT pode ser tomada como referência para a geração das arestas da árvore. Assim, em cada iteração do algoritmo RRT*-SV, seria possível estimar uma rota livre de colisão e dinamicamente viável entre a raiz (posição inicial de navegação) e qualquer outro nó da árvore. Também pode vir a ser desenvolvido um controle de navegação que seja adequado para as rotas dinamicamente viáveis geradas pelo algoritmo RRT*-SV e pelas curvas HP.

REFERÊNCIAS BIBLIOGRÁFICAS

ABBADI, A.; MATOUSEK, R. Hybrid rule-based motion planner for mobile robot in cluttered workspace. **Soft Computing**, v. 22, n. 6, p. 1815–1831, 2018. 8, 62, 68

AHMADYAN, S. N.; KUMAR, J. A.; VASUDEVAN, S. Goal-oriented stimulus generation for analog circuits. In: ANNUAL DESIGN AUTOMATION CONFERENCE, 49., 2012, San Francisco, CA, USA. **Proceedings ...** [S.l.]: ACM, 2012. p. 1018–1023. 8, 50, 52

AKGUN, B.; STILMAN, M. Sampling heuristics for optimal motion planning in high dimensions. In: INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS, 2011, San Francisco, CA, USA. **Proceedings...** [S.l.]: IEEE, 2011. 47, 69, 70

ALIA, C.; GILLES, T.; REINE, T.; ALI, C. Local trajectory planning and tracking of autonomous vehicles, using clothoid tentacles method. In: INTELLIGENT VEHICLES SYMPOSIUM, 4., 2015. **Proceedings...** [S.l.]: IEEE, 2015. p. 674–679. 4

AN, B.; KIM, J.; PARK, F. C. An adaptive stepsize RRT planning algorithm for open-chain robots. **IEEE Robotics and Automation Letters**, v. 3, n. 1, p. 312–319, 2018. 38

ANDERSON, E. P.; BEARD, R. W.; MCLAIN, T. W. Real-time dynamic trajectory smoothing for unmanned air vehicles. **IEEE Transactions on Control Systems Technology**, v. 13, n. 3, p. 471–477, 2005. 4

ARKIN, R. C. Path planning for a vision-based autonomous robot. In: CAMBRIDGE SYMPOSIUM INTELLIGENT ROBOTICS SYSTEMS, 1987. **Proceedings...** [S.l.], 1987. p. 240–250. 4

ARSLAN, O.; TSIOTRAS, P. Use of relaxation methods in sampling-based algorithms for optimal motion planning. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2013. **Proceedings...** [S.l.], 2013. p. 2421–2428. 82

_____. Machine learning guided exploration for sampling-based motion planning algorithms. In: INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS, 2015. **Proceedings...** [S.l.], 2015. p. 2646–2652. 80, 82

ARYA, S.; MOUNT, D. M. Approximate range searching. **Computational Geometry**, v. 17, n. 3-4, p. 135–152, 2000. 125

ARYA, S.; MOUNT, D. M.; NETANYAHU, N. S.; SILVERMAN, R.; WU, A. Y. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. **Journal of the ACM (JACM)**, v. 45, n. 6, p. 891–923, 1998. 38

AURENHAMMER, F. Voronoi diagrams - a survey of a fundamental geometric data structure. **ACM Computing Surveys (CSUR)**, v. 23, n. 3, p. 345–405, 1991. 35

AUSTIN, R. **Unmanned aircraft systems: UAVS design, development and deployment**. [S.l.]: John Wiley & Sons, 2011. 16, 17

BARRAQUAND, J.; LATOMBE, J.-C. Robot motion planning: a distributed representation approach. **The International Journal of Robotics Research**, v. 10, n. 6, p. 628–649, 1991. 4

BENTLEY, J. L. Multidimensional binary search trees used for associative searching. **Communications of the ACM**, v. 18, n. 9, p. 509–517, 1975. 94

BIALKOWSKI, J.; OTTE, M.; KARAMAN, S.; FRAZZOLI, E. Efficient collision checking in sampling-based motion planning via safety certificates. **The International Journal of Robotics Research**, v. 35, n. 7, p. 767–796, 2016. 216

BLIN, N.; TAÏIX, M.; FILLATREAU, P.; FOURQUET, J. I-RRT-c: Interactive motion planning with contact. In: INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS, 2016. **Proceedings...** [S.l.], 2016. p. 4244–4249. ISSN 2153-0866. 73, 74

BOARDMAN, B.; HARDEN, T.; MARTÍNEZ, S. Improved performance of asymptotically optimal rapidly exploring random trees. **Journal of Dynamic Systems, Measurement, and Control**, v. 141, n. 1, p. 011002, 2019. 69, 72

BOSKOVIC, J. D.; PRASANTH, R.; MEHRA, R. K. A multi-layer autonomous intelligent control architecture for unmanned aerial vehicles. **Journal of Aerospace Computing, Information, and Communication**, v. 1, n. 12, p. 605–628, 2004. 18

BOX, G. E.; HUNTER, W. G.; HUNTER, J. S. **Statistics for experimenters**. [S.l.]: New York: John Wiley and Sons, 1978. 158

BRÖNNIMANN, H.; FABRI, A.; GIEZEMAN, G.; HERT, S.; HOFFMANN, M.; KETTNER, L.; PION, S.; SCHIRRA, S. **CGAL user and reference manual: all parts**. [S.l.]: Release, 2009. 167

BROOKS, R. A.; LOZANO-PEREZ, T. A subdivision algorithm in configuration space for findpath with rotation. **IEEE Transactions on Systems, Man, and Cybernetics**, n. 2, p. 224–233, 1985. 27

BRUCE, J. R.; VELOSO, M. M. Safe multirobot navigation within dynamics constraints. **Proceedings of the IEEE**, v. 94, n. 7, p. 1398–1411, 2006. 8, 32

BUDGEN, D.; BRERETON, P. Performing systematic literature reviews in software engineering. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, 28., 2006. **Proceedings...** [S.l.]: ACM, 2006. p. 1051–1052. 10

CACCIA, M.; BIBULI, M.; BONO, R.; BRUZZONE, G. Basic navigation, guidance and control of an unmanned surface vehicle. **Autonomous Robots**, v. 25, n. 4, p. 349–365, 2008. 2

CANNY, J. **The complexity of robot motion planning**. [S.l.]: MIT press, 1988. 27

CGAL, C. **Computational geometry algorithms library**. 2008. Disponível em: <https://www.cgal.org/>. 33

CHAO, N.; LIU, Y.-k.; XIA, H.; BAI, L. Minimum dose path planning in complex radioactive environments with sampling-based algorithms. In: INTERNATIONAL CONFERENCE ON NUCLEAR ENGINEERING, 25., 2017. **Proceedings...** [S.l.]: ASME, 2017. 69, 71

CHENG, P.; LAVALLE, S. M. Reducing metric sensitivity in randomized trajectory design. In: INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS, 2001. **Proceedings...** [S.l.]: IEEE, 2001. v. 1, p. 43–48. 8, 32

CHOSET, H. M.; HUTCHINSON, S.; LYNCH, K. M.; KANTOR, G.; BURGARD, W.; KAVRAKI, L. E.; THRUN, S. **Principles of robot motion: theory, algorithms, and implementation**. [S.l.]: MIT press, 2005. 1

CHOUDHURY, S.; GAMMELL, J. D.; BARFOOT, T. D.; SRINIVASA, S. S.; SCHERER, S. Regionally accelerated batch informed trees (rabit*): a framework to integrate local information into optimal path planning. In: INTERNATIONAL

CONFERENCE ON ROBOTICS AND AUTOMATION, 2016. **Proceedings...** [S.l.]: IEEE, 2016. p. 4207–4214. 216

CLAPPER, J.; YOUNG, J.; CARTWRIGHT, J.; GRIMES, J. **Unmanned systems roadmap 2007-2032**. [S.l.]: Office of the Secretary of Defense, 2007. 15

DADKHAH, N.; METTLER, B. Survey of motion planning literature in the presence of uncertainty: considerations for UAV guidance. **Journal of Intelligent & Robotic Systems**, v. 65, n. 1-4, p. 233–246, 2012. 25

DALAMAGKIDIS, K.; VALAVANIS, K. P.; PIEGL, L. A. **On integrating unmanned aircraft systems into the national airspace system: issues, challenges, operational restrictions, certification, and recommendations**. [S.l.]: Springer Science & Business Media, 2011. 16

DEVAURS, D.; SIMÉON, T.; CORTÉS, J. Parallelizing RRT on distributed-memory architectures. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2011. **Proceedings...** [S.l.]: IEEE, 2011. p. 2261–2266. 216

_____. Efficient sampling-based approaches to optimal path planning in complex cost spaces. In: AKIN, H. L.; AMATO, N. M.; ISLER, V.; DER STAPPEN, A. F. (Ed.). **Algorithmic foundations of robotics XI**. [S.l.]: Springer, 2015. p. 143–159. 216

DIJKSTRA, E. W. A note on two problems in connexion with graphs. **Numerische mathematik**, v. 1, n. 1, p. 269–271, 1959. 4, 9, 91, 211

DO, Q. H.; HAN, L.; NEJAD, H. T. N.; MITA, S. Safe path planning among multi obstacles. In: INTELLIGENT VEHICLES SYMPOSIUM, 4., 2011. **Proceedings...** [S.l.]: IEEE, 2011. p. 332–338. 4

DONG, B.; FAROUKI, R. T. Phquintic: a library of basic functions for the construction and analysis of planar quintic pythagorean-hodograph curves. **ACM Transactions on Mathematical Software (TOMS)**, v. 41, n. 4, p. 28, 2015. 173

DU, M.; MEI, T.; LIANG, H.; CHEN, J.; HUANG, R.; ZHAO, P. Drivers visual behavior-guided RRT motion planner for autonomous on-road driving. **Sensors**, v. 16, n. 1, p. 102, 2016. 61, 66

DUBINS, L. E. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. **American Journal of Mathematics**, v. 79, n. 3, p. 497–516, 1957. 4

ELBANHAWI, M.; SIMIC, M. Sampling-based robot motion planning: a review. **IEEE Access**, v. 2, p. 56–77, 2014. 29, 38, 47, 216, 218

FANG, Z.; LUAN, C.; SUN, Z. A 2d voronoi-based random tree for path planning in complicated 3d environments. In: INTERNATIONAL CONFERENCE ON INTELLIGENT AUTONOMOUS SYSTEMS, 2016. **Proceedings...** [S.l.]: Springer, 2016. p. 433–445. 60, 66

FAROUKI, R. T. The conformal map $z \to z^2$ of the hodograph plane. **Computer Aided Geometric Design**, v. 11, n. 4, p. 363–390, 1994. 173

FAROUKI, R. T.; GIANNELLI, C.; MUGNAINI, D.; SESTINI, A. Path planning with pythagorean-hodograph curves for unmanned or autonomous vehicles. **Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering**, v. 232, n. 7, p. 1361–1372, 2018. 169, 171, 175, 177

FAROUKI, R. T.; NITTLER, K. M. Efficient high-speed cornering motions based on continuously-variable feedrates. i. real-time interpolator algorithms. **The International Journal of Advanced Manufacturing Technology**, v. 87, n. 9-12, p. 3557–3568, 2016. 11

FAROUKI, R. T.; SAKKALIS, T. Pythagorean hodographs. **IBM Journal of Research and Development**, v. 34, n. 5, p. 736–752, 1990. 11, 169, 171

FAROUKI, T. **Pythagorean-hodograph curves**. Berlin: Springer-Verlag, 2008. Berlin Heidelberg. 169, 170, 171, 173

FEI, P.; YAO, Z. Triangle sampling path planning in environment with danger zones for assembly/disassembly. **Electrotechnical Review**, p. 167–172, 2012. 56, 57

FERGUSON, D.; KALRA, N.; STENTZ, A. Replanning with RRTs. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2006. **Proceedings...** [S.l.]: IEEE, 2006. p. 1243–1248. 52

FINKEL, R. A.; BENTLEY, J. L. Quad trees a data structure for retrieval on composite keys. **Acta Informatica**, v. 4, n. 1, p. 1–9, 1974. 4

FLAVIGNE, D.; TAIX, M.; FERRE, E. Interactive motion planning for assembly tasks. In: INTERNATIONAL SYMPOSIUM ON ROBOT AND HUMAN INTERACTIVE COMMUNICATION, 18., 2009. **Proceedings...** [S.l.]: IEEE, 2009. p. 430–435. 74

FLOREANO, D.; WOOD, R. J. Science, technology and the future of small autonomous drones. **Nature**, v. 521, n. 7553, p. 460, 2015. 1

FRAGKOPOULOS, C.; GRAESER, A. Extended RRT algorithm with dynamic n-dimensional cuboid domains. In: INTERNATIONAL CONFERENCE ON OPTIMIZATION OF ELECTRICAL AND ELECTRONIC EQUIPMENT, 12., 2010. **Proceedings...** [S.l.]: IEEE, 2010. p. 851–857. 59, 64

FRAZZOLI, E. **Robust hybrid control for autonomous vehicle motion planning**. Tese (Doutorado em Engenharia Aeronáutica) — Massachusetts Institute of Technology, Cambridge, 2001. 29

FRAZZOLI, E.; DAHLEH, M. A.; FERON, E. Real-time motion planning for agile autonomous vehicles. **Journal of Guidance, Control, and Dynamics**, v. 25, n. 1, p. 116–129, 2002. 8, 32

GAGE, D. W. **UGV history 101: a brief history of Unmanned Ground Vehicle (UGV) development efforts**. [S.l.: s.n.], 1995. 2

GAMMELL, J. D. **Informed anytime search for continuous planning problems**. Tese (Doutorado em Engenharia Aeroespacial) — University of Toronto, Toronto, 2017. 8

GAMMELL, J. D.; SRINIVASA, S. S.; BARFOOT, T. D. Informed RRT*: optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. **arXiv preprint arXiv:1404.2334**, 2014. 8, 47, 76, 77, 85, 126, 224, 229, 230

_____. Batch informed trees (bit*): sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2015. **Proceedings...** [S.l.]: IEEE, 2015. p. 3067–3074. 8, 76, 77, 85, 126, 127, 144, 232, 235, 236, 237

GAO, X.; WU, H.; ZHAI, L.; SUN, H.; JIA, Q.; WANG, Y.; WU, L. A rapidly exploring random tree optimization algorithm for space robotic manipulators

guided by obstacle avoidance independent potential field. **International Journal of Advanced Robotic Systems**, v. 15, n. 3, p. 1729881418782240, 2018. 51, 54

GOERZEN, C.; KONG, Z.; METTLER, B. A survey of motion planning algorithms from the perspective of autonomous UAV guidance. **Journal of Intelligent and Robotic Systems**, Springer, v. 57, n. 1-4, p. 65–100, 2010. 1, 2, 3, 18, 19, 20, 22

GROCHOLSKY, B.; KELLER, J.; KUMAR, R. V.; PAPPAS, G. J. Cooperative air and ground surveillance. **IEEE Robotics & Automation Magazine**, v. 13, n. 3, p. 16–26, 2006. 1

GUENNEBAUD, G. et al. Eigen: a c++ linear algebra library. v. 22, 2014. Disponível em: <http://eigen.tuxfamily.org>. Acesso em: 14 dez. 2018. 167

GUNDLACH, J. **Designing unmanned aircraft systems: a comprehensive approach**. [S.l.]: American Institute of Aeronautics and Astronautics, 2012. 1, 15, 16, 18

HART, P. E.; NILSSON, N. J.; RAPHAEL, B. A formal basis for the heuristic determination of minimum cost paths. **IEEE transactions on Systems Science and Cybernetics**, v. 4, n. 2, p. 100–107, 1968. 4, 91

HOLLAND, J. H. **Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence.** [S.l.]: U Michigan Press, 1975. 4

HSU, D.; KINDEL, R.; LATOMBE, J.-C.; ROCK, S. Randomized kinodynamic motion planning with moving obstacles. **The International Journal of Robotics Research**, v. 21, n. 3, p. 233–255, 2002. 29

HSU, D.; LATOMBE, J.-C.; MOTWANI, R. Path planning in expansive configuration spaces. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 1997. **Proceedings...** [S.l.]: IEEE, 1997. p. 2719–2726. 29

HUAZHONG, L.; YONGSHENG, L.; MEINI, W.; TANGREN, D. Design and implementation of improved RRT algorithm for collision free motion planning of high-dimensional robot in complex environment. In: INTERNATIONAL CONFERENCE ON COMPUTER SCIENCE AND NETWORK TECHNOLOGY, 2., 2012. **Proceedings...** [S.l.]: IEEE, 2012. p. 1391–1397. 50, 52

HUH, J.; LEE, D. D. Learning high-dimensional mixture models for fast collision detection in rapidly-exploring random trees. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2016. **Proceedings...** [S.l.]: IEEE, 2016. p. 63–69. 80, 82

INTERNATIONAL STANDARDIZATION ORGANIZATION (ISO). **DIS I 8373:2012: robots and robotic devices–vocabulary**. Genebra, 2012. 1

ISLAM, F.; NASIR, J.; MALIK, U.; AYAZ, Y.; HASAN, O. RRT*-Smart: rapid convergence implementation of RRT* towards optimal solution. In: INTERNATIONAL CONFERENCE ON MECHATRONICS AND AUTOMATION, 2012. **Proceedings...** [S.l.]: IEEE, 2012. p. 1651–1656. 5, 8, 69, 70, 125, 127, 225, 227, 228

JANSON, L.; ICHTER, B.; PAVONE, M. Deterministic sampling-based motion planning: Optimality, complexity, and performance. **The International Journal of Robotics Research**, v. 37, n. 1, p. 46–61, 2018. 9, 87

JANSON, L.; SCHMERLING, E.; CLARK, A.; PAVONE, M. Fast marching tree: a fast marching sampling-based method for optimal motion planning in many dimensions. **The International journal of robotics research**, v. 34, n. 7, p. 883–921, 2015. 29

JOUANDEAU, N.; CHERIF, A. A. Fast approximation to gaussian obstacle sampling for randomized motion planning. **IFAC Proceedings Volumes**, v. 37, n. 8, p. 591–596, 2004. 79, 81

JUNIPER, A. **The complete guide to drones**. [S.l.]: Hachette UK, 2018. 1

KANG, G.; KIM, Y. B.; YOU, W. S.; LEE, Y. H.; OH, H. S.; MOON, H.; CHOI, H. R. Sampling-based path planning with goal oriented sampling. In: INTERNATIONAL CONFERENCE ON ADVANCED INTELLIGENT MECHATRONICS, 2016. **Proceedings...** [S.l.]: IEEE, 2016. p. 1285–1290. 50, 54

KARAMAN, S.; FRAZZOLI, E. Sampling-based motion planning with deterministic μ-calculus specifications. In: CONFERENCE ON DECISION AND CONTROL, 48., 2008. **Proceedings...** Shanghai, China, 16–18 December 2009, pp. 2222–2229: IEEE, 2009. 7

_____. Incremental sampling-based algorithms for optimal motion planning. **Robotics Science and Systems VI**, v. 104, p. 2, 2010. 7, 32, 38, 43, 125

_____. Sampling-based algorithms for optimal motion planning. **The International Journal of Robotics Research**, v. 30, n. 7, p. 846–894, 2011. 27, 38, 46

KARAMAN, S.; WALTER, M. R.; PEREZ, A.; FRAZZOLI, E.; TELLER, S. Anytime motion planning using the RRT. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2011. **Proceedings...** [S.l.]: IEEE, 2011. p. 1478–1483. 47, 216

KAVRAKI, L.; LATOMBE, J.-C. Randomized preprocessing of configuration for fast path planning. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 1994. **Proceedings...** [S.l.]: IEEE, 1994. p. 2138–2145. 27

KAVRAKI, L. E.; SVESTKA, P.; LATOMBE, J.-C.; OVERMARS, M. H. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. **IEEE Transactions on Robotics and Automation**, v. 12, n. 4, p. 566–580, 1996. 4, 27, 29

KAYACAN, E.; RAMON, H.; SAEYS, W. Robust trajectory tracking error model-based predictive control for unmanned ground vehicles. **IEEE/ASME Transactions on Mechatronics**, v. 21, n. 2, p. 806–814, 2016. 2

KHANMOHAMMADI, S.; MAHDIZADEH, A. Density avoided sampling: an intelligent sampling technique for rapidly-exploring random trees. In: INTERNATIONAL CONFERENCE ON HYBRID INTELLIGENT SYSTEMS, 8., 2008. **Proceedings...** [S.l.]: IEEE, 2008. p. 672–677. 59, 64

KHATIB, O. Real-time obstacle avoidance for manipulators and mobile robots. In: COX, I. J.; WILFONG, G. T. (Ed.). **Autonomous robot vehicles**. [S.l.]: Springer, 1986. p. 396–404. 27, 49

KIM, D.; LEE, J.; YOON, S.-e. Cloud RRT*: sampling cloud based RRT*. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2014. **Proceedings...** [S.l.]: IEEE, 2014. p. 2519–2526. 60, 65

KIM, J.; ESPOSITO, J. M. Adaptive sample bias for rapidly-exploring random trees with applications to test generation. In: AMERICAN CONTROL CONFERENCE, 2005. **Proceedings...** [S.l.]: IEEE, 2005. p. 1166–1172. 79, 81

KIM, J.; ESPOSITO, J. M.; KUMAR, V. **An RRT-based algorithm for testing and validating multi-robot controllers**. [S.l.: s.n.], 2005. 79, 81

KIM, M.-C.; SONG, J.-B. Informed RRT* towards optimality by reducing size of hyperellipsoid. In: INTERNATIONAL CONFERENCE ON ADVANCED INTELLIGENT MECHATRONICS, 2015. **Proceedings...** [S.l.]: IEEE, 2015. p. 244–248. 76, 77

KITCHENHAM, B. **Procedures for performing systematic reviews**. Keele: Keele University, 2004. (Keele University Technical Report TR/SE-0401). 10, 11

KITCHENHAM, B.; CHARTERS, S. **Guidelines for performing systematic literature reviews in software engineering**. 2007. Disponível em: <https://userpages.uni-koblenz.de/~laemmel/esecourse/slides/slr.pdf>. 8, 48, 215, 217, 224

KOBILAROV, M. Cross-entropy motion planning. **The International Journal of Robotics Research**, v. 31, n. 7, p. 855–871, 2012. 69, 70

KOENIG, S.; LIKHACHEV, M.; FURCY, D. Lifelong planning a*. **Artificial Intelligence**, v. 155, n. 1-2, p. 93–146, 2004. 232

KONG, Y.; PAN, Y.; CHEN, X. A kind of two-stage RRT algorithm for robotic path planning. **International Journal of Wireless and Mobile Computing**, v. 6, n. 1, p. 34–38, 2013. 50, 53

KUFFNER, J. J.; LAVALLE, S. M. RRT-connect: an efficient approach to single-query path planning. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2000. **Proceedings...** [S.l.]: IEEE, 2000. v. 2, p. 995–1001. 8, 33, 35, 38

LATOMBE, J.-C. **Robot motion planning**. [S.l.]: Springer Science, 2012. 9, 24, 90

LAVALLE, S. M. Rapidly-exploring random trees: a new tool for path planning. Citeseer, 1998. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.35.1853>. 4, 29, 216

_____. **Planning algorithms**. [S.l.]: Cambridge University Press, 2006. 1, 3, 5, 8, 9, 24, 27, 28, 33, 85, 86, 92

LAVALLE, S. M.; KUFFNER, J.; JAMES, J. Randomized kinodynamic planning. **The International Journal of Robotics Research**, v. 20, n. 5, p. 378–400, 2001. 27, 29

LEE, J.; KWON, O.; ZHANG, L.; YOON, S.-e. SR-RRT: selective retraction-based RRT planner. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2012. **Proceedings...** [S.l.]: IEEE, 2012. p. 2543–2550. 8, 73, 74

LI, D.; LI, Q.; CHENG, N.; SONG, J. Sampling-based real-time motion planning under state uncertainty for autonomous micro-aerial vehicles in gps-denied environments. **Sensors**, Multidisciplinary Digital Publishing Institute, v. 14, n. 11, p. 21791–21825, 2014. 8, 32

LIN, N.; ZHANG, Y. An adaptive RRT based on dynamic step for UAV path planning. **Journal of Information and Computational Science**, v. 12, n. 5, p. 1975–1983, 2015. 8, 32

LINDEMANN, S. R.; LAVALLE, S. M. Incremental low-discrepancy lattice methods for motion planning. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2003. **Proceedings...** [S.l.]: IEEE, 2003. v. 3, p. 2920–2927. 9, 87

_____. Incrementally reducing dispersion by increasing voronoi bias in RRTs. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2004. **Proceedings...** [S.l.]: IEEE, 2004. v. 4, p. 3251–3257. 8, 37, 59, 63, 224

_____. Current issues in sampling-based motion planning. In: INTERNATIONAL SYMPOSIUM ROBOTICS RESEARCH, 11., 2005. **Proceedings...** [S.l.]: Springer, 2005. p. 36–54. 7, 35, 47, 48

_____. Steps toward derandomizing RRTs. In: KOZLOWSKI, K. (Ed.). **Robot motion and control**. [S.l.]: Springer, 2006. p. 287–300. 8, 47, 48, 59, 63, 224

LIU, W.; ANG, M. H. Incremental sampling-based algorithm for risk-aware planning under motion uncertainty. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2014. **Proceedings...** [S.l.]: IEEE, 2014. p. 2051–2058. 8, 33

LOZANO-PÉREZ, T.; WESLEY, M. A. An algorithm for planning collision-free paths among polyhedral obstacles. **Communications of the ACM**, v. 22, n. 10, p. 560–570, 1979. 27

LUDERS, B.; KOTHARI, M.; HOW, J. Chance constrained RRT for probabilistic robustness to environmental uncertainty. In: AIAA GUIDANCE, NAVIGATION, AND CONTROL CONFERENCE, 2010. **Proceedings...** [S.l.]: AIAA, 2010. p. 8160. 8, 33

LUDINGTON, B.; JOHNSON, E.; VACHTSEVANOS, G. Vision based navigation and target tracking for unmanned aerial vehicles. In: VALAVANIS, K. P. (Ed.). **Advances in unmanned aerial vehicles**. [S.l.]: Springer, 2007. p. 245–266. 1

LUMELSKY, V. Algorithmic issues of sensor-based robot motion planning. In: CONFERENCE ON DECISION AND CONTROL, 26., 1987. **Proceedings...** [S.l.]: IEEE, 1987. v. 26, p. 1796–1801. 25

MA, L.; XUE, J.; KAWABATA, K.; ZHU, J.; MA, C.; ZHENG, N. Efficient sampling-based motion planning for on-road autonomous driving. **IEEE Transactions on Intelligent Transportation Systems**, v. 16, n. 4, p. 1961–1976, 2015. 8, 32, 46

MAEKAWA, T.; NODA, T.; TAMURA, S.; OZAKI, T.; MACHIDA, K.-i. Curvature continuous path generation for autonomous vehicle using b-spline curves. **Computer-Aided Design**, v. 42, n. 4, p. 350–359, 2010. 4

MARTIN, S. R.; WRIGHT, S. E.; SHEPPARD, J. W. Offline and online evolutionary bi-directional RRT algorithms for efficient re-planning in dynamic environments. In: INTERNATIONAL CONFERENCE ON AUTOMATION SCIENCE AND ENGINEERING, 2007. **Proceedings...** [S.l.]: IEEE, 2007. p. 1131–1136. 59, 64

MAZER, E.; AHUACTZIN, J. M.; BESSIERE, P. The ariadne's clew algorithm. **Journal of Artificial Intelligence Research**, v. 9, p. 295–316, 1998. 29

MCCOURT, M.; TON, C. T.; MEHTA, S. S.; CURTIS, J. W. Adaptive step-length RRT algorithm for improved coverage. In: AIAA GUIDANCE, NAVIGATION, AND CONTROL CONFERENCE, 2016. **Proceedings...** [S.l.]: AIAA, 2016. p. 0638. 38

MEDEIROS, F. L. L. **Planejamento de trajetórias para veículos aéreos não tripulados usando modelagem computacional de ambientes de navegação através de grafos de visibilidade e modelos digitais de elevação**. Tese (Doutorado em Computação Aplicada) — Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2012. 2, 17, 18

MEDEIROS, F. L. L.; SHIGUEMORI, E. H.; MONTEIRO, M.; DOMICIANO, M.; MARTINS, M. Verificação automática de situações de colisão na navegação de veículos aéreos não tripulados. In: ENCONTRO NACIONAL DE INTELIGÊNCIA ARTIFICIAL, 6., 2007. **Anais...** [S.l.], 2007. p. 932–941. 216

MENG, Z.; QIN, H.; SUN, H.; SHEN, X.; ANG, M. H. Obstacle-guided informed planning towards robot navigation in cluttered environments. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND BIOMIMETICS, 2017. **Proceedings...** [S.l.]: IEEE, 2017. p. 332–337. 56, 58

MOSES, E.; BOON, D.; ANITHA, G. Goal directed approach to autonomous motion planning for unmanned vehicles. **Defence Science Journal**, v. 67, n. 1, 2017. 50, 54

MOTWANI, R.; MOTWANI, M.; HARRIS, F. C. Uniform and efficient exploration of state space using kinodynamic sampling-based planners. In: THOMAS, F.; GARCIA, A. P. (Ed.). **Computational kinematics**. [S.l.]: Springer, 2014. p. 67–74. 60, 65

MOUNT, D. M. **ANN Programming Manual**. Maryland: University of Maryland, 2010. 167

NASIR, J.; ISLAM, F.; MALIK, U.; AYAZ, Y.; HASAN, O.; KHAN, M.; MUHAMMAD, M. S. RRT*-smart: a rapid convergence implementation of RRT. **International Journal of Advanced Robotic Systems**, v. 10, n. 7, p. 299, 2013. 8, 46, 47, 85

NEX, F.; REMONDINO, F. UAV for 3d mapping applications: a review. **Applied Geomatics**, v. 6, n. 1, p. 1–15, 2014. 2

NILSSON, N. J. **A mobile automaton: an application of artificial intelligence techniques**. [S.l.: s.n.], 1969. 4, 211

NOREEN, I.; KHAN, A.; HABIB, Z. Optimal path planning using RRT* based approaches: a survey and future directions. **International Journal of Advanced Computer Science and Applications**, v. 7, n. 11, p. 97–107, 2016. 48, 224

NUSSENZVEIG, H. M. **Curso de física básica, 1: mecânica**. [S.l.]: E. Blucher, 2013. 11

PALMIERI, L.; KOENIG, S.; ARRAS, K. O. RRT-based nonholonomic motion planning using any-angle path biasing. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2016. **Proceedings...** [S.l.]: IEEE, 2016. p. 2775–2781. 60, 66

PARK, C.; PAN, J.; MANOCHA, D. Poisson-RRT. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2014. **Proceedings...** [S.l.]: IEEE, 2014. p. 4667–4673. 8, 37, 79, 81

PENG, F.; ZHAO, Y. Random triangle sampling path planning of assembly/disassembly in environment with dangerzones. In: INTERNATIONAL CONFERENCE ON MEASURING TECHNOLOGY AND MECHATRONICS AUTOMATION, 2010. **Proceedings...** [S.l.]: IEEE, 2010. v. 2, p. 972–976. 56, 57

PENROSE, M. et al. **Random geometric graphs**. [S.l.]: Oxford university press, 2003. 234

PETTICREW, M.; ROBERTS, H. Systematic reviews in the social sciences: a practical guide. **Counselling and Psychotherapy Research**, v. 6, n. 4, p. 304–305, 2006. 217

PHARPATARA, P. **Trajectory planning for aerial vehicles with constraints**. Tese (Doutorado em Automática) — Université Paris-Saclay; Université d'Evry-Val-d'Essonne, Saint Aubin, 2015. 2, 5

PIEGL, L. **Fundamental developments of computer aided geometric design**. [S.l.]: San Diego, CA: Academic Press, 1993. 11

PROTTI, M.; BARZAN, R. **UAV autonomy - which level is desirable? - which level is acceptable? alenia aeronautica viewpoint**. [S.l.: s.n.], 2007. 17

QURESHI, A. H.; AYAZ, Y. Potential functions based sampling heuristic for optimal path planning. **Autonomous Robots**, v. 40, n. 6, p. 1079–1093, 2016. 8, 37, 47, 50, 53

REINSCH, C. H. Smoothing by spline functions. **Numerische Mathematik**, v. 10, n. 3, p. 177–183, 1967. 169

ROADMAP, U. **Unmanned aircraft systems roadmap 2005-2030**. [S.l.: s.n.], 2005. 18

SAKAHARA, H.; MASUTANI, Y.; MIYAZAKI, F. Real-time motion planning in unknown environment: voronoi-based stRRT (spatiotemporal RRT). In: SICE ANNUAL CONFERENCE, 2008. **Proceedings...** [S.l.]: IEEE, 2008. p. 2326–2331. 59, 64

SÁNCHEZ, A. A deterministic sampling approach to robot motion planning. In: MEXICAN INTERNATIONAL CONFERENCE ON COMPUTER SCIENCE, 4., 2003. **Proceedings...** [S.l.]: IEEE, 2003. p. 300–307. 9, 87

SCHWARTZ, J. T.; SHARIR, M. On the “piano movers” problem. ii. general techniques for computing topological properties of real algebraic manifolds. **Advances in Applied Mathematics**, v. 4, n. 3, p. 298–351, 1983. 27

SEEGMILLER, N.; GASSAWAY, J.; JOHNSON, E.; TOWLER, J. The maverick planner: an efficient hierarchical planner for autonomous vehicles in unstructured environments. In: INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS, 2017. **Proceedings...** [S.l.]: IEEE, 2017. p. 2018–2023. 69, 71

SEKHAVAT, S.; SVESTKA, P.; LAUMOND, J.-P.; OVERMARS, M. H. Multilevel path planning for nonholonomic robots using semiholonomic subsystems. **The International Journal of Robotics Research**, v. 17, n. 8, p. 840–857, 1998. 216

SIEGWART, R.; NOURBAKHSH, I. R.; SCARAMUZZA, D.; ARKIN, R. C. **Introduction to autonomous mobile robots**. [S.l.]: MIT press, 2011. 1

STONEMAN, S.; LAMPARIELLO, R. Embedding nonlinear optimization in RRT* for optimal kinodynamic planning. In: CONFERENCE ON DECISION AND CONTROL, 53., 2014. **Proceedings...** [S.l.]: IEEE, 2014. p. 3737–3744. 216

STRAUS, S.; RICHARDSON, S.; GLASZIOU, P.; HAYNES, R. B. **Evidence-based medicine: how to practice and teach EBM**. 3. ed. Edinburgh: Churchill Livingstone, 2005. 10

STUDENT. The probable error of a mean. **Biometrika**, p. 1–25, 1908. 150

STURTEVANT, N. R. Choosing a search space representation. In: RABIN, S. (Ed.). **Game AI Pro: collected Wisdom of game AI professionals**. [S.l.]: CRC Press, 2013. p. 253–258. 4, 21

SUCAN, I. A.; MOLL, M.; KAVRAKI, L. E. The open motion planning library. **IEEE Robotics & Automation Magazine**, v. 19, n. 4, p. 72–82, 2012. 167

SUKHAREV, A. G. Optimal strategies of the search for an extremum. **USSR Computational Mathematics and Mathematical Physics**, v. 11, n. 4, p. 119–137, 1971. 9, 85

SUN, H.; CHEN, X. Exponential backoff-sampling RRT for smart carpet. In: KIM, J.; YANG, W.; JO, J.; SINCAK, P.; MYUNG, H. (Ed.). **Robot intelligence technology and applications 3**. [S.l.]: Springer, 2015. p. 331–342. 60, 65

ŠVEC, P.; THAKUR, A.; RABOIN, E.; SHAH, B. C.; GUPTA, S. K. Target following with motion prediction for unmanned surface vehicle operating in cluttered environments. **Autonomous Robots**, v. 36, n. 4, p. 383–405, 2014. 2

TAHIR, Z.; QURESHI, A. H.; AYAZ, Y.; NAWAZ, R. Potentially guided bidirectionalized RRT* for fast optimal path planning in cluttered environments. **Robotics and Autonomous Systems**, v. 108, p. 13–27, 2018. 61, 67

THRUN, S.; BURGARD, W.; FOX, D. **Probabilistic robotics**. [S.l.]: MIT press, 2005. 5

TOZOUR, P. Search space representations. **AI Game Programming Wisdom**, v. 2, n. 1, p. 85–102, 2003. 4, 21

TSOURDOS, A.; WHITE, B.; SHANMUGAVEL, M. **Cooperative path planning of unmanned aerial vehicles**. [S.l.]: John Wiley & Sons, 2010. 4

UPADHYAY, S.; RATNOO, A. Continuous-curvature path planning with obstacle avoidance using four parameter logistic curves. **IEEE Robotics and Automation Letters**, v. 1, n. 2, p. 609–616, 2016. 4

VÉRAS, L. G.; MEDEIROS, F. L.; aES, L. F. G. Application of rapidly exploring random tree star-smart and G^2 quintic pythagorean hodograph curves to the UAV path planning problem. **International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering**, v. 12, n. 5, p. 522 – 530, 2018. ISSN eISSN:1307-6892. Disponível em: <http://waset.org/Publications?p=137>. 10, 11, 170, 177, 190

_____. _____. In: INTERNATIONAL CONFERENCE ON UNMANNED AIRCRAFT SYSTEMS, 2018. **Proceedings...** Amsterdã, Holanda: World Academy of Science, Engineering and Technology, 2018. 190

VÉRAS, L. G. D. O.; MEDEIROS, F. L. L.; aES, L. N. F. G. Rapidly exploring random tree* with a sampling method based on sukharev grids and convex vertices of safety hulls of obstacles. **International Journal of Advanced Robotic Systems**, v. 16, n. 1, p. 1729881419825941, 2019. Disponível em: <https://doi.org/10.1177/1729881419825941>. 85, 117, 118, 121, 122, 124

_____. Systematic literature review of sampling process in rapidly-exploring random trees. **IEEE Access**, v. 7, p. 50933 – 50953, 2019. Disponível em: <https://doi.org/10.1109/ACCESS.2019.2908100>. 11, 48, 51, 56, 62, 69, 73, 76, 80, 215, 218, 220, 222, 223

VISERAS, A.; LOSADA, R. O.; MERINO, L. Planning with ants: efficient path planning with rapidly exploring random trees and ant colony optimization. **International Journal of Advanced Robotic Systems**, v. 13, n. 5, p. 1729881416664078, 2016. Disponível em: <https://doi.org/10.1177/1729881416664078>. 80, 82

VONÁSEK, V.; KOZLÍKOVÁ, B. Tunnel detection in protein structures using sampling-based motion planning. In: INTERNATIONAL WORKSHOP ON ROBOT MOTION AND CONTROL, 11., 2017. **Proceedings...** [S.l.]: IEEE, 2017. p. 185–192. 61, 67

VORONOÏ, G. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. deuxième mémoire. recherches sur les paralléloèdres primitifs. **Journal für die Reine und Angewandte Mathematik**, v. 134, p. 198–287, 1908. 4

WANG, C.; MENG, M. Q.-H. Variant step size RRT: an efficient path planner for UAV in complex environments. In: INTERNATIONAL CONFERENCE ON REAL-TIME COMPUTING AND ROBOTICS, 2016. **Proceedings...** [S.l.]: IEEE, 2016. p. 555–560. 38

WANG, J.; LI, X.; MENG, M. Q.-H. An improved RRT algorithm incorporating obstacle boundary information. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND BIOMIMETICS, 2016. **Proceedings...** [S.l.]: IEEE, 2016. p. 625–630. 56, 57

WANG, J.; WU, S.; LI, H.; ZOU, J. Path planning combining improved rapidly-exploring random trees with dynamic window approach in ros. In: CONFERENCE ON INDUSTRIAL ELECTRONICS AND APPLICATIONS, 13., 2018. **Proceedings...** [S.l.]: IEEE, 2018. p. 1296–1301. 61, 67

WANG, X.; YADAV, V.; BALAKRISHNAN, S. Cooperative UAV formation flying with obstacle/collision avoidance. **IEEE Transactions on Control Systems Technology**, v. 15, n. 4, p. 672–679, 2007. 23

WANG, Y.; JIANG, S.; CHEN, B.; WU, H. Trajectory tracking control of underwater vehicle-manipulator system using discrete time delay estimation. **IEEE Access**, v. 5, p. 7435–7443, 2017. 2

WANG, Y.; JIANG, S.; YAN, F.; GU, L.; CHEN, B. A new redundancy resolution for underwater vehicle–manipulator system considering payload. **International**

Journal of Advanced Robotic Systems, v. 14, n. 5, p. 1729881417733934, 2017. 2

WELCH, B. L. The generalization of student's' problem when several different population variances are involved. **Biometrika**, v. 34, n. 1/2, p. 28–35, 1947. 158

XIA, Z.; CHEN, G.; XIONG, J.; ZHAO, Q.; CHEN, K. A random sampling-based approach to goal-directed footstep planning for humanoid robots. In: INTERNATIONAL CONFERENCE ON ADVANCED INTELLIGENT MECHATRONICS, 2009. **Proceedings...** [S.l.]: IEEE, 2009. p. 168–173. 50, 52

XIAO, S.; BERGMANN, N.; POSTULA, A. Parallel RRT* architecture design for motion planning. In: INTERNATIONAL CONFERENCE ON FIELD PROGRAMMABLE LOGIC AND APPLICATIONS, 27., 2017. **Proceedings...** [S.l.]: IEEE, 2017. p. 1–4. 216

XIE, C.; BERG, J. van den; PATIL, S.; ABBEEL, P. Toward asymptotically optimal motion planning for kinodynamic systems using a two-point boundary value problem solver. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2015. **Proceedings...** [S.l.]: IEEE, 2015. p. 4187–4194. 216

YANG, C.-Y.; YANG, J.-S.; LIAN, F.-L. Safe and smooth: mobile agent trajectory smoothing by SVM. **International Journal of Innovative Computing, Information and Control**, v. 8, p. 4959–4978, 2012. 4

YI, D.; THAKKER, R.; GULINO, C.; SALZMAN, O.; SRINIVASA, S. Generalizing informed sampling for asymptotically-optimal sampling-based kinodynamic planning via markov chain monte carlo. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2018. **Proceedings...** [S.l.]: IEEE, 2018. p. 7063–7070. 80, 83

YU, X.; TANG, X.; YE, B.; SONG, B.; ZHOU, X. Obstacle space modeling and moving-window RRT for manipulator motion planning. In: INTERNATIONAL CONFERENCE ON INFORMATION AND AUTOMATION, 2016. **Proceedings...** [S.l.]: IEEE, 2016. p. 539–544. 60, 66

YUNCHENG, L.; JIE, S. A revised gaussian distribution sampling scheme based on RRT* algorithms in robot motion planning. In: INTERNATIONAL CONFERENCE ON CONTROL, AUTOMATION AND ROBOTICS, 3., 2017. **Proceedings...** [S.l.]: IEEE, 2017. p. 22–26. 61, 67

ZHANG, J.; WISSE, M.; BHARATHEESHA, M. Guided RRT: a greedy search strategy for kinodynamic motion planning. In: INTERNATIONAL CONFERENCE ON CONTROL AUTOMATION ROBOTICS AND VISION, 13., 2014. **Proceedings...** [S.l.]: IEEE, 2014. p. 480–485. 69, 71, 216

ZHANG, L.; MANOCHA, D. An efficient retraction-based RRT planner. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2008. **Proceedings...** [S.l.]: IEEE, 2008. p. 3743–3750. 73, 74

ZHANG, X.; LüTTEKE, F.; ZIEGLER, C.; FRANKE, J. Self-learning RRT* algorithm for mobile robot motion planning in complex environments. In: MENEGATTI, E.; MICHAEL, N.; BERNS, Y.; YAMAGUCHI, H. (Ed.). **Intelligent Autonomous Systems 13**. [S.l.]: Springer, 2016. p. 57–69. 76, 77

ZHU, Q.; WU, Y.; WU, G.; WANG, X. An improved anytime RRTs algorithm. In: INTERNATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE AND COMPUTATIONAL INTELLIGENCE, 2009. **Proceedings...** [S.l.]: IEEE, 2009. v. 1, p. 268–272. 216

ZIVIANI, N. **Projeto de algoritmos**. [S.l.]: Nova Fronteira, 2004. 124

ZUCKER, M.; KUFFNER, J.; BRANICKY, M. Multipartite RRTs for rapid replanning in dynamic environments. In: INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION, 2007. **Proceedings...** [S.l.]: IEEE, 2007. p. 1603–1609. 50, 52

APÊNDICE A - CÁLCULO DO COMPRIMENTO ÓTIMO DE ROTAS COM DIJKSTRA E GRAFOS DE VISIBILIDADE

Neste apêndice, encontram-se os valores dos comprimentos das rotas ótimas para os ambientes de navegação dos testes computacionais descritos nas Seções 4.5.2 e 4.5.4. As rotas ótimas foram planejadas utilizando o algoritmo de grafos de visibilidade (NILSSON, 1969) e o algoritmo de Dikjstra (DIJKSTRA, 1959). O grafo de visibilidade é um algoritmo que conecta todos os vértices convexos visíveis entre si formando um mapa/grafo de rotas. Conectando os nós visíveis do grafo às posições inicial ($q_{início}$) e final ($q_{destino}$) de navegação, é possível obter a rota de menor comprimento entre ambas por um algoritmo de busca em grafo. O algoritmo de Dijkstra, que possui a propriedade de retornar a rota de menor comprimento entre dois nós de um grafo, é aplicado para tal.

As rotas ótimas foram planejadas para os seguintes ambiente de navegação: a) com um obstáculo; b) com cinco obstáculos; c) com 50 obstáculos; d) com 100 obstáculos; e) com 200 obstáculos; f) com obstáculo com formato em U; g) com obstáculo com formato em espiral; h) com obstáculos posicionados em zigue-zague; i) labirinto; j) com passagem estreita. As rotas ótimas planejadas para cada um desses ambientes de navegação encontram-se ilustradas na Figura A.1. O menor comprimento calculado para cada ambiente de navegação e suas respectivas posições $q_{início}$ e $q_{destino}$ estão listadas na Tabela A.1.

Tabela A.1 - Comprimentos das rotas ótimas planejadas pelos algoritmos de grafo de visibilidade e Dijkstra para diferentes ambientes de navegação e suas respectivas posições $q_{início}$ e $q_{destino}$.

Ambiente de navegação	$q_{início}$	$q_{destino}$	intervalo eixos	Comprimento da rota ótima
um obstáculo	(500, 500)	(925, 925)	x = [0, 1000]; y = [0, 1000]	1246,19
cinco obstáculos	(25, 25)	(975, 975)	x = [0, 1000]; y = [0, 1000]	1413,28
50 obstáculos	(25, 25)	(975, 975)	x = [0, 1000]; y = [0, 1000]	1393,12
100 obstáculos	(25, 25)	(975, 975)	x = [0, 1000]; y = [0, 1000]	1375,31
200 obstáculos	(25, 25)	(975, 975)	x = [0, 1000]; y = [0, 1000]	1392,98
obstáculo com formato em U	(500, 500)	(500, 100)	x = [0, 1000]; y = [0, 1000]	1376,78
obstáculo com formato em espiral	(500, 500)	(925, 925)	x = [0, 1000]; y = [0, 1000]	2302,14
obstáculos em zigue-zague	(500, 50)	(500, 950)	x = [0, 1000]; y = [0, 1000]	1006,45
labirinto	(25, 25)	(940, 940)	x = [0, 1000]; y = [0, 1000]	1790,25
passagem estreita	(25, 25)	(925, 925)	x = [0, 1000]; y = [0, 1000]	1581,25

Fonte: Produção do Autor.

Figura A.1 - Rotas ótimas planejadas pelo algoritmo de Dijkstra sobre grafos de visibilidade para diferentes ambiente de navegação.

Ambientes de navegação: a) com um obstáculo; b) com cinco obstáculos; c) com 50 obstáculos; d) com 100 obstáculos; e) com 200 obstáculos; f) com obstáculo com formato em U; g) com obstáculo com formato em espiral; h) com obstáculos posicionados em zigue-zague; i) labirinto; j) com passagem estreita.

Fonte: Produção do Autor.

APÊNDICE B - REVISÃO SISTEMÁTICA DA LITERATURA

Este anexo descreve o protocolo que orienta o processo de Revisão Sistemática da Literatura (RSL) com o objetivo de coletar evidências científicas na literatura de planejamento de rota, sobre as soluções de processos de amostragem não uniformes/informados aplicados a algoritmos que possuem como base o RRT. Este protocolo, bem como os resultados da execução da RSL, encontram-se publicados em Véras et al. (2019b).

B.1 Objetivo desta Revisão Sistemática da Literatura

Para encontrar trabalhos que ajudem a responder as questões científicas desenvolvidas para esta revisão, foi desenvolvida uma RSL para entender como o assunto é atualmente explorado. Dentro do contexto desse tipo de revisão, alguns conceitos são seguidos para definir os tipos de trabalhos disponíveis e quais deles devem ser incluídos nos resultados da revisão. Esses conceitos seguem as diretrizes declaradas em Kitchenham e Charters (2007). Os trabalhos são classificados da seguinte maneira: aqueles que apresentam métodos, resultados, procedimentos, tecnologia ou esquemas inovadores relacionados ao assunto da revisão são chamados de estudos primários; revisões e estudos comparativos são estudos secundários; as revisões de estudos secundários são consideradas estudos terciários. A RSL apresentada nesta tese considera apenas estudos primários como evidências de que uma solução dentro do escopo considerado foi proposta.

B.1.1 Definição do escopo das questões de pesquisa

O principal objetivo desta revisão é relatar o conjunto de evidências que mostram o que está sendo investigado sobre algoritmos baseados em RRT com processos de amostragem não uniformes/informados, através de uma revisão sistemática da literatura. Apenas estratégias de amostragem não uniformes/informadas propostas são consideradas. Durante a execução da RSL, uma estratégia de amostragem é considerada não uniforme/informada se gerar amostras aleatórias que não seguem uma distribuição uniforme, ou seja, elas são direcionadas de alguma forma. É importante ressaltar que as amostras aleatórias determinam a direção de crescimento da árvore do RRT. Desta forma, se as amostras forem influenciadas por uma amostragem não uniforme/informada, a árvore do RRT também será influenciada. Para ser selecionado na RSL, um estudo preliminar deve introduzir uma nova estratégia de amostragem não uniforme/informada com um dos seguintes objetivos: acelerar a convergência do planejamento para uma rota, reduzir o tempo de planejamento ou

otimizar alguma outra característica de algoritmos baseados em RRT. Para comparar e identificar se um estudo selecionado introduz alguma modificação que adiciona comportamento tendencioso em alguns algoritmos baseados no algoritmo RRT, o procedimento *RANDOM_STATE* do algoritmo RRT em Lavalle (1998) e *NOVO_-NÓ* no Algoritmo 1 são considerados como sendo a amostragem canônica para comparação com as selecionadas na RSL.

Modificações e otimizações em processos de algoritmos RRT que não sejam amostragem não serão consideradas na RSL, por exemplo, reconfiguração de arestas (CHOUDHURY et al., 2016; KARAMAN et al., 2011), paralelização (XIAO et al., 2017; DEVAURS et al., 2011), redução de nós por rejeições (DEVAURS et al., 2015; ZHU et al., 2009), otimizações de detecção de colisão (BIALKOWSKI et al., 2016; MEDEIROS et al., 2007) e otimização considerando dinâmica e cinemática (XIE et al., 2015; STONEMAN; LAMPARIELLO, 2014). Embora esses processos influenciem diretamente o desempenho de planejamento de algoritmos baseados em RRT, eles não estão diretamente relacionados ao processo de amostragem e são acoplados de forma independente de outros processos em algoritmos baseados em amostragem. Além disso, o assunto desta revisão é limitado apenas aos métodos aplicados aos processos de amostragem de algoritmos baseados em RRT. No caso de planejamento de rota com restrições dinâmicas e cinemáticas em algoritmos baseados em RRT, algumas estratégias propostas podem ser selecionadas na RSL. Problemas de planejamento de rota com restrições dinâmicas e cinemáticas são bastante complexos, já que muitas dimensões podem ser necessárias para representar seu espaço de busca, e nem todo estado pode ser considerado válido ou alcançado a partir de um anterior, como no caso de sistemas não holonômicos (SEKHAVAT et al., 1998). No entanto, tradicionalmente o processo de planejamento nesses casos é realizado rejeitando ou adaptando uma amostra aleatória para a geração de uma trajetória que atenda às restrições do movimento de um determinado sistema, ou suavização de rota (ELBANHAWI; SIMIC, 2014) e, desta forma será considerada uma estratégia de amostragem não uniforme/informada nesta revisão. Por outro lado, se o espaço de estados deste sistema for utilizado para modelar ou influenciar a distribuição das amostras aleatórias no RRT, então a estratégia proposta será considerada não uniforme/informada. O estudo apresentado em Zhang et al. (2014) corresponde à última situação descrita.

Diferentes mecanismos de busca de artigos científicos foram consultados para encontrar estudos relacionados ao tema. Os dados extraídos desses estudos foram comparados e analisados para identificar as principais contribuições do uso de amostragem não uniforme/informada como alternativa à amostragem uniforme, que é original-

mente utilizada na RRT. O protocolo de revisão que orienta o processo desta revisão encontra-se descrito na Seção B.2.

B.2 Protocolo de revisão

O protocolo de revisão que orienta o processo de coleta de evidências científicas é composto por seis elementos: questões de pesquisa; *strings* de pesquisa; consultas para busca de estudos primários; motores de busca; critérios de seleção para estudos primários; e, extração de dados (KITCHENHAM; CHARTERS, 2007).

B.2.1 Questões de pesquisa

As questões de pesquisa condicionam todo o processo de revisão, apresentando as principais discrepâncias sobre os métodos de amostragem não uniformes/informados usados nos algoritmos baseados no RRT para planejamento de rota. Para auxiliar no processo de definição das questões de pesquisa, utilizou-se a metodologia proposta em Petticrew e Roberts (2006), denominada *PICOC* (do inglês, *Population, Intervention, Comparator, Outcome, Context*). Para o problema do uso de processos de amostragem não uniformes/informados em RRTs, foram definidos os seguintes valores para cada parâmetro da metodologia PICOC:

a) **População**: algoritmos de planejamento de rota baseados em RRT.

b) **Intervenção**: Adição de estratégia não uniforme/informada na amostragem RRT para obter melhor custo/comprimento de rota e/ou convergência/tempo no processo de planejamento de rota.

c) **Comparação**: Estratégia não uniforme/informada usada no processo de amostragem do RRT.

d) **Resultados**: Trabalhos existentes na literatura sobre amostragem não uniforme/informada aplicada ao RRT e suas características melhoradas pela abordagem utilizada.

e) **Contexto**: pesquisa acadêmica em qualquer contexto de aplicação.

Assim, duas questões de pesquisa foram desenvolvidas para atender a esses parâmetros. Esses problemas são apresentados na Tabela B.1.

Tabela B.1 - Questões de pesquisa desenvolvidas na revisão sistemática da literatura.

Identificação	Questão de pesquisa	Descrição e intenção
Questão 1	Quais são as soluções propostas para incluir o comportamento não uniforme/informado no processo de amostragem do RRT encontrado na literatura de planejamento de rota?	Identificar os métodos propostos na literatura de planejamento de rota que implementam estratégias não uniformes/informadas sobre o processo de amostragem do algoritmo RRT ou qualquer uma de suas variantes (por exemplo, RRT*).
Questão 2	Qual é o algoritmo baseado no RRT para o qual a proposta de solução de amostragem não uniforme/informada é aplicada para melhorar seu desempenho de planejamento?	Esta pergunta ajuda a identificar as técnicas bases (RRT ou alguma de suas variantes) estendidas para gerar um novo algoritmo baseado em RRT não uniforme.

Fonte: Véras et al. (2019b).

A primeira pergunta é: "Quais são as soluções propostas para incluir o comportamento não uniforme/informado no processo de amostragem do RRT encontrado na literatura de planejamento de rota?". Esse é o principal problema considerado na revisão. Para descobrir a resposta, é necessário identificar estudos primários em que os problemas que os autores propõem encontrar soluções envolvam melhorar alguma propriedade (geralmente tempo e/ou custos/comprimento da rota (ELBANHAWI; SIMIC, 2014)) do RRT, usando como principal estratégia a mudança de seu processo de amostragem uniforme. Essa é a expectativa da revisão; no entanto, pode haver estudos com diferentes abordagens que se encaixem nos critérios de revisão.

A segunda pergunta é: "Qual é o algoritmo baseado em RRT ao qual uma solução de amostragem não uniforme/informada proposta é aplicada para melhorar seu desempenho de planejamento?". Esta questão está relacionada ao método a partir do qual a solução, proposta em um determinado estudo primário, foi originada. Os métodos podem ser o próprio RRT ou algumas de suas variantes (RRT*, DRRT, RRT-Connect, etc). A resposta a essa pergunta ajudará a identificar que tipo de RRT é usada como base dos estudos para reduzir a uniformidade de sua amostragem, se os novos aspectos que as variantes incluem são considerados ou se versões que já possuem tempo de planejamento reduzido são levadas em conta na literatura de planejamento de rota.

Os resultados da RSL obtidos a partir das questões de pesquisa acima permitirão a análise de quais soluções de amostragem não uniformes/informadas propostas para o RRT podem produzir os resultados mais promissores no planejamento de rotas, o que pode ajudar equipes de pesquisa em experimentos iniciais de estudos futuros. Por exemplo, uma estratégia de amostragem não uniforme/informada pode ter sido proposta para o RRT, mas sua eficiência na aceleração da convergência para uma rota pode não ter sido verificada junto com algum algoritmo baseado no algoritmo RRT*, que possui otimalidade assintótica. Analisando os resultados da RSL, uma equipe de pesquisa pode verificar o que já foi proposto como solução de amostragem não uniforme/informada para versões específicas do algoritmo RRT e identificar o que ainda não foi. Além disso, uma equipe de projeto tecnológico pode tomar conhecimento de quais estratégias existem através da RSL proposta e analisar quais delas podem preencher alguns requisitos de seu projeto.

B.2.2 *Strings* de pesquisa

Strings de pesquisa são termos que devem estar relacionadas ao problema científico de interesse. Dessa forma, a probabilidade de recuperar um trabalho relacionado ao assunto é maior. Cada *string* também é um componente da consulta de pesquisa aplicada aos mecanismos de pesquisa. Considerando as variações entre termos e palavras-chave semelhantes, as seguintes *strings*, em inglês, foram utilizadas neste trabalho: "Rapidly-exploring Random Tree", "Rapidly exploring Random Tree", "RRT", "Sampling", "Dispersion", "Path Planning", "Trajectory Plannin", "Motion Planning".

B.2.3 Consultas para pesquisa de estudos primários

Por meio de *strings* e questões de pesquisa definidas, as consultas foram formuladas para serem usadas nos mecanismos de pesquisa considerados nesta revisão. Uma consulta é a representação de uma pergunta de pesquisa em um formato adequado para os mecanismos de pesquisa usados.

As consultas são estruturadas usando *strings* de pesquisa e operadores lógicos, que são aceitáveis em todos os mecanismos de pesquisa considerados nesta revisão. Utilizando as *strings* de busca previamente definidas, duas consultas foram estruturadas para buscar os estudos primários pela RSL. Na Tabela B.2 estão descritas as consultas de pesquisa criadas para esta RSL. Todos os trabalhos/estudos primários retornados pela execução dessas consultas devem ser analisados e extraídos.

Tabela B.2 - Consultas de pesquisa estruturadas para a revisão sistemática

Identificação	Definição da consulta
Consulta 1	(“Rapidly Exploring Random Tree” OR “Rapidly-Exploring Random Tree” OR RRT) AND Dispersion
Consulta 2	(“Rapidly Exploring Random Tree” OR “Rapidly-Exploring Random Tree” OR RRT) AND Sampling AND (“Path Planning” OR “Trajectory Planning” OR “Motion Planning”)

Fonte: Véras et al. (2019b).

B.2.4 Mecanismos de busca considerados no processo de revisão

O processo de pesquisa dos estudos primários envolve a definição dos mecanismos de busca, que são as ferramentas que indexam a fonte de publicação destes trabalhos. Os estudos primários retornados por esses mecanismos são filtrados de acordo com as definições de inclusão e exclusão consideradas para a revisão e são descritos na Seção B.2.5. Na Tabela B.3 são listados os mecanismos de busca usados na RSL elaborado neste trabalho.

Tabela B.3 - Lista de mecanismos de busca como fonte de estudos primários para revisão sistemática da literatura

Nome do mecanismo de busca	Endereço *web*
Scopus	www.scopus.com
IEEEXplore	www.ieeexplore.ieee.org
ScienceDirect	www.link.springer.com
ACM Digital Library	www.dl.acm.org
Engineering Village	www.engineeringvillage.com

Fonte: Véras et al. (2019b).

B.2.5 Critérios para seleção dos estudos primários

Através dos resultados da consulta nos mecanismos de busca, é realizada uma filtragem de quais estudos estão relacionados às questões da pesquisa. Para orientar nessa tarefa, um critério de inclusão e um conjunto de critérios de exclusão foram definidos. Esses critérios ajudam a decidir qual trabalho é relevante ou não para a revisão. Os seguintes critérios de inclusão foram definidos para a revisão em questão:

a) Trabalhos acadêmico com contribuição real em planejamento de rota pelo algoritmo RRT. As soluções apresentadas devem propor melhorias no processo de amostragem do RRT através de estratégias não uniformes/informadas, com foco na redução do tempo de planejamento ou na melhoria de parâmetros e características relacionadas.

Em contraste com os critérios de inclusão, o conjunto de critérios de exclusão foi definido para decidir quais artigos retornados pelos mecanismos de busca não devem ser incluídos na revisão, mesmo que atendam aos critérios de inclusão. Por exemplo, documentos de pesquisa, *white papers*, literatura cinza, etc, podem ser retornados, porém, essa revisão não os inclui em seus resultados. Os critérios de exclusão definidos são:

a) trabalho que não apresenta solução aplicada ao planejamento de rota;

b) trabalho que não inclui nova solução para o processo de amostragem do algoritmo RRT;

c) trabalho que apresenta apenas comparação ou testes entre métodos baseados no RRT;

d) apresentação de documentos (slides);

e) estudos secundários e terciários;

f) trabalho que é uma versão antiga de outro, com os mesmos autores e problema;

g) trabalho que não está escrito em inglês;

h) apenas resumo;

i) resultados duplicados. Esses trabalhos são contados apenas uma vez;

j) *white paper* e literatura cinza.

B.2.6 Extração de dados

Para analisar e comparar os estudos primários retornados pela revisão, foi feita uma lista de dados/características que devem ser extraídos de cada um deles. Essas características dos estudos primários são: Título, Ano de publicação, *journal*, conferência,

congresso, simpósio ou oficina (*workshop*) onde as evidências foram publicadas; Autores; Métodos baseado no RRT utilizados como base para o desenvolvimento da solução apresentada; Método de amostragem utilizado; outros métodos baseados no RRT comparados com a solução apresentada para testar sua eficácia; completude; otimalidade; melhorias adquiridas resultantes da solução proposta.

B.3 Desenvolvimento da Revisão Sistemática da Literatura

Esta seção descreve o desenvolvimento da Revisão Sistemática da Literatura (SLR) com base no protocolo de revisão descrito anteriormente. Com a aplicação do protocolo de revisão, espera-se obter apenas os estudos relacionados ao escopo definido para responder às questões de pesquisa, reduzindo assim o viés causado pela visão pessoal do revisor recorrente em processos tradicionais de revisão. A Figura B.1 resume o processo de RSL desenvolvido em diferentes fases.

Figura B.1 - Processo de desenvolvimento da revisão sistemática da literatura.

Descrição das etapas - na 1ª fase, o escopo da revisão é delineado; - na 2ª fase, a primeira filtragem dos estudos primários retornados pelos mecanismos de busca é executada; - na 3ª fase, os dados dos estudos primários são extraídos e analisados para reportar os resultados.

Fonte: Véras et al. (2019b).

B.3.1 Descrição das fases da Revisão Sistemática da Literatura

A RSL está dividida em três fases: na primeira fase, o escopo da revisão é delineado por uma equipe de pesquisa; na segunda fase, a primeira filtragem dos estudos primários é retornada pelos mecanismos de busca; Na fase final, os dados dos estudos primários são extraídos e analisados. Em seguida, os resultados da revisão são divulgados por meio de um documento de relatório.

Na primeira fase, inicialmente as questões científicas de interesse são definidas conforme descrito na Seção B.2.1. O segundo passo foi desenvolver o protocolo de revisão. Isso deve ser construído em conjunto com uma equipe de especialistas no assunto analisado. As reformulações de protocolo antes de chegar a uma versão final podem ser realizadas. Essas reformulações são consequências de *insights* sobre como melhorar os resultados dos estudos retornados, obtidos nas etapas posteriores do desenvolvimento da revisão. A última etapa dessa fase é acessar a página *web* de cada mecanismo de pesquisa e inserir as consultas de pesquisa desenvolvidas com as *strings* de pesquisa em cada uma deles. Um conjunto de trabalhos preliminares de 1136 foram retornados, considerando a soma dos resultados de todos os mecanismos de busca.

A segunda fase começa eliminando os resultados duplicados. Os trabalhos que foram retornados mais de uma vez pelos mecanismos de busca foram removidos. Um total de 403 documentos duplicados foram identificados e removidos, resultando em 733 itens exclusivos. Na etapa seguinte, os critérios de inclusão e exclusão foram aplicados pela leitura do título, resumo e palavras-chave de cada trabalho. Os trabalhos resultantes da aplicação de tais critérios são considerados trabalhos com possíveis soluções relacionadas ao tema da revisão. Após essa fase, foram selecionados 143 trabalhos com potencial contribuição.

Na terceira e última fase, os dados foram extraídos e a revisão concluída. O primeiro passo desta fase foi obter acesso ao texto completo dos estudos selecionados. A segunda etapa consistiu na leitura completa de cada estudo para identificar e extrair os dados necessários para a revisão, conforme descrito na Subseção B.2.6. Após a extração de dados, os trabalhos que violaram os critérios de inclusão ou que atenderam ao conjunto de critérios de exclusão foram removidos da revisão. No final do processo da RSL, foram selecionados 53 estudos primários. Finalmente, o último passo foi divulgar os resultados da revisão através de algum relatório, que no caso desta revisão, é a publicação em Véras et al. (2019b) e este documento.

B.3.2 Validação da Revisão Sistemática da Literatura

Todo o processo de conduzir a RSL foi executado por apenas um dos autores da revisão. Os outros autores contribuíram no desenvolvimento do protocolo de revisão, trabalhando principalmente como supervisores. Assim, um viés na seleção dos estudos pode ser fortemente apresentado no processo realizado. Para identificar possíveis más interpretações sobre quais estudos atendem às questões científicas, foi realizado um processo de validação, como sugerido em Kitchenham e Charters (2007). Essa validação da RSL consiste em um processo de calibração das consultas de busca, identificando se os trabalhos mais citados/relevantes da literatura foram retornados no processo de revisão.

O processo de calibração de consultas é importante para garantir que os mecanismos de busca retornem artigos relevantes para as questões de pesquisa consideradas nesta revisão. As consultas iniciais foram realizadas e os resultados retornados foram analisados com base no tópico científico abordado. As consultas foram modificadas até que trabalhos relevantes pudessem ser retornados.

Foi identificado que estudos como (LINDEMANN; LAVALLE, 2006) e (LINDEMANN; LAVALLE, 2004) foram retornados. Estes são considerados importantes para o tema de amostragem no RRT pois, de acordo com seus autores, trazem as primeiras tentativas de discutir a importância da amostragem não uniforme/informada no algoritmo RRT, o que permite ter uma visão histórica das discussões sobre amostragem direcionada em algoritmos baseados em amostragem. Além disso, um dos mais recentes e bem-sucedidos algoritmos propostos com estratégia de amostragem não uniforme/informada, o algoritmo Informed-RRT* (GAMMELL et al., 2014), foi retornado com sucesso no processo de execução da RSL. Até mesmo algumas extensões deste algoritmo foram identificadas através da RSL. Como declarado em Noreen et al. (2016), um dos mais recentes trabalhos de revisão de algoritmos baseados no RRT, o algoritmo RRT*, tem sido pesquisado ativamente pela comunidade de planejamento de rotas nos últimos anos. Muitos trabalhos listados em Noreen et al. (2016) identificados como sendo algoritmos baseados no RRT* com novo processo de amostragem não uniforme/informada, foram também retornados durante a execução da RSL proposta. Ainda mais algoritmos baseados em RRT* com extensões de amostragem direcionadas foram identificados nesta revisão se comparados aos que estão listados em Noreen et al. (2016). Portanto, acredita-se que foram desenvolvidas consultas que cobrem consideravelmente as possibilidades de trabalhos sobre amostragem não uniforme/informada em algoritmos baseados no RRT.

ANEXO A - ALGORITMOS DA LITERATURA UTILIZADOS NOS TESTES COMPUTACIONAIS COM O ALGORITMO RRT*-SV

Os algoritmos RRT*-Smart, Informed-RRT* e BIT* foram selecionados para comparação com o algoritmo RRT*-SV, o qual foi proposto nesta tese. Portanto, eles são descritos em maiores detalhes nas seções seguintes.

A.1 RRT*-Smart

Proposto em Islam et al. (2012), o RRT*-Smart é um algoritmo baseado no RRT*, onde dois novos conceitos foram incorporados: o de otimização de rota; e o de amostragem inteligente. Foi apresentado em Islam et al. (2012) que esse algoritmo possui uma maior velocidade de convergência para a solução ótima (rota de menor comprimento entre duas posições), possível de ser obtida para um dado ambiente de navegação do que o RRT*. O algoritmo RRT*-Smart encontra-se descrito no Algoritmo 9. Os procedimentos adicionais do algoritmo RRT*-Smart estão descritos a seguir:

a) **OTIMIZAÇÃO_DE_ROTA**: os nós que formam a rota de $q_{início}$ à $q_{destino}$ que são visíveis entre si são conectados diretamente, removendo nós intermediários entre eles. Nós visíveis entre si são aqueles que podem formar uma nova aresta livre de colisão sem a necessidade de outros nós para tal além de ambos.

b) **COLETA_BEACONS**: coleta os nós da rota otimizada. Esses nós, denominados *beacons*, são utilizados para direcionar o processo de amostragem para próximo da última rota planejada.

A otimização de rota é uma estratégia baseada na desigualdade triangular, em que o tamanho do maior lado c de um triângulo sempre é menor que a soma dos tamanhos dos dois lados menores a e b (Figura A.1). Dessa forma, se existirem dois nós que possam ser conectados diretamente, sem que haja colisão, eles são conectados e os nós intermediários entre eles são ignorados. Assim, é obtida uma rota de comprimento menor. Essa estratégia encontra-se descrita no Algoritmo 10.

Algorithm 9 Algoritmo RRT*-Smart

```
1: procedure RRT(Q, q_início, q_destino, Δq, l_d, n, b)
2:     G ← {q_início}
3:     R ← {}
4:     i ← 0
5:     comprimento_antigo ← ∞
6:     while (i ≤ n) do
7:         i ← i + 1
8:         if i = t + b, t + 2b, t + 3b... then
9:             q_aleatório ← AMOSTRAGEM_ALEATÓRIA(Q_beacons)
10:        else
11:            q_aleatório ← AMOSTRAGEM_ALEATÓRIA(Q)
12:        q_próximo ← NÓ_PRÓXIMO(q_aleatório, G)
13:        q_novo ← NOVO_NÓ(q_próximo, q_aleatório, Δq)
14:        if q_próximo q_novo não intercepta Q_obstáculos then
15:            RRT_ESTRELA_ESTENDE(G, q_próximo, q_novo)
16:            if d(q_novo, q_destino) ≤ l_d e (q_novo q_destino não intercepta Q_obstáculos) then
17:                ESTENDE(G, q_novo, q_destino)
18:                rota_encontrada ← true
19:            if rota_encontrada then
20:                t ← i
21:                comprimento_novo ← OTIMIZAÇÃO_DE_ROTA(q_início, q_destino)
22:                if comprimento_novo < comprimento_antigo then
23:                    R ← ROTA(q_início, q_destino)
24:                    Q_beacons ← COLETA_BEACONS(q_início, q_destino)
25:                    comprimento_antigo ← comprimento_novo
26:            rota_encontrada ← false
```

Figura A.1 - Conceito da desigualdade triangular aplicada pelo algoritmo RRT*-Smart para otimizar rotas.

Fonte: Adaptado de Islam et al. (2012).

Algorithm 10 Procedimento de otimização de rota do algoritmo RRT*-Smart

1: **procedure** OTIMIZAÇÃO_DE_ROTA($q_{início}, q_{destino}$)
2: $q_{atual} \leftarrow q_{destino}$
3: **while** $q_{atual} \neq q_{início}$ **do**
4: $q_{antecessor_do_antecessor} \leftarrow ANTECESSOR_DO_ANTECESSOR(q_{atual})$
5: **if** $\overline{q_{atual} q_{antecessor_do_antecessor}}$ não intercepta $Q_{obstáculos}$ **then**
6: $q_{atual}.antecessor \leftarrow q_{antecessor_do_antecessor}$
7: $q_{atual} \leftarrow q_{antecessor_do_antecessor}$

A amostragem inteligente é uma abordagem que utiliza os nós de uma rota previamente planejada para induzir a coleta de novas amostras próximas a eles. Quando uma rota é planejada, seus nós são guardados em Q_{beacon}. Esses nós são atualizados cada vez que uma rota de comprimento menor do que a última planejada for encontrada. Novas amostras são coletadas a partir dos *beacons* quando o índice i da iteração de planejamento for igual a um valor dado por $t + u * b$, onde t é a iteração na qual se inicia o uso da amostragem inteligente, u é um valor incrementado cada vez que uma nova amostra é coletada pela amostragem inteligente e b uma constante que define a frequência com que essa amostragem é executada. Quando i não atende a condição citada, as amostras são coletadas aleatoriamente com distribuição uniforme, como nos algoritmos RRT e RRT*.

As duas estratégias descritas anteriormente foram incorporadas ao algoritmo RRT*, reduzindo consideravelmente o tempo de planejamento das rotas de menor comprimento (ISLAM et al., 2012). Na Figura A.2, encontram-se ilustradas rotas planejadas

pelo algoritmo RRT*-Smart. Nota-se que próximo aos nós da rota (em vermelho) existe uma maior concentração de amostras coletadas, mostrando a influência dos *beacons* na coleta de novas amostras do ambiente de navegação.

Figura A.2 - Exemplo de rota planejada pelo algoritmo RRT*-Smart.

a) Primeira rota planejada pelo algoritmo. b) A rota é então otimizada pelo procedimento de otimização de rota (resultado em azul). c) $Q_{beacons}$ é formado utilizando os nós da rota otimizada. d) Concentração de nós por meio dos *beacons* no procedimento de amostragem inteligente. O processo possui como efeito a otimização do comprimento da rota.

Fonte: Islam et al. (2012).

A.2 Informed-RRT*

No algoritmo Informed-RRT*, proposto em Gammell et al. (2014), um novo tipo de amostragem baseada na redução do ambiente de navegação é introduzido, como descrito na Seção 3.3.2.1. Essa redução é baseada em uma elipse, limitando a coleta de novas amostras que serão coletadas somente no interior dessa elipse. A área dessa elipse é então modificada com base no valor de comprimento da rota. Desta forma, quando novas rotas de menor comprimento são encontradas, a área da elipse diminui. Segundo os autores do algoritmo em Gammell et al. (2014), essa estratégia prioriza a região do ambiente de navegação na amostragem que contém as amostras com maior probabilidade de formar a rota ótima (rota de menor comprimento). Inicialmente, será descrito o processo de geração da elipse e, posteriormente, o acoplamento da estratégia de amostragem ao RRT* para formar o algoritmo Informed-RRT*. Na Figura A.3, encontra-se ilustrado as etapas principais de amostragem do algoritmo Informed-RRT*.

Figura A.3 - Processo de amostragem informada no algoritmo Informed-RRT*.

a) Após uma primeira rota ser planejada, todas as novas amostras são coletadas dentro de uma elipse. b) À medida que o comprimento da rota planejada diminui, a área da elipse também diminui. c) Na ausência de obstáculos no ambiente de navegação, a elipse converge para um segmento de reta.

Fonte: Gammell et al. (2014).

Para a descrição do algoritmo Informed-RRT*, as definições introduzidas em Gammell et al. (2014) foram adaptadas para este trabalho. Dada a posição inicial $q_{início}$ e a posição final $q_{destino}$, o comprimento mínimo possível entre ambos, na hipótese de não existirem obstáculos entre eles, é igual a $c_{ótimo}$, que corresponde à distância euclidiana entre essas duas posições. Durante o planejamento, a elipse pode ser expressa em termos do comprimento da última rota planejada, denominado $c_{corrente}$.

Os pontos focais da elipse serão $q_{início}$ e $q_{destino}$, o eixo maior corresponde a $c_{ótimo}$ e o eixo menor corresponde a $\sqrt{c^2_{corrente} - c^2_{ótimo}}$. A partir desses elementos define-se a elipse Q_{elipse}. Na Figura A.4 encontram-se ilustrados cada um dos elementos da elipse que reduz o ambiente de navegação.

Figura A.4 - Elementos da elipse que limita a coleta de novas amostras de um ambiente de navegação no algoritmo Informed-RRT*.

Fonte: Gammell et al. (2014).

Cada amostra da elipse Q_{elipse} é coletada aleatoriamente dada uma distribuição uniforme. Essa operação é realizada pela transformação de uma amostra $q_{círculo}$ coletada aleatoriamente de um círculo para a sua posição correspondente q_{elipse} na superfície da elipse. Essa transformação é definida por

$$q_{elipse} = \boldsymbol{C}\boldsymbol{L}q_{círculo} + q_{centro} \tag{A.1}$$

onde $\boldsymbol{C}$ é a matriz de transformação, $\boldsymbol{L}$ uma matriz diagonal preenchida com os eixos da elipse (matriz de rotação) e q_{centro} o ponto central da elipse, definida por $(q_{início} + q_{destino})/2$, que translada a amostra resultante para o sistema de referência do ambiente de navegação. A matriz de transformação pode ser calculada pela decomposição de Cholesky. Detalhes de como calcular $\boldsymbol{C}$ podem ser verificados em Gammell et al. (2014). Porém, esses procedimentos estão representados em pseudocódigo no Algoritmo 12.

Baseado nas definições anteriormente descritas, novos procedimentos são adicionados ao RRT* para o desenvolvimento do algoritmo Informed-RRT*. São eles:

a) **COLETA_AMOSTRA_INFORMADA**: coleta uma amostra aleatória dentro da região delimitada pela elipse definida pelos valores de $c_{corrente}$

e $c_{ótimo}$. Enquanto $c_{corrente}$ não é definido, ou seja, nenhuma rota foi planejada, a amostragem segue a coleta de amostras aleatória da RRT padrão, onde o ambiente de navegação completo Q é considerado;

b) **ROTAÇÃO_PARA_REFERÊNCIA**: retorna a matriz de rotação do sistema de referência da elipse alinhada com o sistema de referência do ambiente de navegação;

c) **COLETA_AMOSTRA_CÍRCULO**: coleta uma amostra aleatoriamente dada uma distribuição uniforme sobre a região delimitada por um círculo;

O processo de amostragem encontra-se descrito em maiores detalhes no Algoritmo 12. Esse processo é incorporado à estrutura do algoritmo RRT*, resultando na estrutura do algoritmo Informed-RRT*, descrita no Algoritmo 11.

Algorithm 11 Algoritmo Informed-RRT*

1: **procedure** INFORMED-RRT*$(Q, q_{início}, q_{destino}, \Delta q, l_d, n)$
2: $G \leftarrow \{q_{início}\}$
3: $R \leftarrow \{\}$
4: $i \leftarrow 0$
5: $comprimento_{antigo} \leftarrow \infty$
6: $c_{corrente} \leftarrow \infty$
7: **while** $(i \leq n)$ **do**
8: $i \leftarrow i + 1$
9: $c_{corrente} \leftarrow R.comprimento$
10: $q_{aleatório} \leftarrow COLETA_AMOSTRA_INFORMADA(q_{início}, q_{destino}, c_{corrente})$
11: $q_{próximo} \leftarrow NÓ_PRÓXIMO(q_{aleatório}, G)$
12: $q_{novo} \leftarrow NOVO_NÓ(q_{próximo}, q_{aleatório}, \Delta q)$
13: **if** $\overline{q_{próximo} q_{novo}}$ não intersecciona $Q_{obstáculos}$ **then**
14: $RRT_ESTRELA_ESTENDE(G, q_{próximo}, q_{novo})$
15: **if** $d(q_{novo}, q_{destino}) \leq l_d$ e ($\overline{q_{novo} q_{destino}}$ não intersecciona $Q_{obstáculos}$) **then**
16: $ESTENDE(G, q_{novo}, q_{destino})$
17: $comprimento_{novo} \leftarrow CUSTO(q_{início}, q_{novo})$
18: $rota_encontrada \leftarrow true$
19: **if** $rota_encontrada$ **then**
20: **if** $comprimento_{novo} < comprimento_{antigo}$ **then**
21: $R \leftarrow ROTA(q_{início}, q_{destino})$
22: $comprimento_{antigo} \leftarrow comprimento_{novo}$
23: $rota_encontrada \leftarrow false$

Algorithm 12 Descrição do procedimento que coleta amostras de forma aleatória dentro da região de uma elipse utilizada pelo algoritmo Informed-RRT*.

1: **procedure** COLETA_AMOSTRA_INFORMADA($q_{início}, q_{destino}, c_{corrente}$)
2: **if** $c_{corrente} < \infty$ **then**
3: $c_{ótimo} \leftarrow ||q_{destino} - q_{início}||$
4: $q_{centro} \leftarrow (q_{início} + q_{destino})/2$
5: $\boldsymbol{C} \leftarrow ROTAÇÃO_PARA_REFERÊNCIA(q_{início}, q_{destino})$
6: $r_1 \leftarrow c_{corrente}/2$
7: $r_2 \leftarrow (\sqrt{c^2_{corrente} - c^2_{ótimo}})/2$
8: $\boldsymbol{L} \leftarrow diag\{r_1, r_2\}$
9: $q_{círculo} \leftarrow COLETA_AMOSTRA_CÍRCULO()$
10: $q_{aleatório} \leftarrow (\boldsymbol{CL}q_{círculo} + q_{centro}) \cap Q$
11: **else**
12: $q_{aleatório} = AMOSTRAGEM_ALEATORIA()$
13: **retorna** $q_{aleatório}$

A execução do algoritmo Informed-RRT* se dá de forma semelhante ao algoritmo RRT*, se diferenciando apenas pela substituição do procedimento de amostragem padrão da RRT $AMOSTRAGEM_ALEATORIA$ por $COLETA_AMOSTRA_INFORMADA$. O procedimento está descrito no Algoritmo 12, onde também os demais procedimentos introduzidos anteriormente são utilizados.

A.3 Batch Informed Trees - BIT*

Nesta seção, o algoritmo BIT* (Batch Informed Tree Star), proposto em Gammell et al. (2015), é descrito. Esse algoritmo utiliza uma abordagem baseada no algoritmo de busca em grafos dinâmicos denominado LPA* (long-life A*) (KOENIG et al., 2004) e na amostragem baseada na região elipsoidal definida pelo comprimento das rotas planejadas do algoritmo Informed-RRT*. O algoritmo BIT* encontra-se descrito no Algoritmo 13.

Algorithm 13 Algoritmo BIT*

1: **procedure** BIT*$(Q, q_{início}, q_{destino}, n, m)$
2: $G \leftarrow \{q_{início}\}$
3: $G.V \leftarrow \{q_{início}\}$
4: $R \leftarrow \{\}$
5: $r \leftarrow \infty$
6: $s \leftarrow 0$
7: $i \leftarrow 0$
8: **while** $(s = 0)$ e $(i \leq n)$ **do**
9: $custo_{corrente} \leftarrow CUSTO(q_{início}, q_{destino})$
10: **if** $F_{arestas} = \{\}$ **e** $F_{vértices}$ **then**
11: $PODA(Q_{amostras}, custo_{corrente})$
12: $Q_{amostras} \leftarrow Q_{amostras} \cup LOTE_AMOSTRAS(m, custo_{corrente})$
13: $V_{antigo} \leftarrow G.V$
14: $F_{vértices} \leftarrow G.V$
15: $ORDENA(F_{vértices})$
16: $r \leftarrow raio(|G.V| + |Q_{amostras}|)$
17: **while** $MELHOR_VALOR_DA_FILA(F_{vértices}) \leq MELHOR_VALOR_DA_FILA(F_{arestas})$ **do**
18: $EXPANDE_VÉRTICES(MELHOR_DA_FILA(F_{vértices}))$
19: $(v_m, q_m) \leftarrow MELHOR_DA_FILA(F_{arestas})$
20: $F_{arestas} \leftarrow F_{arestas} \setminus \{(v_m, q_m)\}$
21: **if** $CUSTO(q_{início,v_m}) + \hat{c}(v_m, q_m) + \hat{h}(q_m) < custo_{corrente}$ **then**
22: $custo_{aresta} \leftarrow CUSTO_COM_COLISÃO(v_m, q_m)$
23: **if** $\hat{g}(v_m) + custo_{aresta} + \hat{h}(q_m) < custo_{corrente}$ **then**
24: **if** $CUSTO(q_{início,v_m}) + custo_{aresta} < CUSTO(q_{início,q_m})$ **then**
25: $v_{antecessor} \leftarrow \{\}$
26: **if** $q_m \in G.V$ **then**
27: $v_{antecessor} \leftarrow q_m.antecessor$
28: $REMOVE_ARESTA(G, v_{antecessor}, q_m)$
29: **else**
30: $Q_{amostras} \leftarrow Q_{amostras} \setminus q_m$
31: $F_{vértices} \leftarrow F_{vértices} \cup q_m$
32: $ESTENDE(G, v_m, q_m)$
33: **if** $v_{antecessor}$ **e** $(v_{antecessor}, q_m) \in F_{arestas}$ **then**
34: **if** $CUSTO(q_{início}, v_{antecessor}) + \hat{c}(v_{antecessor}, q_m) \geqslant CUSTO(q_{início}, v_{q_m})$ **then**
35: $F_{arestas} \leftarrow F_{arestas} \setminus (v_{antecessor}, q_m)$
36: **if** $q_{destino}$ possui novo nó antecessor **then**
37: $s \leftarrow 1$
38: $R \leftarrow ROTA(q_{início}, q_{destino})$
39: **else**
40: $F_{arestas} \leftarrow \{\}$
41: $F_{vértices} \leftarrow \{\}$

A seguir estão descritos os procedimentos que constituem o algoritmo BIT*:

a) **ORDENA**: ordena uma fila de vértices ou arestas em ordem crescente do comprimento estimado das rotas hipotéticas formadas por seus elementos;

b) **MELHOR_VALOR_DA_FILA**: retorna o menor comprimento estimado de uma rota hipotética de uma fila de vértices ou arestas;

c) **MELHOR_DA_FILA**: retorna o elemento de uma fila de vértices ou arestas com menor comprimento estimado de uma rota hipotética;

d) **CUSTO_COM_COLISÃO**: retorna o comprimento de uma aresta formada por duas amostras. Entretanto, a aresta formada é submetida a um teste de colisão. Caso a aresta esteja em colisão, o valor do comprimento da aresta é definido como infinito.

e) **CUSTO**: possui mesmo comportamento apresentado na descrição do algoritmo RRT*, onde o comprimento da rota entre dois vértices quaisquer de G é retornado;

f) **PODA**: remove as amostras de $Q_{amostras}$, remove vértices de $G.V$ e desconecta arestas de $G.E$ que não podem mais contribuir com o planejamento de rotas com menor comprimento. Aqueles vértices que foram desconectados em $G.E$ que ainda possuem potencial para constituir rotas de menor comprimento são removidos de $G.V$ e adicionados a $Q_{amostras}$;

g) **EXPANDE_VÉRTICE**: expande a fila $F_{arestas}$ com arestas formadas pelo vértice q_v e as amostras p da vizinhança $Q_{vizinhança}$ que estão a uma distância r de q_v. Uma aresta é adicionada a $F_{arestas}$ se, e somente se, a distância estimada da rota hipotética formada pela aresta (q_v, p) for menor do que o comprimento da rota corrente.

O algoritmo BIT* utiliza um conjunto de amostras geradas aleatoriamente denominado lote (do inglês *batch*), representado por $Q_{amostras}$. O uso desse esquema de lotes de amostras encontra-se ilustrado na Figura A.5. Um novo lote de amostras é gerada toda vez que as filas $F_{vértices}$ e $F_{arestas}$ encontrarem-se vazias. A quantidade de amostras do lote é definida pelo parâmetro m, e essas amostras definem implicitamente uma estrutura denominada Grafo Geométrico Aleatório (RGG, do inglês *Random Geometric Graph*) (PENROSE et al., 2003). Nesta estrutura, todos os nós vizinhos que estiverem dentro de uma região de raio r, a partir de um nó central, são

conectados a esse nó. Entretanto, no algoritmo BIT*, os nós/vértices do RGG não são interligados explicitamente. Em vez disso, o BIT* utiliza uma heurística baseada na estimativa do comprimento de uma rota que, hipoteticamente, os nós da RGG poderia compor para conectar $q_{início}$ à $q_{destino}$. Os vértices associados com as rotas hipotéticas de menor comprimento são preferencialmente submetidos ao teste de colisão e, caso permitam formar uma rota livre de colisão, são adicionados à árvore de exploração G. Dessa forma, evita-se a realização do teste de colisão em arestas que não podem reduzir o comprimento da rota, o que reduz consideravelmente o tempo de planejamento (GAMMELL et al., 2015).

Figura A.5 - Esquema do uso de lotes (*batchs*) de amostras utilizadas pelo algoritmo BIT*.

a) Crescimento da árvore no algoritmo BIT* utilizando o primeiro lote de amostras $Q_{vértices}$ e b) finalização do primeiro planejamento de uma rota quando $q_{início}$ é conectado à $q_{destino}$. Após o planejamento da primeira rota, um segundo lote de amostras é adicionado a partir de uma elipse e c) um novo planejamento é iniciado com um conjunto de amostras $Q_{vértices}$ mais denso. d) Após a finalização do segundo planejamento, uma nova região de amostragem (elipse) é definida considerando o comprimento da nova rota planejada. O processo se repete indefinidamente, toda vez que uma rota de menor comprimento é planejada.

Fonte: Gammell et al. (2015).

O comprimento estimado das rotas hipotéticas é calculado da seguinte forma. Considerando uma amostra qualquer $q \in Q$ do ambiente de navegação, o comprimento da rota hipotética que conecta diretamente q até $q_{início}$, é representada por $\hat{g}(q)$. Já o comprimento da rota hipotética que conecta diretamente q até $q_{destino}$ é representado por $\hat{h}(q)$. Portanto, o comprimento da rota hipotética entre $q_{início}$ e $q_{destino}$ que passa pela amostra q é representado por $\hat{f}(q) = \hat{g}(q) + \hat{h}(q)$. Dessa forma, o comprimento da rota hipotética representa o comprimento do segmento de reta entre uma das posições de início ou destino de navegação e uma amostra q caso não houvesse obstáculos entre elas. Como não há verificação de colisão com obstáculos para o cálculo do comprimento dessas rotas, diz-se que as mesmas são hipotéticas (em Gammell et al. (2015), os autores definem esse cálculo como sendo uma estimativa admissível do comprimento da rota). Tanto em Gammell et al. (2015) quanto nesta tese, a distância euclidiana é utilizada para a estimativa do comprimento das rotas hipotéticas.

A seleção das amostras no lote $Q_{amostras}$ é baseada nos valores de estimativa de comprimento de rotas hipotéticas definidas no parágrafo anterior. Para isso, o uso de duas filas ordenadas é proposto: a fila $F_{vértices}$, que contém os vértices da árvore G ordenados pelo comprimento da rota hipotética associada a cada um; e a fila $F_{arestas}$, que é formada a partir de $F_{vértices}$. $F_{vértices}$ é formada pelos vértices já adicionados à árvore G (linha 14 do Algoritmo 13), porém, ordenados de acordo com o valor $\hat{f}(q_v)$ associada a cada um dos vértices q_v. Por sua vez, $F_{arestas}$ contém possíveis pares de arestas (q, q_v) que podem ser utilizadas para expansão da árvore. Para formar $F_{arestas}$, antes a vizinhança de vértices $Q_{vizinhos}$ é extraída a partir das amostras em $Q_{amostras}$ que estão dentro do alcance do raio r a partir de q_v. A fila $F_{vertices}$ também é utilizada para formar a fila $F_{arestas}$ nas linhas 17 e 18 do Algoritmo 13. As arestas entre $q_v \in F_{vértices}$ e seus vizinhos p da vizinhança $Q_{vizinhos}$ que formarem rotas hipotéticas com comprimento menor do que o comprimento da rota corrente são adicionados à $F_{arestas}$ (esse processo é executado pelo procedimento $EXPANDE_VÉRTICES$ descrito no Algoritmo 14). Após a adição das arestas à fila de arestas, $F_{arestas}$ é ordenado de acordo com o valor $\hat{g}(v) + \hat{c}(v, q) + \hat{h}(q)$ associado a cada uma de suas arestas (v, q). Esse processo é executado enquanto $F_{vértices}$ contiver elementos com valores de comprimento de rota hipotética estimada menores do que os elementos de $F_{arestas}$.

Algorithm 14 Procedimento de expansão da fila de arestas $F_{arestas}$ do algoritmo BIT*

1: **procedure** EXPANDE_VÉRTICES($q_v \in F_{vértices}$)
2: $Q_{próximo} \leftarrow \{\}$
3: $V_{próximo} \leftarrow \{\}$
4: $F_{vértices} \leftarrow F_{vértices} \setminus q_v$
5: **for** $q \in Q_{amostras}$ **do**
6: **if** $DIST\hat{A}NCIA(q, q_v) \leq r$ **then**
7: $Q_{próximo} \leftarrow Q_{próximo} \cup q$
8: **for** $p \in Q_{próximo}$ **do**
9: **if** $\hat{g}(q_v) + \hat{c}(q_v, p) + \hat{h}(p) < CUSTO(q_{início}, q_{destino})$ **then**
10: $F_{arestas} \leftarrow F_{arestas} \cup (q_v, p)$
11: **if** $q_v \notin V_{antigo}$ **then**
12: **for** $w \in G.V$ **do**
13: **if** $DIST\hat{A}NCIA(w, q_v) \leq r$ **then**
14: $V_{próximo} \leftarrow V_{próximo} \cup w$
15: **for** $w \in V_{próximo}$ **do**
16: **if** $(q_v, w) \notin G.E$ e $\hat{g}(q_v) + \hat{c}(q_v, w) + \hat{h}(w) < CUSTO(q_{início}, q_{destino})$ e $CUSTO(q_{início}, q_v) + \hat{c}(q_v, w) < CUSTO(q_{início}, w)$ **then**
17: $F_{arestas} \leftarrow F_{arestas} \cup (q_v, w)$

A ordenação de $F_{arestas}$ permite que as arestas com menores comprimentos de rotas hipotéticas sejam posicionadas nas primeiras posições da fila. Dessas forma, as arestas mais promissoras em formar a rota de menor comprimento possuem preferência em serem usadas na tentativa de expansão da árvore de exploração. A aresta $(v_m, q_m) \in F_{arestas}$ com menor comprimento de rota hipotética associado é selecionada e removida de $F_{arestas}$ (linhas 19 e 20 do Algoritmo 13). Caso essa aresta esteja associada a um valor de comprimento de rota hipotética menor do que o comprimento da rota corrente e seja livre de colisão (linhas 21 à 24 do Algoritmo 13), ela é utilizada para expandir G (linhas 32 do Algoritmo 13). Enquanto $q_{destino}$ não possuir nó antecessor (ou seja, não estiver associado a nenhuma rota que o conecte à $q_{início}$), seu comprimento é definido como sendo infinito. As arestas utilizadas para expandir a árvore são removidas de $F_{arestas}$. Esse processo continua até que algum critério de parada seja satisfeito (que usualmente é o planejamento de uma primeira rota, quantidade de iterações ou tempo de planejamento).

O procedimento $CUSTO$ (utilizado nas linhas 9, 21, 24 e 34 do Algoritmo 13) possui a mesma função que no algoritmo RRT*, onde o comprimento da rota entre dois nós interligados em G é retornado. O procedimento $CUSTO_COM_COLISÃO$ (utilizado na linha 22 do Algoritmo 13) calcula o comprimento da aresta formada

entre dois vértices e submete a mesma ao teste de colisão (retornando infinito caso a aresta esteja em colisão com os obstáculos).

Após o planejamento da primeira rota, o novo lote de amostras que vier a ser gerado seguirá a amostragem informada do algoritmo Informed-RRT*, como descrito no Algoritmo 12. Além disso, todas as amostras de $Q_{amostras}$ que não podem formar rotas com menor comprimento do que a rota atual são removidas. Esse procedimento é executado por $PODA$ descrito no Algoritmo 15.

Algorithm 15 Procedimento de remoção do lote de amostras de $Q_{amostras}$ que não permitem mais planejar rotas de menor comprimento.

1: **procedure** PODA($custo_{corrente}$)
2: **for** $q \in Q_{amostras}$ **do**
3: **if** $\hat{f}(q) \geq custo_{corrente}$ **then**
4: $Q_{amostras} \leftarrow Q_{amostras} \setminus q$
5: **for** $v \in G.V$ **do**
6: **if** $\hat{f}(v) \geq custo_{corrente}$ **then**
7: $G.V \leftarrow G.V \setminus v$
8: **for** $(v, w) \in G.E$ **do**
9: **if** $\hat{f}(v) \geq custo_{corrente}$ ou $\hat{f}(w) \geq custo_{corrente}$ **then**
10: $G.E \leftarrow G.E \setminus (v, w)$
11: **for** $v \in G.V$ **do**
12: **if** $CUSTO(q_{início}, v) \equiv \infty$ **then**
13: $Q_{amostras} \leftarrow Q_{amostras} \cup v$
14: **for** $v \in G.V$ **do**
15: **if** $CUSTO(q_{início}, v) \equiv \infty$ **then**
16: $G.V \leftarrow G.V \setminus v$